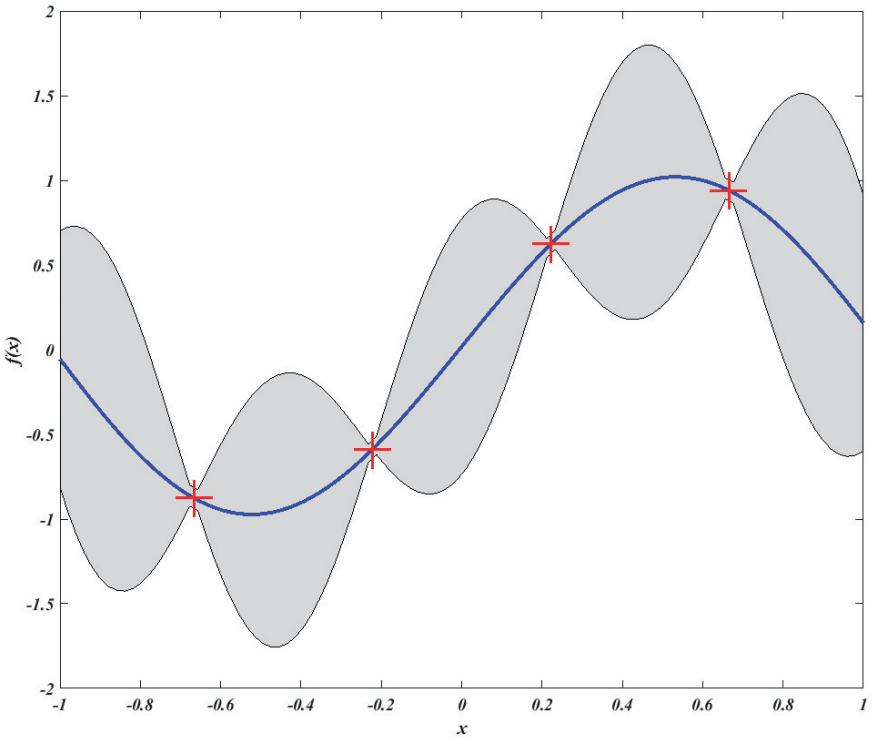
JYVÄSKYLÄ STUDIES IN COMPUTING
263

Tinkle Chugh

Handling Expensive Multiobjective Optimization Problems with Evolutionary Algorithms



JYVÄSKYLÄN YLIOPISTO

JYVÄSKYLÄ STUDIES IN COMPUTING 263

Tinkle Chugh

Handling Expensive Multiobjective Optimization Problems with Evolutionary Algorithms

Esitetään Jyväskylän yliopiston informaatioteknologian tiedekunnan suostumuksella julkisesti tarkastettavaksi yliopiston Agora-rakennuksen Lea Pulkkisen salissa kesäkuun 19. päivänä 2017 kello 12.

Academic dissertation to be publicly discussed, by permission of the Faculty of Information Technology of the University of Jyväskylä, in building Agora, Lea Pulkkinen hall, on June 19, 2017 at 12 o'clock noon.

UNIVERSITY OF JYVÄSKYLÄ

JYVÄSKYLÄ 2017

Handling Expensive Multiobjective Optimization Problems with Evolutionary Algorithms

JYVÄSKYLÄ STUDIES IN COMPUTING 263

Tinkle Chugh

Handling Expensive Multiobjective Optimization Problems with Evolutionary Algorithms

UNIVERSITY OF JYVÄSKYLÄ

JYVÄSKYLÄ 2017

Editors
Timo Männikkö
Faculty of Information Technology, University of Jyväskylä
Pekka Olsbo, Ville Korkiakangas
Publishing Unit, University Library of Jyväskylä

Cover picture by Tinkle Chugh.

Permanent link to this publication: http://urn.fi/URN:ISBN:978-951-39-7090-1

URN:ISBN:978-951-39-7090-1
ISBN 978-951-39-7090-1 (PDF)

ISBN 978-951-39-7089-5 (nid.)
ISSN 1456-5390

Jyväskylä University Printing House, Jyväskylä 2017

ABSTRACT

Chugh, Tinkle
Handling expensive multiobjective optimization problems with evolutionary algorithms
Jyväskylä: University of Jyväskylä, 2017, 66 p. (+included articles)
(Jyväskylä Studies in Computing
ISSN 1456-5390; 263)
ISBN 978-951-39-7089-5 (nid.)
ISBN 978-951-39-7090-1 (PDF)
Finnish summary
Diss.

Multiobjective optimization problems (MOPs) with a large number of conflicting objectives are often encountered in industry. Moreover, these problem typically involve expensive evaluations (e.g. time consuming simulations or costly experiments), which pose an extra challenge in solving them. In this thesis, we first present a survey of different methods proposed in the literature to handle MOPs with expensive evaluations. We observed that most of the existing methods cannot be easily applied to problems with more than three objectives. Therefore, we propose a Kriging-assisted reference vector guided evolutionary algorithm (K-RVEA) for problems with at least three expensive objectives. The algorithm dynamically balances between convergence and diversity by using reference vectors and uncertainty information from the Kriging models.

We demonstrate the practicality of K-RVEA with an air intake ventilation system in a tractor. The problem has three expensive objectives based on time consuming computational fluid dynamics simulations. We also emphasize the challenges of formulating a meaningful optimization problem reflecting the needs of the decision maker (DM) and connecting different pieces of simulation tools. Furthermore, we extend K-RVEA to handle constrained MOPs. We found out that infeasible solutions can play a vital role in the performance of the algorithm.

In many real-world MOPs, the DM is usually interested in one or a small set of Pareto optimal solutions based on her/his preferences. Additionally, it has been noticed in practice that sometimes it is easier for the DM to identify non-preferable solutions instead of preferable ones. Therefore, we finally propose an interactive simple indicator-based evolutionary algorithm (I-SIBEA) to incorporate the DM's preferences in the form of preferable and/or non-preferable solutions. Inspired by the involvement of the DM, we briefly introduce a version of K-RVEA to incorporate the DM's preferences when using surrogates. By providing efficient algorithms and studies, this thesis will be helpful to practitioners in industry and increases their ability of solving complex real-world MOPs.

Keywords: many-objective optimization, decision making, Pareto optimality, surrogate, metamodelling, computational cost

Author

Tinkle Chugh
Faculty of Information Technology
University of Jyväskylä
Finland

Supervisors

Professor Kaisa Miettinen
Faculty of Information Technology
University of Jyväskylä
Finland

Professor Yaochu Jin
Faculty of Information Technology
University of Jyväskylä
Finland
Department of Computer Science
University of Surrey
United Kingdom

Dr. Karthik Sindhya
Faculty of Information Technology
University of Jyväskylä
Finland

Dr. Jussi Hakanen
Faculty of Information Technology
University of Jyväskylä
Finland

Reviewers

Prof. Richard Everson
Department of Computer Science
University of Exeter, United Kingdom

Prof. Kyriakos C. Giannakoglou
School of Mechanical Engineering
National Technical University of Athens, Greece

Opponent

Associate Prof. Michael Emmerich
Leiden Institute of Advanced Computer Science
Leiden University, The Netherlands

ACKNOWLEDGEMENTS

First, I would like to thank my supervisors Prof. Kaisa Miettinen, Prof. Yaochu Jin, Dr. Karthik Sindhya and Dr. Jussi Hakanan for their constant guidance and support during my Ph.D. studies. Their knowledge and expertise, especially in multiobjective optimization helped me to gain the confidence in the field. It was a great opportunity to work in the "Decision Support for Complex Multiobjective Optimization Problems (DeCoMo)" project with Finland Distinguished Professor (FiDiPro) Yaochu Jin. During this project, we came across several real-world problems and collaborated with people from industry, which provided me an insight to solve complex multiobjective optimization problems.

In addition, I would like to thank my international colleagues, Tomas Kratky from Center of Hydraulic Research, Czech Republic, Pekka Makkonen from Valtra Inc. Finland and Prof. Nirupam Chakraborti from Indian Institute of Technology Kharagpur, India for their valuable collaboration. For reviewing my dissertation, I would like to thank Prof. Kyriakos C. Giannakoglou from National Technical University of Athens, Greece and Prof. Richard Everson from University of Exeter, United Kingdom. Suggestions given by them were useful in improving the thesis.

Furthermore, I appreciate my colleagues and friends, in particular, Vesa Ojalehto, Yue-Zhou Kangas and Mohammad Tabatabaei at the industrial optimization group for useful discussions. The discussion about surrogate-assisted optimization in the research group at the University of Surrey was also fruitful. I would like to thank Doctoral Program in Computing and Mathematical Sciences (COMAS) and Tekes: Finnish Funding Agency for Innovation for supporting my Ph.D. thesis. Last but not the least, I thank my family members, especially my parents Mrs. Vidya Wati and Mr. Chunni Lal Chugh for their blessings and support.

LIST OF FIGURES

CONTENTS

LIST OF INCLUDED ARTICLES

PI Tinkle Chugh, Karthik Sindhya, Jussi Hakanen and Kaisa Miettinen. A survey on handling computationally expensive multiobjective optimization problems with evolutionary algorithms. *Submitted to a journal*.

PII Tinkle Chugh, Yaochu Jin, Kaisa Miettinen, Jussi Hakanen, Karthik Sindhya. A Surrogate-assisted Reference Vector Guided Evolutionary Algorithm for Computationally Expensive Many-objective Optimization. *IEEE Transactions on Evolutionary Computation, to appear, doi:10.1109/TEVC.2016.262230*.

PIII Tinkle Chugh, Karthik Sindhya, Kaisa Miettinen, Yaochu Jin, Tomas Kratky, Pekka Makkonen. Surrogate-assisted evolutionary multiobjective shape optimization of an air intake ventilation system. *In Proceedings of IEEE Congress on Evolutionary Computation, IEEE, to appear*.

PIV Tinkle Chugh, Karthik Sindhya, Kaisa Miettinen, Jussi Hakanen, Yaochu Jin. On Constraint Handling in Surrogate-Assisted Evolutionary Many-Objective Optimization. *In 14th International Conference on Parallel Problem Solving from Nature, 2016 Proceedings, Edited by J. Handl et al., 214-224, Springer, 2016*.

PV Tinkle Chugh, Karthik Sindhya, Jussi Hakanen, Kaisa Miettinen. An Interactive Simple Indicator-Based Evolutionary Algorithm (I-SIBEA) for Multiobjective Optimization Problems. *Evolutionary Multi-Criterion Optimization: 8th International Conference, Proceedings, Part II, Edited by A. Gaspar-Cunha, C. Antunes, C. Coello, Springer, Berlin, Heidelberg, 277-291, 2015*.

1 INTRODUCTION

Many industrial optimization problems have several characteristics which introduce various challenges in solving them. One of the challenges is to deal with a large number of conflicting objectives to be optimized simultaneously. Moreover, these objectives are evaluated with costly experiments and/or time consuming simulations. For example, in designing an aircraft [79], several objectives e.g. minimizing the takeoff noise, weight of the aircraft, operating cost and maximizing speed and lift need to be optimized. Such problems with conflicting objectives are known as multiobjective optimization problems (MOPs) and for these problems there is no single optimal solution but typically multiple optimal solutions exist. These solutions are known as Pareto optimal (PO) solutions and they represent different trade-offs between the objectives.

Numerous methods have been proposed in the literature to solve MOPs [23, 59]. Evolutionary multiobjective optimization (EMO) methods based on evolutionary algorithms (EAs) have become popular and been widely used in past few decades [18, 23]. They start with a population of individuals which evolve by using genetic operators e.g reproduction, crossover and mutation. This thesis focuses on using evolutionary algorithms because of their wide applicability and certain advantages. For example, they do not assume any convexity or differentiability of the objective functions involved and can easily deal with the problems with locally optimal solutions.

Despite of several advantages of EAs, they have one major limitation that they can consume many function evaluations to find an approximated set of PO solutions [38, 39]. This issue becomes more prominent when the problem to be solved involves expensive functions which is another challenge in solving industrial optimization problems. For instance, in designing a drug [45], an engineer or a decision maker (DM) who is an expert in the problem domain needs to do some lab experiments to obtain a product of a desired quality and these experiments are usually costly with respect to both time and money. Another example is designing a cooling system of a tractor [PIII], where the DM wants to maximize the cooling efficiency and at the same time minimize the cost. In this case, the DM needs to do complex computational fluid dynamics (CFD) simulations with

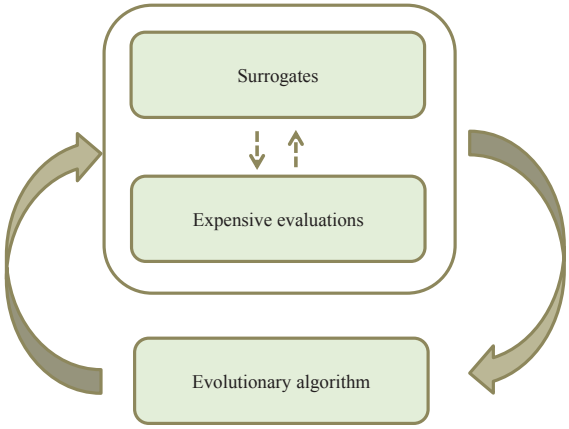


FIGURE 1 An illustration of using surrogates with an evolutionary algorithm

specific design (decision) variables. These simulations are also usually very time consuming. In both the examples, the DM needs to perform expensive experiments/simulations to obtain the solutions. In such problems, using evolutionary algorithms may not be a viable alternative or otherwise, they need to be adapted to make them more useful for real-world applications.

To obtain solutions for expensive problems in a limited number of expensive function evaluations, surrogates (or metamodels or response surface approximations) have been used in the literature as an alternative to expensive evaluations [26, 39, 46]. Several surrogate-assisted evolutionary algorithms (SAEAs) for multiobjective optimization have been proposed in the literature e.g. ParEGO [44], SMS-EGO [71], MOEA/D-EGO [93] and K-RVEA [PII]. The basic idea of using surrogates with an EA is shown in Figure 1. The approximations from the surrogates are used with EA instead of doing expensive evaluations. Although the philosophy of using surrogates sounds very simplistic, it has certain challenges. The most important challenge is to efficiently manage the surrogates. Surrogate management or evolution control [40, 38] is very important in the performance of a SAEA which includes how to select training samples for building the surrogates, when and how to update the surrogates. Many SAEAs exist in the literature and most of them do not address the following three challenges:

1. how to handle a large number of objectives,
2. how to reduce the training time of surrogates, and
3. how to handle constraints ?

Most SAEAs in the literature have not been developed to handle problems with more than three objectives [PI] which means they are not applicable to solve many real-world problems. One of the main reasons of restraining the use of the existing SAEAs for a limited number of objectives is the strategy for surrogate management. In managing the surrogates, samples for training the surrogates should be selected considering both convergence and diversity which is not trivial in solving problems with a large number of objectives.

Another concern is the training time for the surrogates which can be substantial and may even be longer than evaluating the original objective functions. Many SAEAs in the literature usually ignore the aspect of training time of the surrogates. Additionally, most SAEAs in the literature do not consider constrained problems which also raises a question for using them in real-world cases. This thesis focuses on dealing with three aspects in solving an expensive MOP: 1) finding a representative set of PO solutions with a large number of conflicting objectives, 2) selecting an appropriate set of samples for training the surrogates to further reduce the computation time and 3) handling constraints by training the surrogates with appropriate set of samples considering their feasibility.

The thesis is a collection of five articles (PI-PV) published in scientific journals and conference proceedings which are introduced in Chapters 3-7 in this thesis. In [PI] introduced in Chapter 3, a survey of several articles published before the year 2016 to handle expensive MOPs is presented. In this article, we summarize several challenges in applying SAEAs including challenges related to the nature of the problem to be solved and give some guidelines to select an algorithm to solve a given MOP. As mentioned, one of the limitations of existing SAEAs is their applicability for more than three objectives. Therefore, in [PII] introduced in Chapter 4, we propose a Kriging based reference vector guided evolutionary algorithm called K-RVEA for three or more objective functions.

The algorithm K-RVEA uses elements from its underlying evolutionary algorithm RVEA [16] and Kriging models [27] to approximate the objective function values. To efficiently manage the Kriging models, a set of reference vectors is used for selecting the samples. Additionally, the convergence criterion in RVEA called angle penalized distance and the uncertainty information from the Kriging models are used with the reference vectors to enhance convergence and diversity. Another feature of K-RVEA is to limit the size of samples for training the surrogates. These samples are also selected with the help of reference vectors. In [PII], the potential of the algorithm is shown on several benchmark problems and a free-radical polymerization problem [64] by comparing with the state-of-the-art SAEAs.

In [PIII] introduced in Chapter 5, K-RVEA is applied to an air intake ventilation system problem in an automobile industry which involves CFD simulations. We focus on three main challenges, the formulation of the optimization problem, connecting different simulation tools and dealing with time consuming simulations. Most of SAEAs in the literature have been tested on benchmark or academic MOPs, where problems actually do not involve expensive evaluations. The formulation of the objective functions in those problems is known a priori but in contrast in many real-world problems, the formulation of the optimization problem is not straightforward and typically takes many iterations of discussions with the DM and an analyst of the optimization algorithm. Additionally, it may take a considerable amount of effort to connect different simulation tools to obtain objective function values. We consider these issues in [PIII] before using K-RVEA to solve the problem.

The K-RVEA algorithm in [PII] has been developed for problems with only

box constraints. However, many real-world problems have constraints which poses an extra challenge in using a SAEA. Therefore, in [PIV] introduced in Chapter 6, we study three different ways of training the surrogates considering the feasibility of solutions. In the first one, only feasible solutions are used for training, in the second one, some infeasible solutions are used in addition to feasible ones and in the third one, a penalty is assigned to infeasible solutions. Moreover, three different types of penalty are used: fixed, adaptive and parameter free penalty. We found out that infeasible solutions can play an influential role in the performance of surrogates.

In real-world problems, the DM is usually interested in solutions based on her/his preferences. Therefore, finding a small set of PO solutions desirable to the DM may also reduce the number of expensive evaluations. To incorporate the DM's preferences in the solution process, we first propose an algorithm called I-SIBEA in [PV] which is introduced in Chapter 7. In I-SIBEA, the DM iteratively provides her/his preference in selecting preferred and/or non-preferred solutions. The preferences from the DM are then used in calculating the weighted hypervolume and solutions are selected based on their contribution to the weighted hypervolume. In [PV], the I-SIBEA is tested on benchmark problems with 2-3 objectives with different numbers of interactions with the DM. However, the potential of the algorithm is also shown on a 10 objective problem in Chapter 7. Additionally, a preliminary discussion and results on benchmark problems with 3-10 objectives by incorporating the DM's preferences in K-RVEA is presented in Chapter 7.

In this thesis, by identifying advantages and limitations of existing SAEAs in [PI], we propose the K-RVEA algorithm in [PII] to tackle the challenge of solving a MOP with three or more number of expensive objectives. The practicality of the algorithm is shown on a real-world problem of air intake ventilation system in [PIII]. Considering the constrained problems, we modified K-RVEA to make it adaptable for dealing with the infeasible solutions in the presence of constraints in [PIV]. Finally, the preferences from the DM is involved in the solution process in [PV] to find a single or small set of PO solutions desirable to her/him.

The rest of the thesis is organized as follows. In Chapter 2, we introduce the main concepts and terms used in this thesis. A survey of different SAEAs proposed within 2008-2016 with their characteristics, advantages and limitations is presented in Chapter 3. In Chapter 4, the K-RVEA algorithm is detailed followed by its application to solve a real-world problem of an air intake ventilation system in Chapter 5. In Chapter 6, a study is performed with K-RVEA to deal with constrained problems. In Chapter 7, we describe the algorithm I-SIBEA and discuss the initial results of incorporating DM preferences in K-RVEA. The author's contributions are described in Chapter 8. Finally, we conclude and discuss the future research directions in Chapter 9.

2 BACKGROUND

This chapter provides the definitions of the most commonly used terms in this thesis. We also provide a brief description of RVEA [16] and Kriging [27] as they are two major building blocks in the development of K-RVEA.

2.1 Problem statement

We consider multiobjective optimization problems (MOPs) of the form:

$$\begin{array}{ll} \text{minimize} & \{f_1(x), \ldots, f_k(x)\} \\ \text{subject to} & x \in S \end{array} \tag{1}$$

with $k(\geq 2)$ objective functions $f_i(x) : S \rightarrow \Re$. The vector of objective function values is denoted by $f(x) = (f_1(x), \ldots, f_k(x))^T$. The (nonempty) feasible region S is a subset of the decision variable space $\Re^n$ and consists of decision variable vectors $x = (x_1, \ldots, x_n)^T$ that satisfy all the constraints. The image of the feasible region S in the objective space $\Re^k$ is called the feasible objective set denoted by Z. The elements of Z are called feasible objective vectors denoted by $f(x)$ or $z = (z_1, \ldots, z_k)^T$, where $z_i = f_i(x), i = 1, \ldots, k$. For the simplicity of presentation, we assume that all the objective functions are to be minimized. If some objective function f_i is to be maximized, it is equivalent to minimize $-f_i$. In this thesis, we consider the MOPs with at least three or more objective functions and assume that all the objective functions are expensive to evaluate.

As mentioned in Chapter 1, there is typically no single optimal solution but multiple optimal solutions exist for a MOP. A decision vector $x^* \in S$ is Pareto optimal if there does not exist another decision vector $x \in S$ such that $f_i(x) \leq f_i(x^*)$ for all i=1,...,k and $f_j(x) < f_j(x^*)$ for at least one index j. An objective vector is Pareto optimal if the corresponding decision vector is Pareto optimal. A Pareto optimal set consists of all Pareto optimal solutions in the decision space and a Pareto front consists of all Pareto optimal solutions in the objective space.

2.2 Terminologies in using surrogates

In using SAEAs, different terms have been used in the literature and one can observe that different terms are used for describing the same concept e.g. metamodels or response surface approximations are used as synonyms for surrogates. We define here the most commonly used terms in using a SAEA for solving expensive MOPs. These terminologies will be used in the rest of the this thesis.

1. **Surrogate:** A surrogate or a metamodel approximates the expensive elements which are usually the objective functions in solving a MOP. Neural networks [28], radial basis functions [12], support vector regression [13] and Kriging [27] are some examples of commonly used surrogate techniques.

2. **Ensemble of surrogates:** An ensemble of surrogates refers to using more than one surrogate. In using an ensemble of surrogates, the characteristics of different surrogate techniques can be utilized simultaneously. There can be different ways to build an ensemble of surrogates. For instance, in [82], different surrogates are trained with the same samples and the one with the highest accuracy is chosen to be used in the solution process. In contrast, in [54], a weighted sum of approximations from different surrogates is used.

3. **Surrogate management:** Surrogate management or evolution control [38] is a strategy to manage the surrogates in an EA. For instance, selecting samples for training, that is, fitting a surrogate, when and how to update the surrogates are major components in the surrogate management. Selected samples also affect the performance of the evolutionary algorithm used. Therefore, managing the surrogate is very important in the performance of a SAEA.

4. **Infill criterion:** An Infill criterion or an updating criterion is the most important part of the surrogate management. Surrogates often need to be trained several times with expensive function evaluations and selecting samples for training is critical for their performance. An efficient infill criterion ensures that samples are selected in such a way that both convergence and diversity are taken into account. The most widely used infill criteria are expected improvement (EI) [41, 44], probability of improvement [20], expected hypervolume improvement [26] and lower confidence bound [71].

5. **Fitness inheritance:** In fitness inheritance [77], the objective function values of new samples are approximated based on their similarity to the already evaluated ones. For instance, if the sample to be evaluated has decision variable values similar to the already evaluated one, the corresponding values of the objective functions can also be assumed to be similar to the previously evaluated ones without doing any expensive evaluation.

2.3 A reference vector guided evolutionary algorithm

The reference vector guided evolutionary algorithm (RVEA) [16] is used in this thesis as an underlying algorithm for the development of the surrogate based algorithm K-RVEA. RVEA is a recently proposed algorithm to handle MOPs with a large number of objectives (or many objectives). In contrast to conventional dominance based EAs e.g. NSGA-II [23] and SPEA2 [96], RVEA uses an adaptive set of reference vectors to guide the solutions towards the Pareto front. In conventional EAs, the selection is usually based on dominance which is not suited for dealing with a large number of objectives. In RVEA, a selection criterion called angle penalized distance (APD) is defined to maintain a balance between convergence and diversity.

In the last few years, several evolutionary algorithms devoted for solving problems with a large number of objectives have been proposed. These algorithms can be divided into three categories based on their selection criteria. In the first one, dominance-based selection is modified to enhance the convergence rate. Some examples of EAs in this category are Borg-MOEA [31] and GrEA [90]. In the second category, EAs use an indicator (usually hypervolume) based selection e.g. IBEA [95] and HypE [6]. These algorithms suffer from the high computational cost of calculating the hypervolume especially with a large number of objectives. In the third category, decomposition based algorithms are used which decompose the objective space into a number of subspaces e.g. using reference vectors. Some of the widely used EAs in this category are MOEA/D [91] and its variants [53, 92] and NSGA-III [22]. RVEA belongs to decomposition based EAs. For more details about these algorithms and challenges in solving problems with more than three objectives, see [35, 52, 88]. In [16], RVEA performed better than other algorithms e.g. NSGA-III and MOEA/D on many benchmark problems and therefore, we adopt some elements from RVEA in using the surrogates to develop an algorithm applicable for expensive MOPs.

Two major differences between RVEA and other decomposition based EAs are the selection criterion and adaptive reference vectors. The selection criterion, APD is designed to dynamically balance convergence and diversity. Moreover, the adaptive set of reference vectors is used to deal with problems having objectives with different scales. Such an adaptation in the reference vectors ensures a uniform distribution of solutions in the objective space. The components of RVEA are presented in Algorithm 1.

In RVEA, first a set of uniformly distributed reference points is generated on a unit hyperplane using the canonical simplex-lattice design method [15, 19]. The corresponding reference vectors are then obtained by projecting the reference points from the hyperplane to a hypersphere. An illustration for a two and a three objective optimization problem is shown in Figure 2, where open circles and filled circles represent reference points on the hyperplane and the hypersphere, respectively. In this way, the reference vectors partition the objective space into a number of subspaces.

Algorithm 1: Reference vector guided evolutionary algorithm

Input: t_{max} = maximum number of generations; N = number of reference vectors; $V_0 = \{v_{01}, v_{02}, \ldots v_{0N}\}$ a set of unit reference vectors

Output: nondominated solutions from population $P_{t_{max}}$

1. Create an initial population P_0 of size N randomly and set generation counter $t = 0$

while $t < t_{max}$ **do**

2. Generate offspring Q_t
3. Combine parent and offspring populations, $P_t = P_t \cup Q_t$
4. Select parents (P_{t+1}) for the next generation
5. Update reference vectors and $t = t + 1$

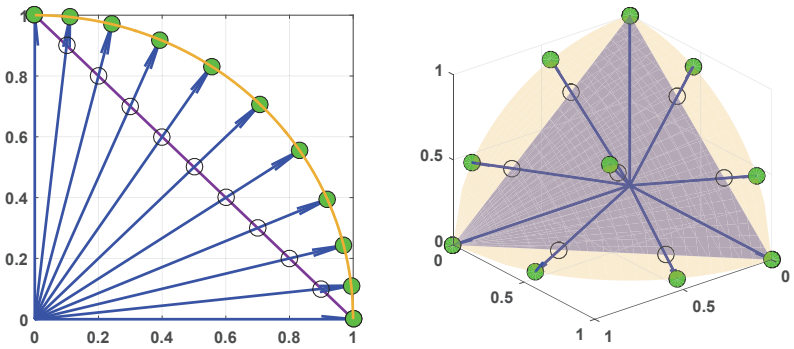


FIGURE 2 An illustrative example of reference vectors for a biobjective and three objective optimization problem

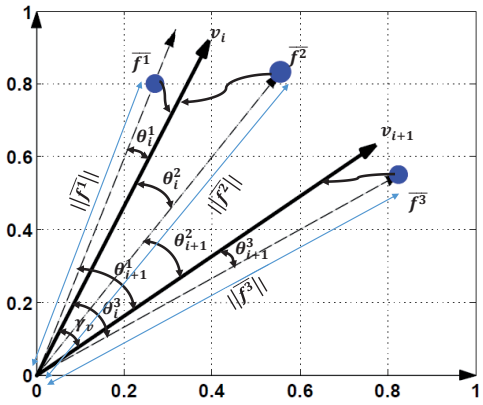


FIGURE 3 An illustration of the assignment of individuals to a reference vectors

To assign individuals to these reference vectors, first all the objective function values are translated i.e. $\bar{f_i^j} = f_i^j - f_i^*$, where f_i^j represents the objective value of f_i for the j^{th} individual and f_i^* the minimum objective values of f_i at the current generation. The translation is done to ensure that all values lie in the positive orthant and the initial point of reference vectors is always the origin. Individuals are then assigned to the reference vectors based on the acute angle measurement between the reference vectors and vectors corresponding to the individuals. One illustration is shown in Figure 3 with two reference vectors v_i and v_{i+1} and three individuals $\bar{f^1}$, $\bar{f^2}$ and $\bar{f^3}$. As the angle θ_i^1 between the individual $\bar{f^1}$ and the reference vector v_i is less than the angle θ_{i+1}^1 between the individual and the other reference vector v_{i+1}, this individual is assigned to the first reference vector v_i. Similarly, $\bar{f^2}$ and $\bar{f^3}$ are assigned to reference vector v_i and v_{i+1}, respectively. In this way, the population is partitioned into different subpopulations.

After the assignment of individuals, one individual is selected from each subpopulation. The selection criterion APD is then used to select individuals:

$$d^j = (1 + P(\theta^j)) \cdot ||\bar{f^j}||, \tag{2}$$

where $||\bar{f^j}||$ is the distance from the translated objective vector corresponding to the j^{th} individual to the origin, and θ^j is the angle between the j^{th} individual and the reference vector it is assigned to. In (2), $P(\theta^j)$ is the penalty function defined as follows:

$$P(\theta^j) = k \cdot (\frac{t}{t_{max}})^{\alpha} \cdot \frac{\theta^j}{\gamma_v}, \tag{3}$$

where k is the number of objectives, α is the parameter controlling the rate of change of the penalty function, t is the generation counter and t_{max} is the maximum number of generations. The smallest angle between the reference vector v_i and its closest neighboring reference vector v_j is defined by γ_v i.e. $\gamma_v = min_{i \in \{1,\ldots,N\}, i \neq j} \langle v_i, v_j \rangle$. The angle γ_v is used to normalize the angles and is important when the distribution of the reference vectors is either too dense or too

sparse. The penalty term is defined to balance between convergence and diversity. For instance, in the early generations, the criterion APD focuses on convergence and later when the generation counter t approaches t_{max}, the diversity is prioritized. The individual with the minimum APD is selected from each subpopulation and used in the next generation.

2.4 Kriging as a surrogate model

Kriging or Gaussian process regression has been one of the most popular choices in surrogate techniques used mainly because of its ability to provide uncertainty information of the approximated values [PI]. The term Kriging was proposed by Matheron in 1963 [58] in the honor of the South African mining engineer Danie G. Krige [50]. His research was focused on the distribution of gold samples found in mines and correlation between these samples. He implemented a statistical technique based on a limited amount of samples which is now known as Kriging.

The first work in using Kriging for approximation of simulation based or computer experiments was proposed in 1989 by Sacks et al. [76]. However, the mostly cited algorithm in using Kriging is efficient global optimization (EGO) proposed in 1998 by Jones et al. [41] for single-objective optimization problems. EGO uses a criterion called expected improvement (EI) to select samples for training the Kriging model. Several versions of EGO have been proposed in the literature afterwards for both single-objective [26, 42, 87] and multiobjective [44, 71, 93] optimization problems. For more details about algorithms using Kriging, see recent reviews [43, 80]. Next, we present the working methodology of Kriging.

Kriging approximates the objective function value of an individual x as

$$y(x) = \mu(x) + \epsilon(x), \tag{4}$$

where $\epsilon(x)$ is a Gaussian stationary process with the zero mean, variance σ^2 and covariance Ψ i.e. $\epsilon(x) = \mathcal{N}(0, \Psi)$. The mean is represented by μ and is usually assumed to be the form $\mu(x) = \sum_{j=1}^{l} \beta_j g_j(x) = g(x)^T\beta$ with l basis functions and coefficients β. In many cases, $\mu(x)$ is just taken as a constant value to avoid estimating the coefficients β.

For training a Kriging model, first a set of input samples is generated in the decision space which are evaluated with the expensive objective function evaluations. Let matrix $X = \left[x^1, \ldots, x^{N_I}\right]^T$ represent the training data in the decision space with their corresponding objective vector $y = \left[y^1, \ldots, y^{N_I}\right]^T$, where N_I represents the sample size, that is the size of the training data set. The covariance between two samples x^i and x^j is calculated as:

$$cov[\epsilon(x^i), \epsilon(x^j)] = \sigma^2 R(x^i, x^j), \tag{5}$$

where R is the correlation function or kernel. The Gaussian kernel i.e. $R(x^i, x^j) = exp(-\sum_{j=1}^{n} \theta_j |x_j^i - x_j^j|^2)$ is the most commonly used, where n is the number of

decision variables and θ represent the hyperparameters. Such a correlation is calculated for all input samples and a correlation matrix $\mathbf{R}$ is generated:

$$\mathbf{R} = \begin{bmatrix} R(x^1, x^1) & \cdots & R(x^1, x^{N_I}) \\ \vdots & \ddots & \vdots \\ R(x^{N_I}, x^1) & \cdots & R(x^{N_I}, x^{N_I}) \end{bmatrix}. \tag{6}$$

The covariance matrix Ψ is then calculated as $\Psi = \sigma^2 \mathbf{R}$.

For a new input $\hat{x}$, an approximated value $\hat{y}$ from (4) can be written as

$$\hat{y}(\hat{x}) = g^T(\hat{x})\beta + r^T(\hat{x})\mathbf{R}^{-1}(y - \mathbf{F}\beta), \tag{7}$$

where $\mathbf{F}$ is the matrix representation of the vectors $g(x^1), \ldots, g(x^{N_I})$ and $r(\hat{x})$ is the correlation vector of size N_I between the new input $\hat{x}$ and the training data $\left[x^1, \ldots, x^{N_I}\right]$ i.e.

$$r(\hat{x}) = \left[R(\hat{x}, x^1), \ldots, R(\hat{x}, x^{N_I})\right]^T. \tag{8}$$

To get an approximated value from formula (7), we need to estimate the hyperparameters β, θ and σ^2. Equation (7) has the generalized least square solution:

$$\beta = (\mathbf{F}^T\mathbf{R}^{-1}\mathbf{F})^{-1}\mathbf{F}^T\mathbf{R}^{-1}y \tag{9}$$

and the estimated variance σ^2 is given by:

$$\sigma^2 = \frac{1}{N_I}(y - \mathbf{F}\beta)^T\mathbf{R}^{-1}(y - \mathbf{F}\beta). \tag{10}$$

Values of θ can be obtained by maximizing the following likelihood function:

$$\psi(\theta) = -\frac{N_I}{2}(ln\ \sigma^2 + ln\ 2\pi) - \frac{1}{2}ln\ det(\mathbf{R}) - \frac{1}{2\sigma^2}(y - \mathbf{F}\beta)^T\mathbf{R}^{-1}(y - \mathbf{F}\beta), \tag{11}$$

where $det(\mathbf{R})$ is the determinant of the correlation matrix $\mathbf{R}$.

The uncertainty estimate or estimated mean is then calculated as

$$\hat{s}^2(\hat{x}) = \sigma^2 \left[1 - \left(g(\hat{x})^T, r(\hat{x})^T\right) \begin{pmatrix} 0 & \mathbf{F}^T \\ \mathbf{F} & \mathbf{R} \end{pmatrix} \begin{pmatrix} g(\hat{x}) \\ r(\hat{x}) \end{pmatrix}\right]. \tag{12}$$

Although, Kriging has been a popular surrogate technique, it has one major limitation of its computational complexity. The computational complexity of the Kriging model is $O(N_I^3)$ [33]. Moreover, calculating hyperparameters by maximizing the likelihood function using an optimization algorithm can further increase the computation time and many algorithms in the literature using Kriging ignore this issue. The uncertainty measure from the Kriging models is the most important building block in a Kriging based algorithm. Sampling based on the uncertainty information not only helps in improving the performance of the surrogates but also helps in searching for unexplored regions [38]. In the literature, Bayesian optimization is also used as the term for sequentially selecting samples

by optimizing an acquisition function [80] involving Kriging models. As evolutionary algorithms are also used with other surrogate techniques in the literature, we use the term surrogate-assisted evolutionary algorithm when solving MOPs with expensive evaluations. In the next chapter, we present a survey of different SAEAs proposed in the literature to deal with expensive multiobjective optimization problems.

3 SURROGATE-ASSISTED EVOLUTIONARY ALGORITHMS

In the literature, several SAEAs have been proposed and selecting one of them to solve a given expensive MOP is not trivial. One needs to know their characteristics, advantages and limitations before they are applied to solve the problem. Therefore, a survey of existing SAEAs is essential and in [PI], we cover 45 different algorithms in 2008-2016 published in English in different journals and conference proceedings. We first present a general framework of using surrogates in Figure 4 and most SAEAs proposed in the literature follow this framework.

In the general framework in Figure 4, initial samples are generated e.g. with some design of experiment (DOE) technique which are evaluated with expensive objective functions. The evaluated samples are then used to build surrogates to approximate the objective function values. An evolutionary algorithm is then used with these surrogate models to find samples (in the decision space) for updating the surrogates. The samples are selected based on some infill criterion defined in the previous chapter. The selected samples are evaluated with the expensive objective functions and combined with the initial samples for updating the surrogates. This loop continues until a termination criterion for instance, the maximum number of expensive function evaluations is reached. Before using the framework, one needs to address some challenges which are described in the next section.

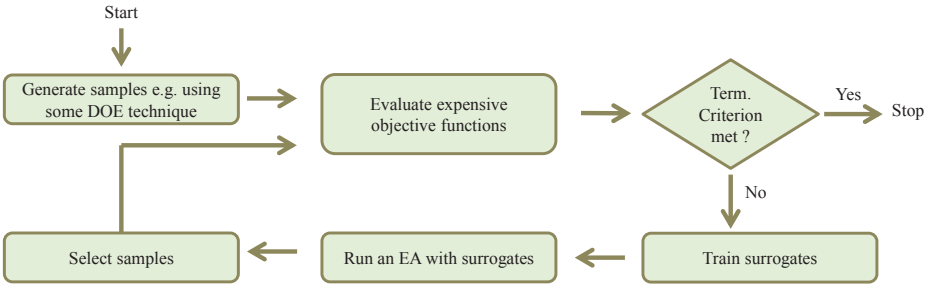


FIGURE 4 A general framework of a surrogate-assisted evolutionary algorithm

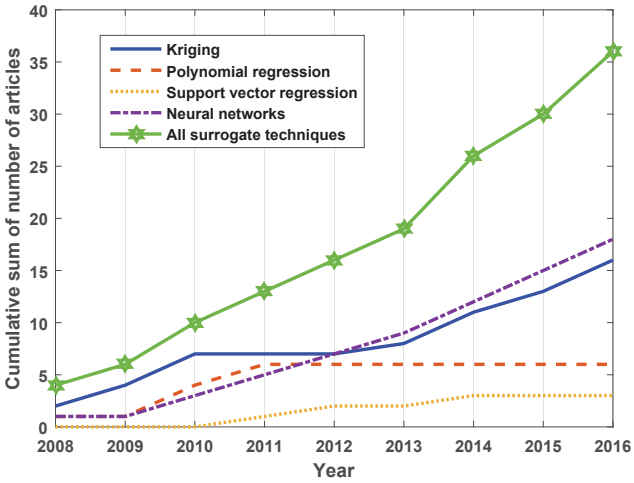


FIGURE 5 Usage of surrogate techniques in 2008-2016 for multiobjective optimization problems

3.1 Challenges in surrogate-assisted evolutionary algorithms

Before developing and applying a SAEA to expensive MOPs, the following questions need to be answered. Some of these questions can also be considered as challenges as it is not straightforward to address them.

1. Which surrogate technique is to be used?
2. How to generate the initial samples for training the surrogates?
3. How many samples are to be generated?
4. Which evolutionary algorithm is to be used?
5. How to select samples for re-training the surrogates?
6. How many samples are to be selected for re-training?
7. What termination criterion is to be used?

There is no clear answer to the first question and not much guidance has been provided in the literature for selecting a particular surrogate technique. Unfortunately, many authors do not always even justify their choice of surrogate technique. Most of the times, a technique is selected either based on its popularity or its usage in the particular application domain. For instance, in [49], radial basis functions are used because the authors found it to be a successful technique in the coastal aquifer management problems. In Figure 5, we present a cumulative sum of the number of times a particular surrogate technique is used in articles published in 2008-2016. As can be seen, Kriging and neural networks have been the most widely used techniques compared to polynomial and support vector regression.

For generating the initial samples for training, usually some design of experiment technique e.g. Latin hypercube sampling is used. Such techniques ensure

a uniform distribution of samples in the decision space which is vital for training the surrogates. The initial sample size is also very important but many algorithms use a size from existing articles without any proper justification. For instance, a size of $11n - 1$ has been used several times in the literature [PII, 41, 44, 71, 93]. The parameter value was first suggested in 1998 when introducing the efficient global optimization by Jones et al. in [41] for expensive single-objective optimization problems. It was clearly stated in the article that based on the past experience of the authors, $10n$ samples are needed for training the surrogate initially and to have a convenient, finite decimal value for the spacing between initial samples, they deviated slightly from the $10n$ rule and made it $11n - 1$. Knowles at al. [44] in 2006 used the same value in ParEGO because of the suggestion for [41]. In 2008, it was used again for SMS-EGO in [71] because of ParEGO and in 2010 for MOEA/D-EGO in [93]. The initial sample size should be related to the budget e.g. the maximum number of function evaluations. For instance, if the budget is less than $11n - 1$, training of surrogates with this number of samples is not possible.

Another challenge in using the surrogates is the training time. It may happen that the time needed for training the surrogates is longer than evaluating an objective function and the whole aim of reducing the computation time is jeopardized. The issue for training time becomes more prominent in case of problems with a large number of decision variables. A large number of samples is needed to train the surrogates for high dimensional problems which can substantially increase the training time. The issue of training time is usually ignored in the literature and few algorithms [3, 4, PII] consider it by fixing the number of samples for training the surrogates.

The next question to be answered is the choice of the evolutionary algorithm (EA). As different EAs perform differently, they can lead to different samples for re-training the surrogate. Therefore, the choice of an EA to be used is very important for the performance of the surrogates. We present the number of articles using different types of EAs for years between 2009 and 2016 years in Figure 6. As can be seen, most of the algorithms have used dominance based EAs and few of them indicator or decomposition based EAs.

The fifth question is also very important in any surrogate based algorithm. An infill criterion is used to select the samples for re-training. In the literature, different types of infill criteria have been used. The usual approaches are to select a set of uniformly distributed samples in the objective space [3, 63, 66, 78, PII], a set of isolated samples in the decision space [1, 56, 65, 68], use expected improvement [93], lower confidence bound [71] or expected hypervolume improvement [5, 83, 70]. The selected samples are then evaluated with expensive objective functions for updating the surrogates.

The fourth and fifth questions are also related to the efficiency of a SAEA for problem with a large number of objectives. The samples using an infill criterion should be selected by taking into account of both convergence and diversity. To select such samples, a surrogate needs to be integrated with the evolutionary algorithm used. As mentioned, most existing SAEAs are dominance based and

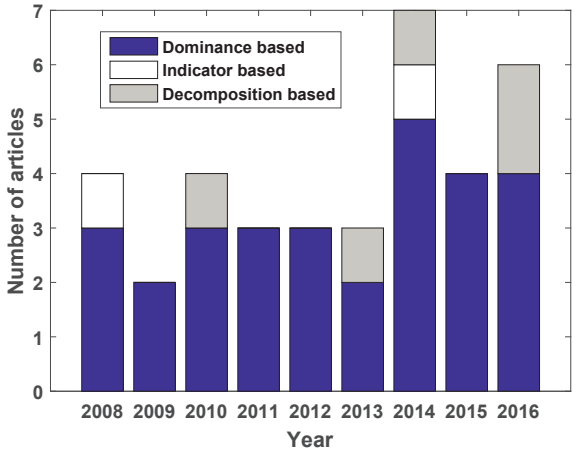


FIGURE 6 Number of articles with respect to the type of evolutionary algorithm used in using surrogates for multiobjective optimization problems

therefore are not well suited for handling a large number of objectives.

The next question is how many samples are to be selected for updating the surrogates. The answer to this question is or at least should be related to the budget available. In the literature, different algorithms use different numbers without any proper justification. For instance, if simulations can be done in parallel, one can afford several evaluations simultaneously. However, this parameter is usually prefixed before the solution process and varies in the range of 1-50 in the literature.

The last concern is to use a justifiable termination criterion. In the literature, most SAEAs use a prefixed number of evaluations. It may happen that after a certain number of evaluations, there is no improvement in the quality of solutions and running more expensive evaluations could be of no use. The maximum number of function evaluations in the literature has varied from 50 to 30000.

3.2 Challenges related to the characteristics of the problem

In addition to the challenges based on the framework, some challenges related to the nature and the characteristics of the problem need to be addressed before applying a SAEA. They are as follows:

1. handling a large number of objectives and decision variables,
2. handling constraints,
3. handling mixed-integer variables and
4. formulating a MOP in real-world cases.

Many real-world problems involve a large number of objectives and decision variables. Most SAEAs in the literature have been developed and tested with

up to three objective functions. Few algorithms [69, 71, PII] have been tested on more than four objectives. One of the main reasons restraining the use of existing SAEAs for a large number of objectives is the lack of an efficient surrogate management strategy. For the number of decision variables, SAEAs with Kriging have been used for problems with a small number of variables compared to other surrogate techniques e.g. neural networks and two algorithms in [54, 67] have been used for problems with more than 30 variables.

Additionally, the scarce consideration of problems having constraints is also one of the important observations from the literature. Some algorithms [29, 55, 82] have been proposed with a strategy to handle constraints in using surrogates. Another challenge is related to the types of variables. In the literature, most SAEAs have been proposed to solve an expensive MOP with continuous variables. This is due to the fact that most of the conventional surrogate techniques cannot be used to deal with integer and categorical variables.

The next challenge is the usage of algorithms on real-world problems with expensive evaluations. Most SAEAs in the literature have been tested on benchmark problems which do not involve any expensive evaluations. On the other hand, the algorithms which have been tested on real-world problems do not mention the expensive nature of the problem e.g. the computation time of a function evaluation. Most real-world problems solved in the literature with SAEAs are simplified versions of industrial problems and are not of a black-box nature.

One more challenge which many algorithms do not consider is the formulation of the optimization problem. In real-world problems, the formulation usually takes several iterations or discussions between the decision maker or an expert in the application domain and an analyst in the optimization algorithm and the verification of the formulation is of high value. In academic problems, such an effort is not needed as the formulation is known before starting the solution process.

3.3 Guidelines for selecting a surrogate-assisted evolutionary algorithm

In the literature, different SAEAs use different evolutionary algorithms, surrogate techniques and other relevant parameters. Therefore, it is difficult to generalize the efficiency of an algorithm or choose an algorithm. However, one can consider the following points to select an algorithm:

1. budget e.g. maximum number of function evaluations available and
2. dimensions in both objective and decision spaces.

The first major point in selecting an algorithm should be the capability of the algorithm to obtain solutions in a given limited budget. In many real-world problems, the number of function evaluations is limited e.g. because of a time limit and it may be infeasible to do initial experimental runs to select an algorithm.

Some algorithms in the literature use up to 50000 function evaluations to solve benchmark problems and such a high number may not be realistic in solving real problems.

Another point to select an algorithm is dimension of the problem to be solved both in objective and decision spaces. As discussed, many surrogate-assisted algorithms have not been applied to problems with a large number of objectives and variables.

In addition to these guidelines, one can look at other elements before selecting a SAEA. For instance, using fitness inheritance [85] and approximating other elements instead of objective functions e.g. approximating rank [7, 57, 78] and distance to the already evaluated samples [68, 69] can also be considered in developing and applying a SAEA. In some real-world problems, it is also possible to simplify the problem e.g. replacing Navier-Stokes equations with Euler equations in doing CFD simulations [30, 51]. Moreover, enhancing the quality of approximations with local search [29, 54] is also possible. In the next chapter, we introduce the K-RVEA algorithm and focus on how to select samples for updating the surrogates, how to reduce the training time, and in particular how to handle problems with a large number of objectives.

4 A KRIGING-ASSISTED REFERENCE VECTOR GUIDED EVOLUTIONARY ALGORITHM

Problems with a large number of objectives and involving expensive functions evaluations have not received much attention in the evolutionary community as tackling the challenge of handling a large number of objectives in using surrogates is not straightforward. One of the main difficulties in addressing this challenge is the appropriate selection of samples to train and improve the accuracy of the surrogates and also the performance of the evolutionary algorithm (EA) used. Additionally, the incorporation of elements of the EA used in managing the surrogates is essential for a better performance. The algorithm K-RVEA starts the process of filling the gap between two fields focusing on the large number of objectives and expensive function evaluations.

One of the important elements in K-RVEA proposed in [PII] is the integration of the surrogates and the EA used. In the literature, most SAEAs manage surrogates without incorporating the elements of EA used. In such a way, the characteristics or the benefits of the EA used cannot be utilized properly. Therefore, in K-RVEA we have used the incorporation of reference vectors and angle penalized distance (APD) from RVEA in managing the surrogates. Additionally, one cannot ignore the training time for surrogates especially when dealing with problems with the expensive function evaluations. Therefore, we have used the reference vectors from RVEA to select appropriate samples for training the surrogates. We have used Kriging models to alleviate the computational cost of expensive objective functions because of their capability to provide uncertainty information of the approximated values. Such uncertainty information is also used with the reference vectors and the APD in managing the surrogates to improve the performance of the algorithm.

K-RVEA uses two sets of reference vectors, adaptive and fixed for managing the surrogates. These reference vectors are used in selecting solutions either based on APD or uncertainty information from the Kriging models. Another feature of K-RVEA is to efficiently manage the size of samples for training. A prefixed number of samples is selected using the reference vectors to train the surrogates for further reducing the computation time. The algorithm consists of

three phases as presented in Algorithm 2: 1) initialization, 2) using RVEA with the surrogates and 3) updating the surrogates. In the algorithm, two archives $A1$ and $A2$ are used for storing the samples for training the surrogates and for storing all the evaluated samples, respectively. In what follows, the notation $|A|$ is used to represent the size of the set A. We also want to recall that the algorithm K-RVEA is developed to deal with only box constrained problems.

Algorithm 2: K-RVEA

Input: FE^{max}, maximum number of expensive function evaluations
Output: nondominated solutions of all evaluated ones in $A2$
Initialization
1. Create an initial population P generated with some design of experiment technique
2. Initialize the number of function evaluations FE = 0 and two empty archives $A1 = A2 = \phi$
3. Evaluate the population P with the original expensive functions and add them to $A1$ and $A2$, update $FE = FE + |P|$
while $FE \leq FE^{max}$ **do**
  4. Train surrogates for each objective function by using individuals in $A1$
  Using RVEA with the surrogates
  5. Run RVEA with Kriging models to find the individuals to update the surrogates
  Updating the surrogates
  6. Select individuals from the previous step using a selection strategy and denote the set by I
  7. Re-evaluate I with the original expensive functions and update $FE = FE + |I|$, update $A1 = A1 \cup I$ and $A2 = A2 \cup I$
  8. Remove extra individuals from $A1$ using management of training data set
  9. Go to step 4

The initialization phase is similar to conventional SAEAs, where initial samples are generated e.g. using a Latin hypercube sampling. The number of samples in this phase is kept to $11n - 1$ for a fair comparison with other existing SAEAs. However, one is allowed to change this parameter based on the maximum number of function evaluations available. These samples are then evaluated with expensive objective functions and added to the archives $A1$ and $A2$. Samples in $A1$ are then used for training the surrogates which are further used with RVEA to approximate the objective function values. In other words, the original expensive objectives have been replaced by the surrogates and the resulting problem is solved with RVEA. The termination criterion in using RVEA with surrogates e.g. maximum number of generations or function evaluations is also relevant. For instance, if after a certain number of generations there is no improvement in convergence or diversity in using surrogates, the second phase i.e. using RVEA with

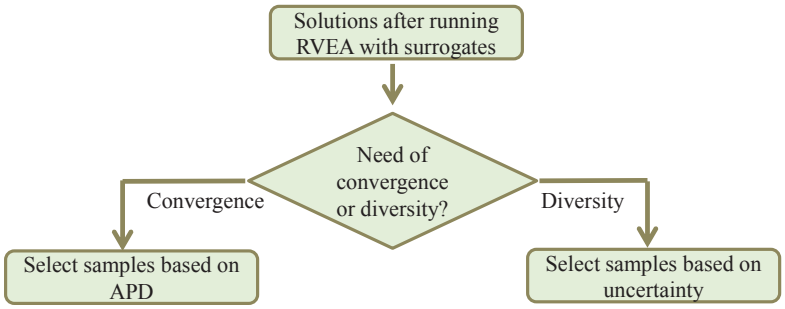


FIGURE 7 An illustration of the selection strategy for updating the surrogates in K-RVEA

the surrogates should be terminated. In [PII], a sensitivity analysis is provided by using the different number of generations as the termination criterion on the performance of the algorithm.

After using the surrogates with RVEA, the samples for updating the surrogates are selected considering both convergence and diversity. As mentioned in [38], the selection of samples with high uncertainty can help in finding unexplored regions and improving the performance of the surrogates. Therefore, samples with high uncertainty are selected whenever diversity is needed. If a satisfactory degree of diversity has already been achieved, samples are selected with the minimum APD. Next, we present the selection strategy for updating the surrogates in K-RVEA.

4.1 Balance of convergence and diversity in K-RVEA

The selection strategy is presented in Figure 7 and uses the reference vectors and the uncertainty information from the Kriging models. To measure the needs of convergence and diversity, a fixed set of reference vectors V_f is used in addition to adaptive ones V_a. First, the solutions obtained using RVEA with surrogates are assigned to V_f and the number of reference vectors in V_f is measured. If the change in the number of empty reference vectors compared to the previous update is smaller than a predefined parameter δ, convergence is prioritized. Otherwise, diversity is used in selecting the samples. In updating the surrogates for the first time, we use convergence as the criterion for selecting the samples. Once the needs of convergence and diversity are checked, the solutions from the latest generation are assigned to the adaptive set of reference vectors V_a and the non-empty reference vectors in V_a are identified. These non-empty vectors are then clustered into a prefixed number of clusters and one sample from each cluster is selected either based on the minimum APD value or the maximum uncertainty. The selected samples I are then evaluated with expensive objective functions and added to the archives $A1$ and $A2$. An advantage of such a selection strategy is that

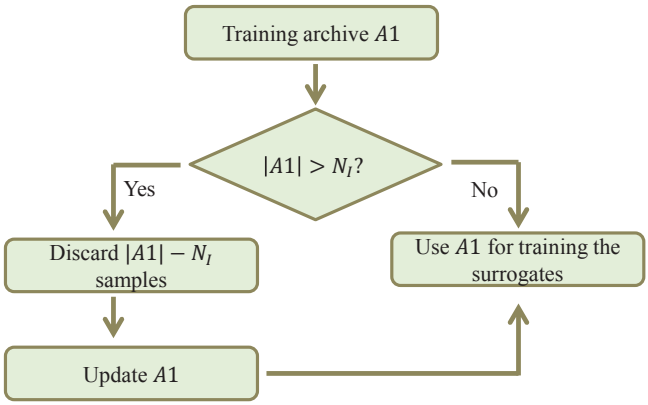


FIGURE 8 An illustration of managing the number of samples in the training archive $A1$

samples can be evaluated in parallel. In the literature, some SAEAs e.g. ParEGO [44] and SMS-EGO [71] select only one sample each time the surrogates are updated. Such an approach may reduce their applicability to problems, where the resources are available for parallel evaluations. One example is provided in [45], where the ParEGO algorithm could not be applied because it did not have the option to do parallel evaluations.

4.2 Managing the size of training samples

Next in K-RVEA, the focus is on managing the size of samples for training the surrogates. Training time of surrogates can be substantial if a large number of samples is used. Therefore, to further reduce the computation time, the prefixed maximum sample size N_I is used for training and the other samples are discarded from the archive $A1$ as also shown in Figure 8. For this, the selected samples I are assigned to the adaptive set of reference vectors V_a. The remaining solutions in $A1$ i.e. $A1 \setminus I$ are then assigned to the non-empty reference vectors in V_a which are then clustered into $N_I - |I|$ number of clusters. One sample from each cluster is selected randomly and combined with I and these samples are then used for training the surrogates. In this way, a prefixed number of diverse samples is maintained in $A1$ to improve the performance of Kriging models. In doing experiments with K-RVEA, we kept $N_I = 11n - 1$ which is also the initial sample size for training the surrogates. One can also use all the evaluated samples in $A1$ for training without discarding any sample if the training time is not significant. The algorithm is terminated after a prefixed maximum number of function evaluations. Next, we discuss the performance of K-RVEA on benchmark problems and on a free-radical polymerization problem.

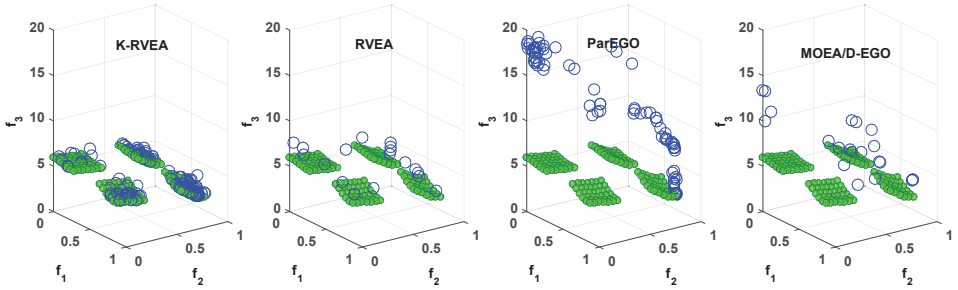


FIGURE 9 Solutions for the DTLZ7 problem with three objectives from K-RVEA, RVEA, ParEGO and MOEA/D-EGO in 300 function evaluations, where the filled circles represent the Pareto front

4.3 Performance of K-RVEA

The K-RVEA algorithm was tested on the DTLZ [23] and the WFG [34] benchmark suites with 3-10 objectives in [PII]. The potential of the algorithm was also shown by comparing it with three representative Kriging based SAEAs, ParEGO, SMS-EGO and MOEA/D-EGO in the same number of function evaluations using inverted generational distance (IGD) and hypervolume. To show the potential of using surrogates, the algorithm was also compared with its underlying algorithm RVEA. K-RVEA performed clearly better than RVEA in most of the problems. One example of results from different algorithms on a three objective DTLZ7 problem is presented in Figure 9. The problem has a disconnected Pareto front and as can be seen, the solutions of K-RVEA are much closer to the Pareto front compared to the solutions from other algorithms. In case of the DTLZ2 problem with 10 objectives in Figure 10, K-RVEA was able to obtain wider ranges of objective functions compared to RVEA. The results in [PII] on various benchmark problems showed the potential of K-RVEA for 3-10 objectives.

The training time of surrogates in K-RVEA was the lowest because of a prefixed number of samples used for training the surrogates. In ParEGO, the number of samples was also kept fixed which was the reason that the training time was comparable to K-RVEA. However, ParEGO randomly rejects the samples in the training archive and K-RVEA uses reference vectors to efficiently manage the samples for training. The number of variables was set to 10 in all the problems because of the applicability of Kriging models for a limited number of variables.

In addition to the results, the importance of performance indicators especially for problems with a large number of objectives was emphasized in [PII]. For example, the IGD metric needs a reference set to calculate the performance and the size of the set can be relevant in measuring the performance. In [PII], the size was chosen according to the number of objectives, in other words, a bigger size was used for a larger number of objectives compared to a smaller number of objectives.

A free-radical polymerization problem [64] with three objectives and four

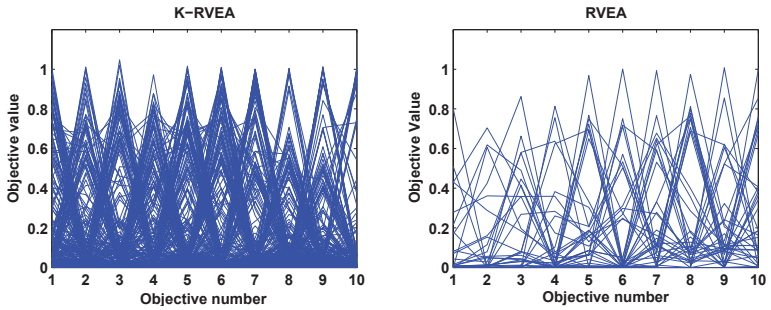


FIGURE 10 Solutions for the DTLZ2 problem with 10 objectives from K-RVEA and RVEA in 300 function evaluations

variables was also solved in [PII]. The average computation time for one function evaluation was 45 minutes which was mainly because of the time in solving the governed differential equations. To compare different algorithms, nondominated solutions obtained from each algorithm were combined and used as the reference set in calculating the IGD metric. In this problem, The K-RVEA algorithm in this problem did perform better than the other algorithms when the same number of function evaluations was available.

Although, K-RVEA can tackle the issue of a large number of objectives, it still lacks means to handle a large number of variables which is one of the future research directions. Additionally, some other relevant parameters were fixed to a constant value and a sensitivity analysis was provided. Adapting these parameters based on the performance of the algorithm does also deserve further future work. In the next chapter, we solve a real-world problem of an air intake ventilation system in an automobile industry by applying K-RVEA.

5 SHAPE OPTIMIZATION OF AN AIR INTAKE VENTILATION SYSTEM

This chapter presents an application of K-RVEA for an air intake ventilation system in an automobile industry considered in [PIII]. The air intake ventilation system in a tractor is used to maintain a uniform temperature in the cabin and deforest the windscreen. The particular component is shown in Figure 12 and consists of four outlets. A good balance in the flow rates from all the outlets with low pressure losses are the two objectives needed to be achieved for maintaining a uniform temperature distribution.

The problem to be solved is expensive because of time consuming computational fluid dynamics (CFD) simulations. The average wall clock time for one function evaluation on a computer with 32 GB RAM and Intel Xeon CPU E5-1607 processor is 3 to 5 minutes. Thanks to K-RVEA and the surrogates involved, solutions can be obtained in few expensive function evaluations. In addition to the expensive nature of the problem, two other important challenges which are also typical in solving real-world problems need to be addressed. One is the formulation of the multiobjective optimization problem and the second is combining different simulation tools to obtain objective function values to be connected to the optimization algorithm.

In benchmark problems, the formulation of optimization problems is given. On the other hand, the formulation of a real-world optimization problem is not necessarily straightforward and may need several discussions and iterations between the DM and an analyst who knows optimization algorithms. In [PIII], it took three iterations to finalize the problem formulation. Therefore in solving real-world problems, one must make sure that an appropriate formulation is derived which properly reflects the needs of the DM and the objective function values are understandable.

The next challenge in solving real-word problems which does not appear in solving academic problems is combining different simulation tools. The problem considered in [PIII] needed two commercial software, ANSYS ICEM [2] for meshing the component according to the values of decision variables and ANSYS CFX [86] for running CFD simulations to obtain objective function values. These two

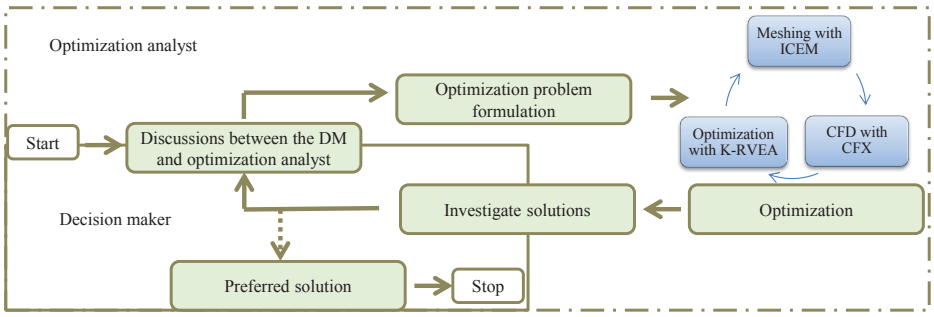


FIGURE 11 An illustration of the problem formulation and selecting a preferred solution by the DM

pieces of software are combined with K-RVEA to find solutions to be shown to the DM. An illustration of the big picture of the problem formulation and selecting a preferred solution by the DM is presented in Figure 11. In [PIII], MATLAB and python scripts are used to combine different pieces of the simulation tools and K-RVEA. After solving the problem, a preferable solution among 40 alternatives was selected by the DM based on his preferences.

5.1 Problem formulation

As mentioned, the particular component of interest consists of four outlets and the diameters of all the outlets play an important role in maintaining a uniform flow rate. Before starting the solution process, an initial design used in the ventilation system was provided by the DM. In using K-RVEA, the scaling factors of the diameters of the initial design provided by the DM are used as the decision variables i.e.

$$x_i = \frac{D_i}{D_i^{(initial)}} \quad \text{for} \quad i = 1, \ldots, 4, \tag{13}$$

where $D_i^{initial}$ is the diameter of the i^{th} outlet in the initial design. The following lower and upper bounds are used for the decision variables during the optimization:

$$\begin{aligned} x_i^{lb} &= 0.5 \quad \text{for} \quad i = 1, \ldots, 4 \\ x_i^{ub} &= 1.5 \quad \text{for} \quad i = 1, \ldots, 4, \end{aligned} \tag{14}$$

where x_i^{lb} and x_i^{ub} are the lower and upper bounds of the decision variables.

To increase the efficiency of the ventilation system, the flow rates from all the outlets should be the same. Additionally, the average pressure drop i.e. the difference between the pressure at the inlet and at the outlets should be as low as possible to achieve better flow rates. Therefore, minimizing the difference between flow rates from different outlets and minimizing the pressure drop were considered as two objectives. Moreover, the diameter of outlet four is the smallest among all the outlets. Therefore, special attention had to be paid towards this

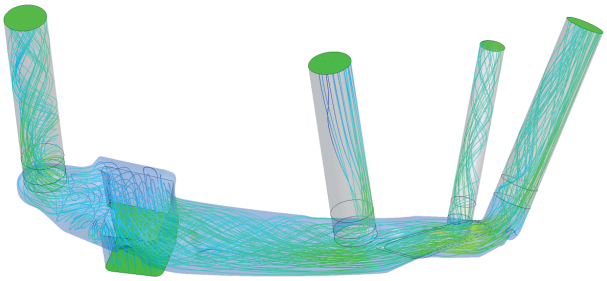


FIGURE 12 CFD simulation results of the initial design provided by the DM

outlet and a third objective was formulated. A difference of average flow rates from outlets one to three and the flow rate from outlet four was minimized to maintain a balance in the flow rates. The objective functions finally formulated are as follows:

f_1: Minimize variance between flow rates at outlets 1 to 3
f_2: Minimize pressure loss
f_3: Minimize the difference between the flow rate at outlet 4 and the average of the flow rates at outlets 1 to 3

5.2 Results with K-RVEA

In [PIII], the K-RVEA algorithm was applied to solve the given problem in 200 function evaluations in one a single run. In Figure 13, 40 nondominated solutions obtained with K-RVEA are presented with the solution corresponding to the initial design. The objective function values were normalized to maintain the confidentiality of the data. These solutions were then shown to the DM who selected the final solution based on his preferences which is also shown in Figure 13. We observed that out of 40 solutions, only one solution has the variable values as upper or lower bounds and for almost all the solutions, the box constraints were not active.

The diameters of the outlets in the final design selected by the DM were $[1.43, 0.92, 0.73, 1.38] \times D_i^{initial}$ for $i = 1, \ldots, 4$. Additionally, in the final design, a significant improvement was achieved in the first and the third objective with a similar value in the second objective compared to the initial design. In [PIII], K-RVEA was also compared with its underlying algorithm RVEA to show the potential of using surrogates. The IGD values with the number of function evaluations with K-RVEA and RVEA are shown in Figure 14 and as can be seen, K-RVEA performed better than RVEA in the same number of function evaluations. So far, we have applied K-RVEA for box-constrained problems. In the next chapter, we show the influence of infeasible solutions in using K-RVEA on constrained problems.

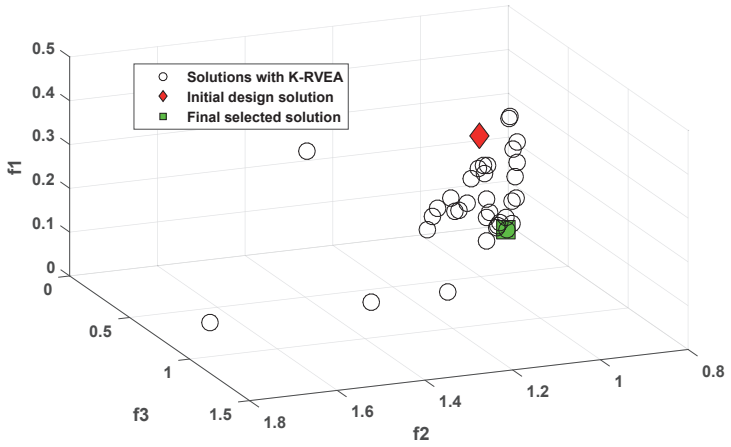


FIGURE 13 Nondominated solutions in the objective space on a normalized scale

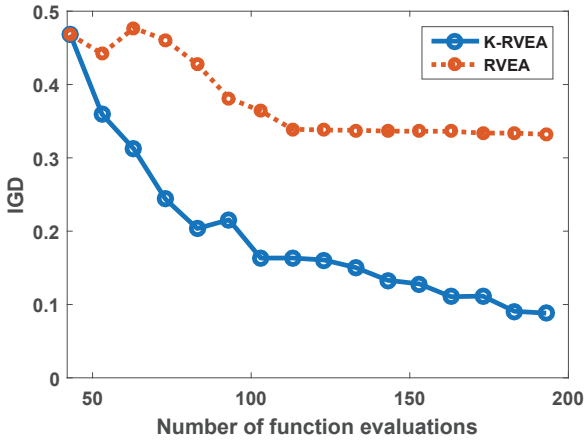


FIGURE 14 IGD values with the number of function evaluations with K-RVEA and RVEA

6 CONSTRAINT HANDLING IN SURROGATE-ASSISTED EVOLUTIONARY ALGORITHMS

In the literature, many SAEAs have been proposed to handle only box constrained problems. Therefore, one cannot easily apply these algorithm to real-world constrained problems. One of the major challenges in handling constraints is to properly deal with the infeasible solutions when training the surrogates. Therefore, in [PIV], we studied the effect of infeasible solutions on the performance of surrogates and consequently on the performance of the optimization algorithm. We used K-RVEA to train the surrogates with different approaches of handling infeasible solutions in the presence of constraints. In the study, we assumed that constraint functions are computationally inexpensive and trained the surrogates for only objective functions.

In Figure 15, we present a flowchart of the constrained version of K-RVEA which consists of two phases. In the first phase, samples for training Kriging models are generated based on the feasibility of solutions. In the second phase, K-RVEA is used with modifications in managing the Kriging models for instance change in the selection criterion in using RVEA with the Kriging models, selection of samples for updating the Kriging models and maintaining the training archive $A1$ based on the constraint violations.

6.1 Handling infeasible training data

In [PIV], three different ways of training the surrogates are tested with K-RVEA. In the first one, only feasible solutions are used to train the surrogates. This approach is useful especially when the feasible region is very small as in C1-DTLZ1 [37]. In Figure 16, we show the solutions generated with the Latin hypercube sampling (LHS), Pareto front and the feasible region for a biobjective C1-DTLZ1. As can be seen, solutions generated with the LHS are very far from the Pareto

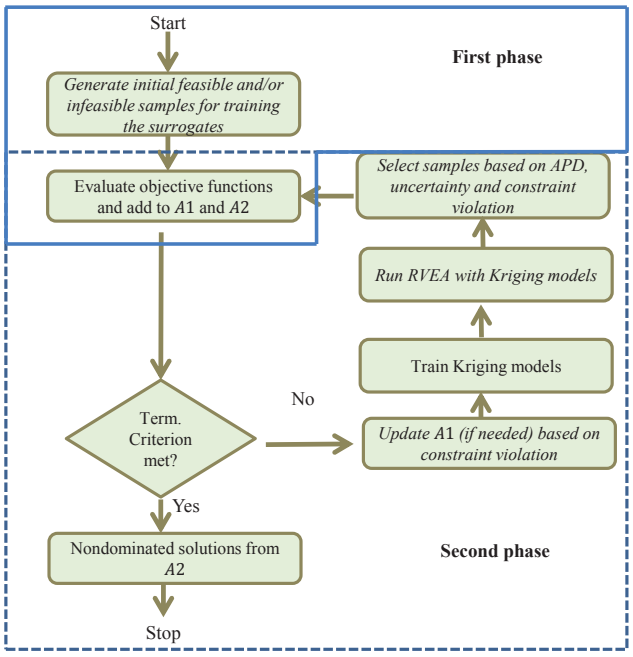


FIGURE 15 A flowchart representing the steps in the constrained version of RVEA

front and all of them are infeasible. Therefore in such problems, it is important to find the feasible solutions first and then use surrogates for approximations.

In the second approach, some infeasible solutions are used in addition to feasible ones to train the surrogates. It may be possible that in the first approach, all feasible solutions are close to each other and therefore, surrogates are not trained with well distributed solutions. Therefore, combining feasible and infeasible solutions for training the surrogates may be helpful to increase their performance. However, the number of infeasible solutions and how close they should be from the feasible region are two relevant parameters.

The third approach is based on assigning a penalty to infeasible solutions whenever they are encountered. However, assigning a penalty parameter is not straightforward and in [PIV], three approaches are adopted based on an extensive study in [61]. In the first approach, a constant penalty parameter is added to penalize the infeasible solutions. In the second approach, the value of the penalty parameter is adapted based on the number of feasible solutions after every time the surrogates are updated. In the third approach, a parameter free approach adopted from [21] is used. Samples generated using any of these three approaches are evaluated with expensive objective functions and added to the archives $A1$ and $A2$ and this completes the first phase.

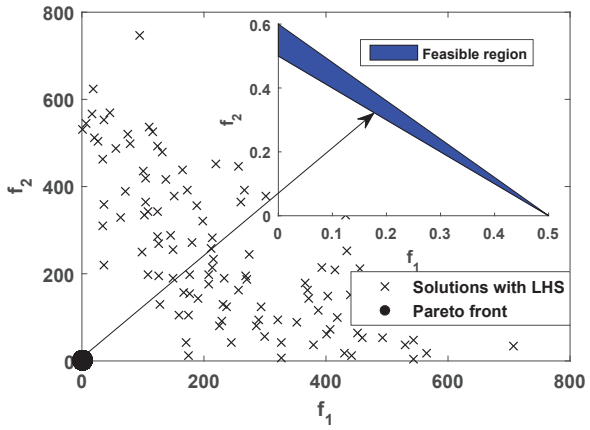


FIGURE 16 Solutions generated with the Latin hypercube sampling (LHS), Pareto front and the feasible region in the biobjective C1-DTLZ1 problem

6.2 Management of surrogates

Samples evaluated in the first phase are used to train Kriging models which are used with RVEA to approximate the objective function values. The selection criterion in this phase is based on APD and the constraint violation. One solution from each subpopulation is selected with a minimum APD if there is at least one feasible solution. Otherwise, the solution with a minimum constraint violation is selected for the next generation.

After having used RVEA for a prefixed number of generations, samples are selected to update the surrogates using the information from reference vectors, uncertainty information from the Kriging models and constraint violations. In each subpopulation, the solution with the minimum APD or the maximum uncertainty is selected if there is at least one feasible solution. Otherwise, the solution with the minimum constraint violation is selected for updating the surrogates. This selection strategy ensures that infeasible solutions are always dominated by the feasible ones. Additionally, selection using either APD or uncertainty is based on the needs of convergence and diversity as described in K-RVEA. Selected samples are then evaluated with expensive objective functions and added to the archives $A1$ and $A2$. To maintain a prefixed size of $A1$, some solutions are discarded based on the constraint violation. For example, if all the solutions are infeasible, the ones with the minimum constraint violations are kept from each subpopulation. This loop continues for a prefixed maximum number of function evaluations and nondominated feasible solutions from $A2$ are used as the final solutions.

6.3 Discussions on the results with K-RVEA

In [PIV], experiments were performed on the constrained version of the DTLZ problems [37] with three to 10 objectives and one to 10 constraints. In the first and the second approach in the first phase, a prefixed number of feasible and infeasible solutions were generated for training the Kriging models. For this purpose, a genetic algorithm using niche based selection [21] was used to solve a single objective optimization problem with constraint violation as the objective function. Additionally, a sensitivity analysis was performed on the numbers of infeasible solutions and their closeness to the feasible region in the second approach. The number of variables was set to 10 in all the problems and a maximum of 300 functions evaluations was used as the termination criterion.

Overall, the first approach when the surrogates were trained with only feasible solutions performed the best. However in some cases e.g. C3-DTLZ4, the approach when infeasible solutions were used for training also performed equivalently to the first approach. In the C1-DTLZ1 problem, where the feasible region is very small, none of the penalty based approaches was able to find even one feasible solution. This is because initial solutions generated for training the surrogates were very far from the feasible region and penalizing infeasible solutions was not useful. Additionally, the adaptive penalty approach performed the best among all penalty based approaches.

These results indicate the influence of infeasible solutions in using the surrogates. A combined and adaptive approach combining different ways of training the surrogates could be beneficial and is a topic for future research. Moreover, only few constrained benchmark problems with more than three objectives exist in the literature and developing constrained expensive problems to test surrogate-assisted algorithms could also be useful.

7 INCORPORATION OF DECISION MAKER'S PREFERENCES

In many real world problems, the DM is usually interested in finding a small set of PO solutions or a single solution based on her/his preferences. Incorporation of preference information can also be helpful in reducing the number of expensive function evaluations by focusing the search and finding a small set of PO solutions desirable to the DM. In the last few years, some evolutionary algorithms [32, 72, 81] have been proposed to incorporate the DM's preferences in the solution process. There are typically three ways to utilize preferences in solving MOPs: a priori, a posteriori and interactive [59], where the preferences from the DM are utilized before, after and iteratively during the solution process, respectively. For more details about preference incorporation in EAs in solving MOPs, see a recent review [72].

In interactive approaches, the DM sees some solutions and expresses preferences and iteratively directs the solution process to find solutions desirable to her/him. Interactive approaches have been found to be suitable for solving real-world problems [60, 62] as they provide possibilities to the DM to learn relationship among the objectives. Interactive approaches have a long history in the non-evolutionary multiobjective optimization field [11, 59, 84]. However, utilizing the DM's preferences in the solution process is not straightforward and different types of preferences can be utlilized [75]. Examples of them are reference points [14], pairwise comparison [24], ranges of the objective function values [32] and selecting a preferred solution from a small set [74].

Most EAs incorporating preference information from the DM do not deal with the non-preferable or undesirable solutions provided by the DM. Such information about non-preferable solutions can also be helpful for guiding the search process in an EA to find a small set of interesting PO solutions. Furthermore, it has been noticed in practice that sometimes it is easier for the DM to tell which solutions are not interesting instead of indicating the most promising ones. Therefore, in [PV], we have proposed an interactive simple indicator-based evolutionary algorithm (I-SIBEA) in which the DM can indicate not only preferable but also non-preferable solutions.

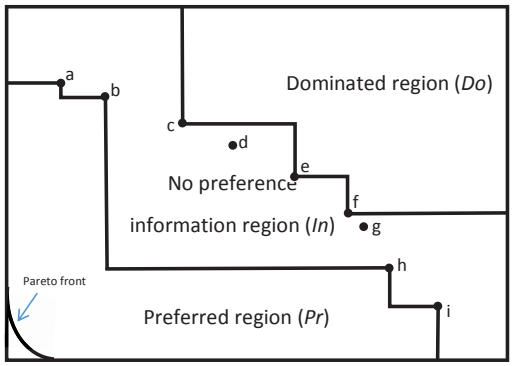


FIGURE 17 Dominated (Do), no preference information (In) and preferred regions (Pr) based on the DM's preferences

7.1 Interactive simple indicator-based evolutionary algorithm

The I-SIBEA algorithm is based on the the SIBEA algorithm [94], where the solutions are selected based on their contribution to the hypervolume. The hypervolume indicator is the only indicator known so far with the Pareto compliance [47] property and has been frequently used in the literature [25, 48, 97] as the selection criterion in evolutionary algorithms. Therefore, we have SIBEA as the underlying algorithm to utilize the advantages of using hypervolume based selection in guiding the solutions based on the DM's preferences. In I-SIBEA, in addition to the hypervolume, the weighted hypervolume is used based on the DM's preferences. In other words, solutions are selected based on the weighted hypervolume after every iteration with the DM. The weighted hypervolume makes sure that the selected solutions represent the DM's preferences.

The I-SIBEA algorithm works as follows. Before the first iteration with the DM, the SIBEA algorithm is used for a prefixed number of generations to find solutions to be shown to the DM. Among the solutions obtained, a prefixed number of solutions is shown to the DM. In I-SIBEA, we provide the option to the DM to give the number of solutions to be shown to her/him. A clustering technique e.g. K-means clustering [36] can be used if the number of solutions obtained with SIBEA is higher than the DM expects. The DM is then asked to provide her/his preferences by selecting preferred and/or non-preferred solutions. One illustration of an interaction with the DM is shown in Figure 17, where a set of non-dominated solutions $A = \{a, b, \ldots, h, i\}$ is shown to DM. The DM then identifies preferable solutions $AA = \{a, b, h, i\}$ and non-preferable solutions $RA = \{c, e, f\}$. Rest of the solutions are considered as solutions with no preference information $IA = \{d, g\}$.

Based on the preferences from the DM, three regions called dominated Do, preferred Pr and no-preference information region In are defined in the objective space. For instance, after the first iteration, the solutions dominated by the non-preferred solutions lie in the dominating region, the solutions dominate the pre-

ferred solutions lie in the preferred region and the remaining solutions lie in the non-preference information region. The following weight distribution function is then used to calculate the weighted hypervolume for the selection of solutions in each generation before the second iteration with the DM:

$$w(z) = \begin{cases} 0 & \text{for all } z \in Do \\ 1 & \text{for all } z \in In \\ 1 + \frac{\mu(Do)}{\mu(Pr)} & \text{for all } z \in Pr \end{cases} \tag{15}$$

where $z = (z_1, \ldots, z_k)$ is the objective vector and $\mu(Do)$ and $\mu(Pr)$ are the hypervolumes of the dominated and preferred regions, respectively. The (weighted) hypervolume is calculated by using Monte-Carlo approach [8]. In the second iteration, the set of preferable, non-preferable and no-preference information solutions are updated based on the DM's preferences. The number of iterations can also be chosen by the DM in the beginning or the solution process can be terminated if the DM identifies the most preferable solution and does not want to continue. In the final iteration, the DM is asked to provide the most preferred solution which is then projected to the Pareto front by optimizing an achievement scalarzing function (ASF) [89] to ensure that the final solution is at least locally Pareto optimal:

$$\text{minimize} \max_{i=1,\ldots,k} [w_i(f_i(x) - f_i^*)] + \rho \sum_{i=1}^{k} w_i(f_i(x) - f_i^*), \tag{16}$$

where f_i^* is the final preferred solution by the DM and ρ is an augmentation coefficient that takes a small value e.g. 10^{-6}. The weight vector $w_i = \frac{1}{z_i^{max} - z_i^{min}}$ is used to assign each objective.

7.2 Discussion on results

In [PV], the I-SIBEA algorithm was tested on the DTLZ1, DTLZ2 and ZDT4 benchmark problems with 2-3 objectives and 7-11 decision variables. To compare with another interactive evolutionary algorithm W-Hype [10], the DM was replaced by a weighted Chebyshev function [9] $\max_{i=1,\ldots,k} [w_i(f_i(x) - z_i^*)]$ at each interactive solution process, where z^* as the ideal objective vector. The weight vector $(w_1, \ldots, w_k)^T$ is assigned to each objective function and used to simulate different DM's preferences.

After every iteration, the solution that minimized the weighted Chebyshev function was considered as the preferable solution and the rest were considered as non-preferable solutions. In a given number of iterations with the DM, I-SIBEA obtained an equivalent or better value of the Chebyshev function in few function evaluations. In [PV], the algorithm was also tested on a biobjective ZDT4 problem with a real DM. In providing the preferences, the DM was deliberately inconsistent to see how the algorithm follows the preferences. The algorithm was flexible

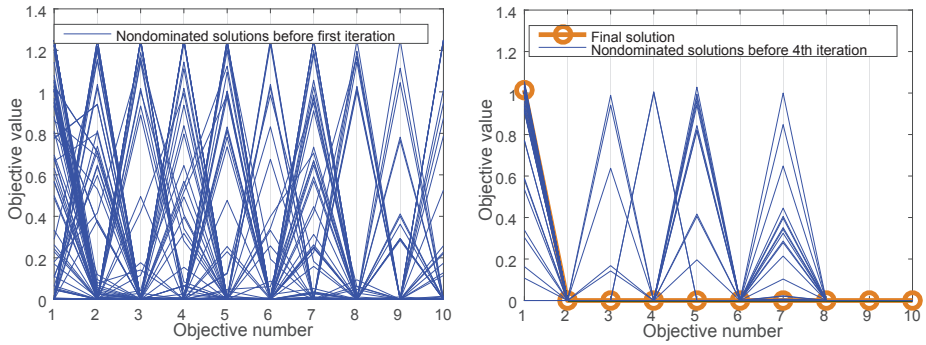


FIGURE 18 Nondominated solutions before first and fourth iteration and the final solution obtained on 10 objective DTLZ4

to the DM's preferences and found a satisfactory solution to him after four iterations.

Evolutionary algorithms using hypervolume based selection are suitable to deal with a large number of objectives. Therefore in this chapter, we apply I-SIBEA on the DTLZ4 problem with 10 objectives. We use the weighted Chebyshev function to replace the DM in the solution process and use four iterations. In Figure 18, we present the nondominated solutions before the first and the fourth iteration and the final solution obtained. The following weight vector in the weighted Chebyshev function:
$w_i = [0, 0.1, 0, 0, 0, 0.1, 0, 0.3, 0.5, 0.1]$.

As can be seen in the figure, before the first iteration, a uniform set of solutions is obtained in the objective space. After three iterations, nondominated solutions converged to one part of the objective space based on the DM's preferences. For instance, the values of objectives number two, six and 8-10 are zero before the fourth iteration because of the weights assigned to these objective in the weighted Chebyshev function. Finally, in the last iteration, the final solution is obtained by projecting the most preferable solution to the Pareto front by solving an ASF. One of the limitations in the current approach is the computation time in calculating the (weighted) hypervolume especially with a large number of objectives and dealing with the high computation cost of calculating (weighted) hypervolume will be our future work.

7.3 Preference incorporation in K-RVEA

As mentioned at the beginning of this chapter, finding a small set of PO solutions based on the DM's preferences can reduce the number of expensive function evaluations and the number can further be substantially reduced if surrogates are used. Therefore, the K-RVEA algorithm is used here where the DM is assumed to provide her/his preferences as a priori i.e. before starting the solutions

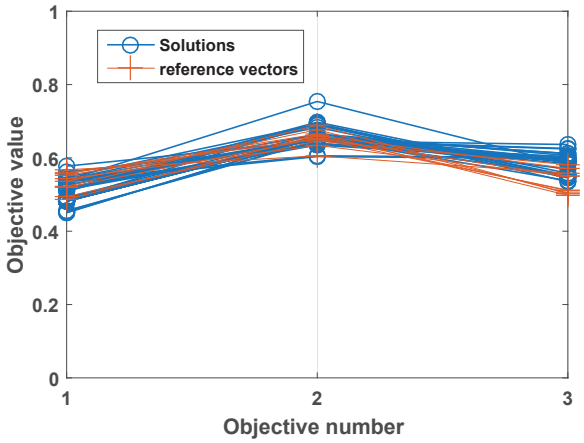


FIGURE 19 Solutions obtained with K-RVEA and the reference vectors on the three objective DTLZ2 problem with preferred ranges $0.4 \leq f_1 \leq 0.5, 0.5 \leq f_2 \leq 0.6$ and $0.4 \leq f_3 \leq 0.5$ in 300 function evaluations

process. We give the possibility to the DM by specifying her/his preferences in three ways which are adopted from [32]: 1) by specifying a reference point, 2) by providing desirable ranges of the objective function values and 3) by providing non-preferred solutions. We want to remind the reader that in K-RVEA a uniform set of reference vectors is generated in the objective space which can be adjusted accordingly based on the DM's preferences. For instance, if the DM provides a reference point z, the reference vectors $V = \{v^i \in R^k | i = 1, \ldots, N\}$ are adjusted as:

$$\bar{v}^i = \frac{r \cdot v^i + (1-r) \cdot v^c}{\|r \cdot v^i + (1-r) \cdot v^c\|}, \tag{17}$$

where $v^c = \frac{z}{\|z\|}$ is considered as the central vector and $r \in (0,1)$ is the radius which is prefixed and determines how far the reference vectors are from the central vector v^c.

Similarly, if the DM provides a set of non-preferable solutions, the reference vectors can be generated by considering each non-preferable solution as the reference point from (17). The part of the objective space corresponding to these reference vectors is then identified and these vectors are eliminated. Another set of reference vectors can be generated in other parts of the objective space which is then used in the algorithm. If the DM provides the preferences in terms of ranges of the objective functions $\left[f_i^l, f_i^u\right]$ for $i = 1, \ldots, k$, a k-dimensional hyperbox is created and a set of points is generated within the hyperbox which are then projected to the unit hypersphere to be used as the reference vectors. This new set of reference vectors is then used within K-RVEA to obtain solutions preferable to the DM. We show an example on the DTLZ2 problem with 3 objectives in Figure 19 and 10 objectives in Figure 20, where the DM is assumed to provide the preferences in terms of ranges of the objective function values and the reference

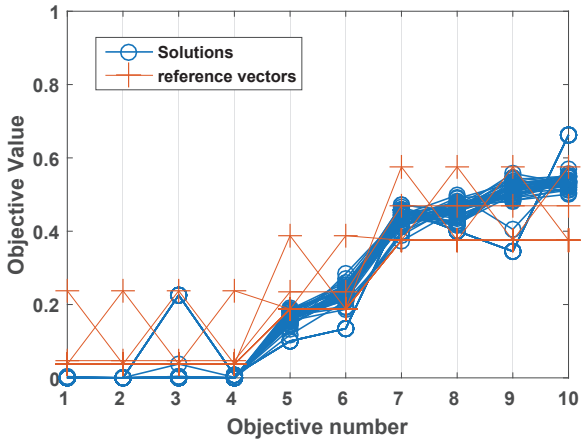


FIGURE 20 Solutions obtained with K-RVEA and the reference vectors on the 10 objective DTLZ2 problem with reference point $[0.1, 0.1, 0.1, 0.1, 0.5, 0.5, 1, 1, 1, 1]$ in 300 function evaluations

point, respectively. In both cases, a maximum of 300 function evaluations and 10 reference vectors are used.

In the three objective DTLZ2 problem, the reference points in the three dimensional hyperbox are generated using the Latin hypercube sampling which are then projected to the unit hypersphere. As can be seen in Figure 19, K-RVEA is able to obtain solutions for the corresponding reference vectors. In the 10 objective case, (17) is used to adjust the reference vectors with the radius $r = 0.15$. For the 10 objective case as well, K-RVEA is able to obtain solutions corresponding to the DM's preferences. Based on these results, we can say that K-RVEA has the potential to find solutions corresponding to the DM's preferences.

The I-SIBEA and K-RVEA algorithms deal with the DM's preferences in different ways. I-SIBEA by using hypervolume as the selection criterion provides the option to the DM for identifying preferable and/or non-preferable solutions. On the other hand, K-RVEA includes surrogate and uses the reference vectors to obtain solutions based on the preferences. In this chapter, we have not incorporated the preferences iteratively during the solution process in K-RVEA and developing an interactive version by considering other types of preferences and using some elements e.g. hypervolume from I-SIBEA will be our future work.

8 AUTHOR'S CONTRIBUTION

The research topic of solving expensive MOPs with evolutionary algorithms was suggested by the author's supervisors. In the beginning, the author performed a literature survey of existing surrogate based evolutionary algorithms proposed to deal with expensive MOPs and found their advantages, promising elements and limitations. After feedback and comments from the supervisors, a survey was written in [PI].

The first idea of developing K-RVEA in [PII] started in 2015 when Prof. Yaochu Jin joined the Industrial Optimization Group at the University of Jyvaskyla as a Finland Distinguished Professor (FiDiPro) in the project called Decision Support for Complex Multiobjective Optimization Problems (DeCoMo). The initial plan was to develop an algorithm for two to three objective optimization problems. The author developed and proposed an efficient methodology and tested the resulting algorithm on benchmark problems. Prof. Kaisa Miettinen suggested to increase the efficiency of the algorithm for a large number of objectives. Therefore, the author introduced some more elements after useful discussions with his supervisors to make the algorithm adaptable for more than three objectives. In addition, the author compared K-RVEA with other state-of-the-art surrogate-based algorithms and put a substantial effort in running different algorithms, which were implemented in different programming languages.

The air intake ventilation system problem in [PIII] was provided by one of automobile industries involved in the DeCoMo project. The author combined K-RVEA with other simulation tools and solved the problem with the help of Dr. Karthik Sindhya and Tomas Kratky from Center of Hydraulic Research Czech Republic. The author analyzed the results and the resulting article was submitted and accepted for publication in the IEEE Congress on Evolutionary Computation (IEEE CEC) 2017.

The constrained version of K-RVEA in [PIV] was also motivated by one of the industries involved in the project. The optimization problem to be solved had constraints and K-RVEA had not been developed to deal with infeasible solutions. Therefore, a study was performed by the author on K-RVEA to see the effect of infeasible solutions. The resulting article after some discussions and

comments was then submitted and accepted in the conference Parallel Problem Solving from Nature (PPSN) 2016.

An initial idea of developing an interactive evolutionary algorithm for multiobjective optimization problems using the weighted hypervolume was first discussed in the Dagstuhl seminars in 2009 and 2012. Later on, the author refined, implemented and tested it on different MOPs which led to the algorithm I-SIBEA in [PV]. With the feedback and comments from his supervisors, an article was submitted and accepted in Evolutionary Multi-Criterion Optimization (EMO) 2015 conference. In [PV], the algorithm was tested on a maximum of three objective optimization problems. In this thesis, the author showed the potential of the algorithm on a 10 objective optimization problem in Chapter 7. Additionally, the author worked with Dr. Jussi Hakanen to develop a version of K-RVEA which employs preference information a priori in Chapter 7.

Overall, working in the DeCoMo was a useful opportunity for the author to apply and implement his primary ideas. Feedback and comments from the co-authors helped him to refine the ideas for publishing the articles in different journals and conference proceedings.

9 CONCLUSIONS

The ultimate motivation in this thesis is to solve real-world multiobjective optimization problems (MOPs) and develop algorithms that can tackle various challenges involved. We have adopted evolutionary algorithms (EAs) to solve MOPs because of their certain advantages mentioned in Chapter 1. However, one cannot ignore the fact that EAs consume a lot of function evaluations to obtain an approximated set of Pareto optimal (PO) solutions. This issue becomes even more severe when dealing with problems with expensive function evaluations. In this thesis, we have recognized the importance of adapting EAs for solving such problems by using surrogates. In the first part of the thesis, we have collected several articles proposed in the field of surrogate-assisted evolutionary algorithms (SAEAs) in [PI]. Additionally, we have raised the key challenges in applying and developing a SAEA for a MOP with expensive function evaluations. Some of them are how to select samples for training the surrogates and how to deal with the training time. Moreover, we have observed that most SAEAs cannot be simply applied for problems with a large number of objectives. Therefore, we have developed the K-RVEA algorithm to deal with problems with three and more number of objectives.

The K-RVEA algorithm by using the elements from its underlying evolutionary algorithm RVEA efficiently manage the surrogates. Additionally, Kriging models because of their capability to provide uncertainty information are used to approximate the expensive objective functions. The algorithm consists of three phases: initialization, using RVEA with the surrogates and updating the surrogates. All three phases play an important role in the entireness, e.g. the initialization phase is important for the accuracy of the surrogates in the beginning, using RVEA with the surrogates ensures the selection of appropriate samples for the next phase and updating the surrogates ensures the performance of the algorithm in successive iterations. We have used the reference vectors, the angle penalized distance and the uncertainty information from the Kriging models for selecting samples to update the surrogates considering both convergence and diversity. One more important issue which cannot be ignored in solving problems with expensive function evaluations is the training time of the surrogates. In K-

RVEA, we have selected a predefined maximum number of samples for training by using the reference vectors for further reducing the computation time.

In [PIII], we have solved a multiobjective shape optimization of air intake ventilation system. The problem involves time consuming computational fluid dynamics simulations and K-RVEA was able to obtain solutions in few function evaluations. In other words, the practical validity of the algorithm was demonstrated. A solution was finally selected by the DM who had the substance knowledge of the application domain based on his preferences. In addition to expensive evaluations, in [PIII], we have focused on other two important challenges, which are usually faced by practitioners in the industry. The first one is the formulation of the optimization problem and the second one is connecting the different pieces of simulation tools to obtain objective function values. The formulation of the optimization problem considering the needs of the DM is relevant and in the literature, the effort of formulation is not often visible as most algorithms have been applied to benchmark problems, where the formulation of the problem is already given. Also, for doing optimization, practitioners typically use different software or simulation tools and combining these pieces of simulation tools with the optimization algorithm needs an effort. These two challenges reflect and provide a message of the importance of an appropriate formulation of the optimization problem and finally solving it by connecting different pieces of simulation tools.

The next part in the thesis was to provide an insight to the practitioners to use an appropriate set of samples based on their feasibility for solving constrained problems. We have shown the influence of infeasible solutions in using K-RVEA in [PIV] by using different approaches to deal with the constraints. We have used only feasible solutions, a mixture of feasible and infeasible solutions and three penalty based methods to handle constraints while training the surrogates. We have found out that infeasible solutions can play a vital role on the performance of surrogates and consequently, on the performance of the algorithm.

Finally, in [PV], we have considered the preferences of the DM in the solution process and have shown that not only preferred but also non-preferred solutions can be used in guiding the search in an evolutionary algorithm. We have developed an interactive simple indicator-based evolutionary algorithm (I-SIBEA) which uses (weighted) hypervolume based selection to incorporate the DM's preferences. Eliminating the need of finding the whole set of PO solutions by including the DM's preferences can also be helpful in reducing the number of expensive function evaluations. Moreover, we have also presented the results of using the DM's preferences in K-RVEA and such an approach can also substantially decrease the number of expensive function evaluations to find solutions desirable to the DM.

We have addressed several challenges for solving real-world problems in this thesis. However, there are still other challenges, which are not covered in the thesis and are considered as the topics for future research. One important challenge is to deal with a large number of decision variables with K-RVEA. Handling a large number of decision variables is not straightforward as a large number of samples is typically needed to train the surrogates which can substantially in-

crease the training time. Using other surrogate techniques e.g. sparse Gaussian processes [73] to select appropriate samples for training may be helpful in handling problem with a large number of variables. Another challenge which cannot be ignored is to set the parameter values in using surrogates. Developing a strategy to adapt e.g. the termination criterion, especially when dealing with expensive function evaluations and the size of the samples to be selected for updating the surrogates will be our future work. Dealing with the high computation cost of calculating hypervolume and developing an interactive version of K-RVEA are also included in the topics of our future research.

Furthermore, data-driven optimization problems, where exact formulation or the simulation models for the objective functions are not available and rather some data is available obtained e.g. through some physical experiments are also common in industry. We have also dealt with a such kind of problem in [17], where a small size data from a blast furnace was available. We built eight Kriging models for each objective function after pre-processing of the data and RVEA was used to obtain the nondominated solutions. In data-driven optimization problems, the methodology of using surrogates can be applied with an adaptation in the surrogate management. K-RVEA has potential in being adapted for data-driven optimization problems which broadens its capability. Extending K-RVEA by modifying the surrogate management strategy for data-driven optimization problems will also be future work.

To be summarized, this thesis has addressed and highlighted several key issues and challenges which practitioners usually face in solving industrial optimization problems. They are the formulation of the optimization problem, connecting different pieces of simulation tools with the optimization algorithm, dealing with a large number of objectives, getting solutions in few expensive function evaluations, influence of infeasible solutions in constrained problems and incorporating the DM's preferences in the solution process. The algorithms and other studies from different articles in this thesis will provide an insight and guidelines to the practitioners to tackle different challenges for solving a real world multi-objective optimization problem.

YHTEENVETO (FINNISH SUMMARY)

Laskennallisesti vaativien monitavoiteoptimointitehtävien ratkaiseminen evoluutioalgoritmeilla

Teollisuudessa on usein monitavoiteoptimointitehtäviä, joissa on monia ristiriitaisia tavoitteita. Niiden ratkaiseminen vaikeutuu, jos tavoitteiden arvojen laskeminen on kallista (esim. aikaa vievien simulaatioiden tai kalliiden kokeiden vuoksi). Tässä tutkimuksessa esitetään ensin katsaus kirjallisuudessa esitettyihin monitavoiteoptimoinnin menetelmiin, joilla kallista funktioarvojen laskentaa sisältäviä tehtäviä voidaan ratkoa. Katsauksessa havaittiin että useimmat menetelmistä eivät sovi yli kolmen tavoitteen tehtäville. Siksi tutkimuksessa esitellään uusi Kriging-sijaismallipohjainen evoluutiomenetelmä K-RVEA, joka käyttää referenssivektoreita ja soveltuu tehtäville joissa on vähintään kolme tavoitetta. Menetelmä tasapainoilee dynaamisesti konvergenssin ja diversiteetin välillä käyttäen referenssivektoreita ja Kriging-mallista saatavaa epävarmuustietoa.

Uuden K-RVEA-menetelmän käyttökelpoisuutta havainnollistetaan traktorin ilmanvaihtokanavan suunnittelutehtävällä. Siinä on kolme tavoitetta, joiden arvojen laskeminen edellyttää laskennallisesti kalliita virtausmekaniikan simulointeja. Samalla kuvataan optimointitehtävän muotoilemisen haasteita, jotta päätöksentekijän tarpeet saadaan kuvattua ja erilaiset simulointityökalut kytkettyä yhteen. Lisäksi tutkimuksessa laajennetaan K-RVEA rajoitteita sisältäville tehtäville. Tarkastelussa korostui rajoitteita toteuttamattomien ratkaisujen merkitys menetelmän toiminnalle.

Monissa käytännön monitavoiteoptimoinnin tehtävissä päätöksentekijä on kiinnostunut yhdestä tai pienestä Pareto-optimaalisten ratkaisujen joukosta, jotka noudattavat hänen preferenssejään. Lisäksi on havaittu, että päätöksentekijälle voi olla helpompaa sulkea pois tarkastelusta huonoja kuin valita mieleisiään ratkaisuvaihtoehtoja. Siksi tutkimuksessa esitellään vuorovaikutteinen indikaattoripohjainen evoluutiomenetelmä I-SIBEA, jossa päätöksentekijä ohjaa ratkaisuprosessia valitsemalla huonoja ratkaisuja, joita tulee välttää ja/tai hänelle mieleisiä ratkaisuja. Päätöksentekijän roolin innoittamana lopuksi esitellään K-RVEA -menetelmän versio, joka käyttää päätöksentekijän preferenssitietoa Kriging sijaismallien kanssa. Tämä väitöskirja auttaa käytännön monitavoiteoptimointiongelmien parissa työskenteleviä tarjoamalla heille tehokkaita menetelmiä ja parantaa heidän kykyään ratkaista vaativia tehtäviä.

REFERENCES

[1] T. Akhtar and C. Shoemaker. Multi objective optimization of computationally expensive multi-modal functions with RBF surrogates and multi-rule selection. *Journal of Global Optimization*, 64:17–32, 2015.

[2] ANSYS, Inc. *ANSYS ICEM CFD Tutorial Manual*, 2013.

[3] A. Arias-Montano, C. Coello, and E. Mezura-Montes. Multi-objective airfoil shape optimization using a multiple-surrogate approach. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pages 1–8. IEEE, 2012.

[4] V. Asouti, S. A. Kyriacou, and K. Giannakoglou. *PCA-Enhanced Metamodel-Assisted Evolutionary Algorithms for Aerodynamic Optimization*, pages 47–57. Springer International Publishing, 2016.

[5] N. Azzouz, S. Bechikh, and L. Said. Steady state IBEA assisted by MLP neural networks for expensive multi-objective optimization problems. In C. Igel, editor, *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 581–588. ACM, 2014.

[6] J. Bader and E. Zitzler. HypE: An algorithm for fast hypervolume-based many-objective optimization. *Evolutionary Computation*, 19(1):45–76, 2011.

[7] S. Bandaru, A. Ng, and K. Deb. On the performance of classification algorithms for learning Pareto-dominance relations. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pages 1139–1146. IEEE, 2014.

[8] N. Beume, B. Naujoks, and M. Emmerich. SMS-EMOA: Multiobjective selection based on dominated hypervolume. *European Journal of Operational Research*, 181(3):1653–1669, 2007.

[9] V. Bowman. On the relationship of the Tchebycheff norm and the efficient frontier of multiple-criteria objectives. In H. Thiriez and S. Zionts, editors, *Multiple Criteria Decision Making*, volume 130, pages 76–85. Springer-Verlag, Berlin, Heidelberg, 1976.

[10] D. Brockhoff, J. Bader, L. Thiele, and E. Zitzler. Directed multiobjective optimization based on the weighted hypervolume indicator. *Journal of Multi-Criteria Decision Analysis*, 20:291–317, 2013.

[11] J. Buchanan. Multiple objective mathematical programming: A review. *New Zealand Operational Research*, 14:1–27, 1986.

[12] M. Buhmann. *Radial basis functions: Theory and Implementations*. Cambridge University Press, 2003.

[13] C. Campbell and Y. Ying. *Learning with Support Vector Machines*. Morgan and Claypool Publishers, 2011.

[14] S. Chaudhuri and K. Deb. An interactive evolutionary multi-objective optimization and decision making procedure. *Applied Soft Computing*, 10:496–511, 2010.

[15] R. Cheng, Y. Jin, K. Narukawa, and B. Sendhoff. A multiobjective evolutionary algorithm using gaussian process-based inverse modeling. *IEEE Transactions on Evolutionary Computation*, 19:838–856, 2015.

[16] R. Cheng, Y. Jin, M. Olhofer, and B. Sendhoff. A reference vector guided evolutionary algorithm for many objective optimization. *IEEE Transactions on Evolutionary Computation*, 20:773–791, 2016.

[17] T. Chugh, N. Chakraborti, K. Sindhya, and Y. Jin. A data-driven surrogate-assisted evolutionary algorithm applied to a many-objective blast furnace optimization problem. *Materials and Manufacturing Processes, to appear*, doi: 10.1080/10426914.2016.1269923.

[18] C. Coello, G. Lamont, and D. Veldhuizen. *Evolutionary Algorithms for Solving Multi-objective Problems*. Springer, New York, 2nd edition, 2007.

[19] J. Cornell. *Experiments with Mixtures: Designs, Models, and the Analysis of Mixture Data*. John Wiley & Sons, 2011.

[20] I. Couckuyt, D. Deschrijver, and T. Dhaene. Fast calculation of multiobjective probability of improvement and expected improvement criteria for Pareto optimization. *Journal of Global Optimization*, 60:575–594, 2014.

[21] K. Deb. An efficient constraint handling method for genetic algorithms. *Computer Methods in Applied Mechanics and Engineeringe*, 186:311–338, 2000.

[22] K. Deb and H. Jain. An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: solving problems with box constraints. *IEEE Transactions on Evolutionary Computation*, 18:577–601, 2014.

[23] K. Deb, A. Prarap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6:182–197, 2002.

[24] K. Deb, A. Sinha, P. J. Korhonen, and J. Wallenius. An interactive evolutionary multiobjective optimization method based on progressively approximated value functions. *IEEE Transactions on Evolutionary Computation*, 14:723–739, 2010.

[25] M. Emmerich, N. Beume, and B. Naujoks. An EMO algorithm using the hypervolume measure as selection criterion. In C. Coello, A. Aguirre, and E. Zitzler, editors, *Proceedings of the Evolutionary Multi-Criterion Optimization*, pages 62–76. Springer, Berlin, Heidelberg, 2005.

[26] M. Emmerich, K. Giannakoglou, and B. Naujoks. Single- and multiobjective evolutionary optimization assisted by Gaussian random field metamodels. *IEEE Transactions on Evolutionary Computation*, 10:421–439, 2006.

[27] A. Forrester and A. Keane. Recent advances in surrogate-based optimization. *Progress in Aerospace Sciences*, 45:50–79, 2009.

[28] A. Forrester, A. Sobester, and A. Keane. *Engineering Design via Surrogate Modelling: A practical guide*. Wiley, 2008.

[29] C. A. Georgopoulou and K. Giannakoglou. A multi-objective metamodel-assisted memetic algorithm with strength-based local refinement. *Engineering Optimization*, 41:909–923, 2009.

[30] K. Giannakoglou and I. Kampolis. Multilevel optimization algorithms based on metamodel- and fitness inheritance-asisted evolutionary algorithms. In Y. Tenne and C.-K. Goh, editors, *Computational Intelligence in Expensive Optimization Problems*, pages 61–84. Springer, Berlin, Heidelberg, 2010.

[31] D. Hadka and P. Reed. Evolutionary computation. *Borg: An auto-adaptive many-objective evolutionary computing framework*, 21:231–259, 2013.

[32] J. Hakanen, T. Chugh, K. Sindhya, Y. Jin, and K.Miettinen. Connections of reference vectors and different types of preference information in interactive multiobjective evolutionary algorithms. In *IEEE Symposium Series on Computational Intelligence (IEEE SSCI)*, 2017.

[33] J. Hensman, N. Fusi, and N. Lawrence. Gaussian processes for big data. In *Conference on Uncertainty in Artificial Intelligence*, 2013.

[34] S. Huband, L. Barone, L. While, and P. Hingston. A scalable multi-objective test problem toolkit. In C. Coello, A. H. Aguirre, and E. Zitzler, editors, *Evolutionary Multi-Criterion Optimization*, pages 280–295. Springer, 2005.

[35] H. Ishibuchi, N. Tsukamoto, and Y. Nojima. Evolutionary many-objective optimization: A short review. In *Proceedings of IEEE Congress on Evolutionary Computation*, pages 2419–2426. IEEE, 2008.

[36] A. Jain, M. Murty, and P. Flynn. Data clustering: A review. *ACM Computing Surveys*, 31:264–323, 1999.

[37] H. Jain and K. Deb. An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part II: Handling constraints and extending to an adaptive approach. *IEEE Transactions on Evolutionary Computation*, 18:602–622, 2014.

[38] Y. Jin. A comprehensive survey of fitness approximation in evolutionary computation. *Soft Computing*, 9:3–12, 2005.

[39] Y. Jin. Surrogate-assisted evolutionary computation: Recent advances and future challenges. *Swarm and Evolutionary Computation*, 1:61–70, 2011.

[40] Y. Jin, M. Olhofer, and B. Sendhoff. A framework for evolutionary optimization with approximate fitness functions. *IEEE Transactions on Evolutionary Computation*, 6:481–494, 2002.

[41] D. Jones, M. Schonlau, and W. Welch. Efficient global optimization of expensive black-box functions. *Journal of Global Optimization*, 13:455–492, 1998.

[42] J. Kleijnen and W. van Beers. Application-driven sequential designs for simulation experiments: Kriging metamodelling. *Journal of the Operational Research Society*, 55:876–883, 2004.

[43] J. P. Kleijnen. Regression and kriging metamodels with their experimental designs in simulation: A review. *European Journal of Operational Research*, 256(1):1–16, 2017.

[44] J. Knowles. ParEGO: A hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems. *IEEE Transactions on Evolutionary Computation*, 10:50–66, 2006.

[45] J. Knowles. Closed-loop evolutionary multiobjective optimization. *IEEE Computational Intelligence Magazine*, 4:77–91, 2009.

[46] J. Knowles and H. Nakayama. Meta-modeling in multiobjective optimization. In J. Branke, K. Deb, K. Miettinen, and R. Slowinski, editors, *Multiobjective Optimization: Interactive and Evolutionary Approaches*, pages 245–284. Springer, Berlin, Heidelberg, 2008.

[47] J. Knowles, L. Thiele, and E. Zitzler. A tutorial on the performance assessment of stochastic multiobjective optimizers. Technical report, ETH Zurich, 2006.

[48] J. D. Knowles, D. W. Corne, and M. Fleischer. Bounded archiving using the lebesgue measure. In *Proceedings of IEEE Congress on Evolutionary Computation*, pages 2490–2497. IEEE, 2003.

[49] G. Kourakos and A. Mantoglou. Development of a multi-objective optimization algorithm using surrogate models for coastal aquifer management. *Journal of Hydrology*, 479:13–23, 2013.

[50] D. Krige. A statistical approach to some basic mine valuation problems on the witwatersrand. *Journal of the Chemical, Metallurgical and Mining Engineering Society of South Africa*, 52:119–139, 1951.

[51] V. Lattarulo, P. Seshadri, and G. Parks. Optimization of a supersonic airfoil using the multi-objective alliance algorithm. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 1333–1340. ACM, 2013.

[52] B. Li, J. Li, K. Tang, and X. Yao. Many-objective evolutionary algorithms: A survey. *ACM Computing Surveys*, 48:13–35, 2015.

[53] H. Li and Q. Zhang. Multiobjective optimization problems with complicated Pareto sets, MOEA/D and NSGA-II. *IEEE Transactions on Evolutionary Computation*, 12:284–302, 2009.

[54] D. Lim, Y. Jin, Y.-S. Ong, and B. Sendhoff. Generalizing surrogate-assisted evolutionary computation. *IEEE Transactions on Evolutionary Computation*, 14:329–354, 2010.

[55] G. Liu, X. Han, and C. Jiang. A novel multi-objective optimization method based on an approximation model management technique. *Computer Methods in Applied Mechanics and Engineering*, 197:2719–2731, 2008.

[56] Y. Liu and M. Collette. Improving surrogate-assisted variable fidelity multi-objective optimization using a clustering algorithm. *Applied Soft Computing*, 24:482–493, 2014.

[57] I. Loshchilov, M. Schoenauer, and M. Sebag. A mono surrogate for multiobjective optimization. In G. Raidl, editor, *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 471–478. ACM, 2009.

[58] G. Matheron. Principles of geostatistics. *Economic Geology*, 58:1246–1266, 1963.

[59] K. Miettinen. *Nonlinear Multiobjective Optimization*. Kluwer, Boston, 1999.

[60] K. Miettinen, J. Hakanen, and D. Podkopaev. Interactive nonlinear multiobjective optimization methods. In S. Greco, M. Ehrgott, and J. Figueira, editors, *Multiple Criteria Decision Analysis: State of the Art Surveys*, pages 927–976. Springer New York, 2016.

[61] K. Miettinen, M. M. Makela, and J. Toivanen. Numerical comparison of some penalty-based constraint handling techniques in genetic algorithms. *Journal of Global Optimization*, 27:427–446, 2003.

[62] K. Miettinen, F. Ruiz, and A. Wierzbicki. Introduction to multiobjective optimization: Interactive approaches. In J. Branke, K. Deb, K. Miettinen, and R. Słowiński, editors, *Multiobjective Optimization: Interactive and Evolutionary Approaches*, pages 27–57. Springer Berlin Heidelberg, 2008.

[63] K. Mitra and S. Majumder. Successive approximate model based multi-objective optimization for an industrial straight grate iron ore induration process using evolutionary algorithm. *Chemical Engineering Science*, 66:3471–3481, 2011.

[64] A. Mogilicharla, T. Chugh, S. Majumder, and K. Mitra. Multi-objective optimization of bulk vinyl acetate polymerization with branching. *Materials and Manufacturing Processes*, 29:210–217, 2014.

[65] P. S. Palar, T. Tsuchiya, and G. Parks. Comparison of scalarization functions within a local surrogate assisted multi-objective memetic algorithm framework for expensive problems. In *IEEE Congress on Evolutionary Computation (CEC)*, pages 862–869, 2015.

[66] L. Pavelski, M. Delgado, C. Almeida, R. Gonaslves, and S. Venske. Extreme learning surrogate models in multi-objective optimization based on decomposition. *Neurocomputing*, 180:55–67, 2016.

[67] L. Pavelski, M. Delgado, C. Almeida, R. Goncalves, and S. Venske. ELMOEA/D-DE: Extreme learning surrogate models in multi-objective optimization based on decomposition and differential evolution. In *Proceedings of the Brazilian Conference on Intelligent Systems*, pages 318–323. IEEE, 2014.

[68] M. Pilát and R. Neruda. ASM-MOMA: Multiobjective memetic algorithm with aggregate surrogate model. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pages 1202–1208. IEEE, 2011.

[69] M. Pilát and R. Neruda. Improving many-objective optimizers with aggregate meta-models. In *Proceedings of the 11th International Conference on Hybrid Intelligent Systems*, pages 555–560. IEEE, 2011.

[70] M. Pilát and R. Neruda. Hypervolume-based local search in multi-objective evolutionary optimization. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 637–644. ACM, 2014.

[71] W. Ponweiser, T. Wagner, D. Biermann, and M. Vincze. Multiobjective optimization on a limited budget of evaluations using model-assisted S-metric selection. In G. Rudolph, T. Jansen, S. Lucas, C. Poloni, and N. Beume, editors, *Proceedings of the Parallel Problem Solving from Nature-PPSN X*, pages 784–794. Springer, Berlin, Heidelberg, 2008.

[72] R. Purshouse, K. Deb, M. Mansor, S. Mostaghim, and R. Wang. A review of hybrid evolutionary multiple criteria decision making methods. In *Proceedings of IEEE Congress on Evolutionary Computation*, pages 1147–1154. IEEE, 2014.

[73] J. Quiñonero Candela and C. E. Rasmussen. A unifying view of sparse approximate gaussian process regression. *The Journal of Machine Learning Research*, 6:1939–1959, 2005.

[74] A. Ruiz, M. Luque, K. Miettinen, and R. Saborido. An interactive evolutionary multiobjective optimization method: Interactive WASF-GA. In A. Gaspar-Cunha, A. Henggeler, and C. Coello, editors, *Evolutionary Multi-Criterion Optimization 2015*, pages 249–263. Springer International Publishing, 2015.

[75] F. Ruiz, M. Luque, and K. Miettinen. Improving the computational efficiency in a global formulation (GLIDE) for interactive multiobjective optimization. *Annals of Operations Research*, 197:47–70, 2012.

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

[77] K. Sastry, D. E. Goldberg, and M. Pelikan. Don't evaluate, inherit. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 551–558. Morgan Kaufmann, 2001.

[78] C.-W. Seah, Y.-S. Ong, I. W. Tsang, and S. Jiang. Pareto rank learning in multi-objective evolutionary algorithms. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pages 1–8. IEEE, 2012.

[79] R. Shah, P. Reed, and T. Simpson. Many-objective evolutionary optimisation and visual analytics for product family design. In L. Wang, A. Ng, and K. Deb, editors, *Multi-objective Evolutionary Optimisation for Product Design and Manufacturing*, pages 137–159. Springer, 2011.

[80] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas. Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE*, 104(1):148–175, 2016.

[81] K. Sindhya, A. Ruiz, and K. Miettinen. A preference based interactive evolutionary algorithm for multi-objective optimization: PIE. In H. Takahashi, K. Deb, E. Wanner, and S. Greco, editors, *Proceedings of Evolutionary Multi-Criterion Optimization 2011*, pages 212–225, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[82] H. Singh, T. Ray, and W. Smith. Surrogate assisted simulated annealing (SASA) for constrained multi-objective optimization. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pages 1–8. IEEE, 2010.

[83] P. Singh, I. Couckuyt, F. Ferranti, and T. Dhaene. A constrained multi-objective surrogate-based optimization algorithm. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pages 3080–3087. IEEE, 2014.

[84] T. Stewart. A critical survey on the status of multiple criteria decision making theory and practice. *OMEGA*, 20:569–586, 1992.

[85] A. Syberfeldt, H. Grimm, A. Ng, and R. John. A parallel surrogate-assisted multi-objective evolutionary algorithm for computationally expensive optimization problems. In *Proceedings of the IEEE World Congress on Computational Intelligence*, pages 3177–3184. IEEE, 2008.

[86] N. Trev. *CFX: Computational Fluid Dynamics, Ansys, HVAC*. International Book Market Service Limited, 2012.

[87] W. van Beers and J. Kleijnen. Kriging for interpolation in random simulation. *Journal of the Operational Research Society*, 54:255–262, 2003.

[88] C. von Lücken, B. Barán, and C. Brizuela. A survey on multi-objective evolutionary algorithms for many-objective problems. *Computational Optimization and Applications*, 58:707–756, 2014.

[89] A. Wierzbicki. The use of reference objectives in multiobjective optimization. In G. Fandel and T. Gal, editors, *Proceedings of the Multiple Criteria Decision Making Theory and Application*, pages 468–486. Springer, Berlin, Heidelberg, 1980.

[90] S. Yang, M. Li, X. Liu, and J. Zheng. A grid-based evolutionary algorithm for many-objective optimization. *IEEE Transactions on Evolutionary Computation*, 5:721–736, 2013.

[91] Q. Zhang and H. Li. MOEA/D: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, 11:712–731, 2007.

[92] Q. Zhang, W. Liu, and H. Li. The performance of a new version of MOEA/D on CEC09 unconstrained MOP test instances. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pages 203–208. IEEE, 2009.

[93] Q. Zhang, W. Liu, E. Tsang, and B. Virginas. Expensive multiobjective optimization by MOEA/D with Gaussian process model. *IEEE Transactions on Evolutionary Computation*, 14:456–474, 2010.

[94] E. Zitzler, D. Brockhoff, and L. Thiele. The hypervolume indicator revisited: On the design of pareto-compliant iindicators via weighted integration. In S. Obayashi, K. Deb, C. Poloni, T. Hiroyasu, and T. Murata, editors, *Proceedings of Evolutionary Multi-Criterion Optimization*, pages 862–876. Springer Berlin Heidelberg, 2007.

[95] E. Zitzler and S. Kunzli. Indicator-based selection in multiobjective search. In X. Yao, E. Burke, J. Lozano, J. Smith, J. Merelo-Guervós, J. Bullinaria, J. Rowe, P. Tiňo, and H.-P. S. A. Kabán, editors, *Proceedings of the Parallel Problem Solving from Nature-PPSN VIII*, pages 832–842. Springer, Berlin, Heidelberg, 2004.

[96] E. Zitzler, M. Laumanns, and L. Thiele. SPEA2: Improving the strength Pareto evolutionary algorithm for multiobjective optimization. In K. Giannakoglou, editor, *Proceedings of the Evolutionary Methods for Design, Optimisation and Control with Application to Industrial Problems (EUROGEN 2001)*, pages 95–100. CIMNE, 2002.

[97] E. Zitzler and L. Thiele. Multiobjective optimization using evolutionary algorithms-a comparative case study. In A. Eiben, editor, *Proceedings of the*

Parallel Problem Solving from Nature-PPSN V, pages 292–301. Springer Heidelberg, 1998.

ORIGINAL PAPERS

PI

A SURVEY ON HANDLING COMPUTATIONALLY EXPENSIVE MULTIOBJECTIVE OPTIMIZATION PROBLEMS WITH EVOLUTIONARY ALGORITHMS

by

Tinkle Chugh, Karthik Sindhya, Jussi Hakanen and Kaisa Miettinen

Submitted to a journal

A survey on handling computationally expensive multiobjective optimization problems with evolutionary algorithms

Tinkle Chugh, Karthik Sindhya, Jussi Hakanen and Kaisa Miettinen

University of Jyvaskla, Faculty of Information Technology,
P.O. Box 35 (Agora), FI-40014, University of Jyvaskyla, Finland

Abstract

Evolutionary algorithms are widely used for solving multiobjective optimization problems but are often criticized because of a large number of function evaluations needed. Approximations, especially function approximations also refer as surrogates or metamodels are commonly used in the literature to reduce the computation time. This paper presents a survey of 45 different recent algorithms proposed in the literature between 2008-2016 to handle computationally expensive multiobjective optimization problems. Several algorithms are discussed based on what kind of an approximation such as problem, function or fitness approximation they use. Most emphasis is given to function approximation based algorithms. We also compare these algorithms based on different criteria such as metamodelling technique and evolutionary algorithm used, type and dimensions of the problem solved, handling constraints, training time and the type of evolution control. Furthermore, we identify and discuss some promising elements and major issues among algorithms in the literature related to using an approximation and numerical settings used. In addition, some guidelines are also provided in selecting a particular algorithm based on the characteristic of the problem to be solved, dimensions in both objective and decision spaces and the budget available.
Keywords:surrogate, metamodel, approximations, multicriteria optimization , computational cost, response surface approximation

1 Introduction

Many engineering problems have multiple objectives to be optimized and these objectives are typically conflicting in nature, i.e. improvement in one objective is possible only by allowing deterioration of at least one of the other objectives. These kinds of problems are known as multiobjective optimization problems (MOPs). Because of the conflicting nature, there typically does not exist one optimal solution, but multiple so-called Pareto optimal solutions. The set of all Pareto optimal solutions in the objective space is called a Pareto front. In many problems, explicit formulations of objective or constraint functions are not known and such functions are called black box functions. Usually, problems involving such functions need a long time to be solved e.g. problems involving computational fluid dynamics simulations utilizing finite element algorithms take substantial time to obtain one solution. These are examples of problems that we refer to as computationally expensive multiobjective optimization problems.

In the last few decades, EAs [21, 28] have been widely used for solving MOPs because of their advantages such as obtaining a set of nondominated solutions in one solution process, ability to handle problems with multiple local and nonconvex Pareto fronts, ability to easily deal with different kinds of variables (such as binary, real, integer or mixed) and no assumptions set on convexity and differentiability of objectives and constraints involved. Despite of these advantages, EAs do not guarantee convergence to optimal solutions. Moreover, they are often criticized as they consume many function evaluations which increases the computation time. This concern is particularly relevant when dealing with computationally expensive problems. It is therefore required to adapt EAs in a way that they can be used to obtain solutions in less computation time without too much reduction in the quality of solutions. In this paper, a survey of different algorithms proposed in the literature to handle computationally expensive MOPs based on EAs is presented.

Several algorithms have been proposed reduce the computational cost while solving MOPs using EAs. Different surveys exist in the literature [58, 59, 111, 70] on this topic. One of the popular approaches mentioned in these surveys is the use of approximation, especially, function approximation to handle computationally expensive problems. In [58], different ways of using approximation such as problem, function and fitness approximation were discussed based on their use in the literature. Additionally, various issues such as how evolutionary algorithms can benefit from these approximations and what kind of approximation to be selected etc. were also discussed. It is to be noted that fitness approximation is also used as evolutionary ap-

proximation in the literature [58]. In [111], the main focus was to discuss various algorithms proposed in the literature before the year 2008 based on the classification of [58]. Some real-world applications were also mentioned where different algorithms had been applied to reduce the computation time. In [59], the main focus was to discuss the recent developments for using function approximation in reducing the computation time. In that survey, different issues such as strategies for managing approximation, selection of individuals to be evaluated using approximation functions etc. were discussed. In addition, the author also mentioned the potential of using function approximation in dynamic, robust and constrained optimization problems. A taxonomy of metamodel based optimization algorithms was provided in [47]. In [70], several algorithms based on the management of using function approximation (e.g. Kriging, radial basis function etc.) were discussed. In addition, the importance of interactive algorithms with EAs was detailed. Two real-world applications were also mentioned, which were solved by two different algorithms, ParEGO [68] and the algorithm proposed in [93]. In a recent article [5], an overview of the state-of-the-art in surrogate-assisted multiobjective optimization algorithms was presented. However, the article does not give an exhaustive analysis of the existing algorithms in all types of approximations nor provides the guidelines in selecting a particular algorithm for the given problem to be solved.

In this survey, we extend these surveys and discuss 45 algorithms based on the classification of [58] from the year 2008 to 2016 published in different journals and conference proceedings in English. We also found that most of the algorithms use Kriging when compared to other function approximation techniques and therefore, classify different function approximation based algorithms into Kriging and non-Kriging based algorithms. This survey is also different from other surveys in following ways. 1. Algorithms using function approximation or surrogates are emphasized as they are more widely used. 2. Classification of function approximation based algorithms further into Kriging and non-Kriging based algorithms shows the wide applicability of Kriging. 3. Algorithms are described in reference to a general function approximation framework which will provide the reader an understanding for using any function approximation in EAs. 4. The efficiency of different algorithms in terms of reducing computational cost or number of function evaluations is emphasized. 5. Various shortcomings in algorithms are observed and discussed e.g. handling constraints, dimensions (both in decision and objective spaces) of the problem solved, training time etc. 6. Some promising elements are also identified which can be helpful in overcoming the limitations of several issues e.g. efficient management of function approximation technique to handle more than three objectives. 7. Some guidelines are also provided in selecting a particular algorithm based on the characteristic of the problem to be solved, dimensions in both objective and decision spaces and the budget available.

In the rest of this paper, in Section 2, some relevant concepts are described which are frequently used when reducing the computational burden. In Section 3, different algorithms are discussed based on the steps of general approximation framework. In addition, different algorithms are compared based on the type of approximation, evolutionary algorithm, evolution control and characteristics of the optimization problem solved. A brief discussion on various issues and using promising elements related to the use of different algorithms is presented in Section 4. Finally, conclusions are drawn in Section 5 along with future research directions.

2 Basic concepts and terminology

We consider multiobjective optimization problems of the form [88]:

$$\begin{aligned} &\text{minimize } \{f_1(x), \ldots, f_k(x)\} \\ &\text{subject to } x \in S \end{aligned} \tag{1}$$

with $k(\geq 2)$ objective functions $f_i(x) : S \rightarrow \Re$. The vector of objective function values is denoted by $f(x) = (f_1(x), \ldots, f_k(x))^T$. For the simplicity of presentation, we assume that all the objective functions are to be minimized. If some objective function f_i is to be maximized, it is equivalent to minimize $-f_i$. The (nonempty) feasible region S is a subset of the decision variable space $\Re^n$ and consists of decision variable vectors $x = (x_1, \ldots, x_n)^T$ that satisfy all the constraints. The image of the feasible space S in the objective space $\Re^k$ is called the feasible objective set denoted by Z. The elements of Z are called feasible objective vectors denoted by $f(x)$ or $z = (z_1, \ldots, z_k)^T$, where $z_i = f_i(x), i = 1, \ldots, k$, are the objective function values. As discussed in the introduction, objective functions in a MOP are typically conflicting in nature, and, thus, there is no single well-defined optimal solution but a set of so-called Pareto optimal solutions exist. We say that a vector $z^1 \in \Re^k$ is said to dominate a vector $z^2 \in \Re^k$ and denoted by $z^1 \prec z^2$ if and only if for all $1, \ldots, k$: $f_i^1(x) < f_i^2(x)$.

A decision vector $x^* \in S$ is Pareto optimal if there does not exist another decision vector $x \in S$ such that $f_i(x) \leq f_i(x^*)$ for all i=1,...,k and $f_j(x) < f_j(x^*)$ for at least one index j. An objective vector is Pareto optimal if the corresponding decision vector is Pareto optimal. A Pareto optimal set consists of all Pareto optimal solutions in the decision space and a Pareto front consists of all Pareto optimal solutions in the objective space.

Ideal and nadir objective vectors represent bounds for objective function values in the Pareto front. Components of an ideal objective vector $z^* \in \Re^k$ are deter-

mined by minimizing each objective function individually, that is, $z_i^* = \underset{x \in S}{\text{minimize}} f_i(x)$. A nadir objective vector consists of upper bounds of objective functions in the Pareto front. It is usually difficult to obtain for problems with more than two objectives. Several ways of approximating it have been proposed in the literature (see e.g. [29, 88]). In what follows, we define concepts relevant to the present survey.

A *metamodel* is an approximation of some computationally expensive element in a multiobjective optimization problem. A metamodel replaces the computationally expensive element by an element which consumes less computation time. There can be different computationally expensive elements while solving a MOP. For example, original functions or constraints, hypervolume etc. A surrogate is often used as a synonym for a metamodel and the process to build a metamodel is known as metamodelling or function approximation. Therefore, what to approximate using a metamodel is an important concern in reducing the computation time. Neural networks [89, 134], radial basis functions [104], support vector regression [8] and Kriging [49] are some examples of commonly used metamodelling techniques.

An *ensemble of metamodels* means using more than one metamodel to reduce the computation time and usually there are two ways to use an ensemble of metamodels in the literature. In the first one, one metamodel having the highest accuracy among different metamodels is selected to evaluate individuals [118]. In the second one, different metamodels are used to evaluate individuals with a different weight coefficient (ω_j) assigned to them. For instance, in [79], the predicted fitness value from an ensemble of m metamodels was defined as:

$$F_{ens}(x) = \sum_{j=1}^{m} \omega_j \overline{F^j}(x) \quad \sum_{j=1}^{m} \omega_j = 1 \tag{2}$$

where $\overline{F^j}(x)$ is the approximated fitness value of $F = \sum_{i=1}^{k} w_i f_i(x)$ using j^{th} metamodel and $w_i, i = 1, \ldots, k$ is the weight vector assigned to each objective function. A weighted sum method was used in [79], however, other scalarizing methods such as ϵ-constraint method, achievement scalarizing function [88] can also be used to convert a multiobjective optimization problem into a single objective optimization problem. Fitness values, $F_{ens}(x)$ is approximated by m metamodels and ω_j is the weight assigned to the fitness value of the j^{th} metamodel. The fitness value of a metamodel is assigned a larger weight if it is found to be more accurate than other metamodels and the accuracy can be obtained by statistical measurements such as root mean square error [123] etc.

An *evolution control* or *model management [61]* is a strategy to manage metamodels in an EA. There are two different ways to manage metamodels, fixed and adaptive evolution control. In a fixed evolution control, metamodels are used for a prefixed number of generations and updated afterward with a predefined criterion. On the other hand, in an adaptive evolution control, the frequency of using the metamodel is adjusted according to the accuracy of the metamodel. Therefore, the model management is also concerned with when to update the metamodel.

In *fitness inheritance* [113, 138], the fitness values of offspring are evaluated using fitness values of the parents. In *fitness imitation* [67], individuals are grouped into several clusters e.g. using K-means clustering [54] in the decision space and only those individuals are evaluated which represent the clusters (e.g. individuals closest to the centroid of each cluster). The fitness values of other individuals are estimated using the fitness values of the representative individuals.

Next, we discuss the general function approximation framework and describe various algorithms based on the steps of the framework.

3 Approximation based algorithms

In approximation based algorithms, the computationally expensive element of the problem is replaced with an approximation which consumes less computation time. As classified in [58], an approximation can be applied in three ways in multiobjective optimization problems: problem, function and fitness approximation. In problem approximation, the original problem is replaced with a simplified problem which is faster to solve. In function approximation, an approximation of a computationally expensive function is formed which is faster to evaluate. On the other hand, in fitness approximation, the fitness value referring typically to the function value of an individual is derived from the fitness values of the existing evaluated individuals in its vicinity. The function approximation is more widely used than other approximations and algorithms applying it are discussed next in Section 3.1. Approaches using problem and fitness approximation are discussed in Section 3.2.

3.1 Function approximation

As said, function approximation is the most commonly used approach among approximation based algorithms and there, an explicit or an implicit approximation of a computationally expensive, function is formed, which is faster to evaluate. In what follows, we refer to the functions of the original, computationally expensive problem as original functions. In the literature, metamodel is often used for all objective functions, therefore all objective functions are assumed as computationally expensive. Neural networks [72, 89, 9, 79], Kriging [68, 56, 102, 78] and polynomial regression [49, 118]

are examples of algorithms used for function approximation.

The general steps for using function approximation in an EA can be divided into two stages consisting of ten steps and most of the algorithms discussed in this section follow these steps. In what follows, we present a function approximation framework that captures the core of the EA proposed in the literature utilizing function approximation. Additionally, it is assumed that one type of metamodel is used for all objective and constraint functions involved although it is possible to use different metamodels for different objective functions.

Stage 1

1. Initialize the population either randomly or using a sampling method.
2. Evaluate the individuals of the population using the original functions and add them to an archive.
3. If a prefixed number of generations is completed, go to stage 2.
4. Use EA operators (selection, crossover and mutation) to create a new population and go to step 2.

Stage 2

5. *Build or update a metamodel for each computationally expensive objective and constraint function using the individuals from the archive.*
6. Use EA operators (selection, crossover and mutation) to create a new population.
7. For each objective and constraint function, evaluate the individuals of the new population either using the metamodel or the original function (fixed or adaptive evolution control strategy).
8. If a stopping criterion is met, select the nondominated individuals from step 7 as the final population and stop. Otherwise, continue.
9. *Select individuals from step 7 for re-evaluation using the original functions if needed.*
10. *Add the individuals from step 9 to the archive and go to step 5.*

In stage 1, a general evolutionary algorithm works and in stage 2, an EA with a metamodel is used. In step 1, a population is initialized either randomly or using a sampling method such as Latin hypercube sampling [86]. Individuals of the population are evaluated using the original functions and the evaluated individuals are added to an archive in step 2. If a prefixed number of generations is completed in step 3, stage 1 is terminated and the archive is carried over to stage 2. Otherwise, a new population (offspring population) is generated using EA operators such as selection, crossover and mutation in step 4. A prefixed number of generations is required to obtain an archive of a fixed size. In most of the papers cited in this paper, there is no explicit criterion mentioned for choosing the prefixed number of generations. However, it is mentioned in [7] that the prefixed number of generations depends on the dimensions of the problem (both in objective and decision spaces) and the budget of evaluations with the original functions. For instance, in three popular algorithms known as ParEGO [68], SMS-EGO [102] and MOEA/D-EGO [136], a data set of size *11n-1* was used for training the metamodels. We provide the size of the data set used in different algorithms in Table 1.

After completion of stage 1, a metamodel is created for each computationally expensive objective and constraint function to work with the EA algorithm for evaluating individuals in stage 2. A metamodel is created using the individuals from the archive in step 5. In step 6, an offspring population is generated using EA operators. In step 7, either the metamodel or the original functions are used to evaluate individuals i.e. a fixed or an adaptive evolution control strategy is used to manage the metamodel. In step 8, if a termination criterion such as maximum number of generations or function evaluations is met, nondominated individuals from the last population are selected as the final population. Otherwise, some individuals are selected from step 7 and re-evaluated using the original functions in step 9. There are different criteria mentioned in the literature to select individuals for re-evaluation e.g. selection of nondominated individuals, using expected improvement [64], expected hypervolume improvement [102] etc. These individuals are then added to the archive in step 10 to update or re-train the metamodel. If the size of the archive is prefixed, some individuals (e.g. random or dominated individuals) are eliminated from the archive using a predefined criterion.

3.2 Challenges in using metamodels

Before going into the details of different algorithms, we mention here major challenges when applying function approximation in multiobjective optimization when compared to single objective optimization. It is not straightforward to use a metamodel with an EA because several challenges exist which affect the performance of the metamodel used. These challenges are also the main differences in several algorithms in using function approximation.

1. *Using the metamodel:* In case of single objective optimization, often one metamodel is used for the approximation of the objective and constraint function. Using metamodels in multiobjective optimization is not that trivial. For example, one can use different metamodels for different objective functions, one metamodel for all objective functions, different metamodels for one ob-

jective function or a single metamodel for scalarized objective functions. The same options may arise for different constraints. In the literature, most of the algorithms use a single metamodel for all objective functions. In addition, there are no guidelines for using different metamodels for different objective functions.

2. *How to update the metamodel:* Updating the metamodel is an important step both in single objective and multiobjective optimization. In case of single objective optimization, different strategies have been used for updating the metamodel e.g. lower confidence bound [32], probability of improvement [126], expected improvement [112, 64] etc. In case of multiobjective optimization, two goals, convergence and diversity have to be achieved in contrast to single objective optimization. In case of multiobjective optimization, one of the simplest ways is to select the nondominated individuals for re-evaluation [60, 61] using the original functions. Alternatively, a clustering method such as K-means clustering [54] can be applied to cluster the individuals in the objective space and the representative individuals are re-evaluated using the original functions [43, 62]. The representative individuals can be the individuals closest to each cluster center or the best individuals (with highest fitness values) in each cluster. Individuals with low approximation accuracy can also be selected for re-evaluation [14, 38] and this selection may be effective in improving the approximation accuracy of the metamodel.

   In the literature, criteria for updating the metamodel in single objective optimization are also modified to be used in multiobjective optimization. For instance, expected improvement in [68, 57, 66, 24, 130] and probability of improvement in [24] are used to update the metamodel in multiobjective optimization. In addition, expected hypervolume improvement is used in [37, 36, 115] to select the individuals for re-evaluation so that the metamodel can be updated. It is to be noted that infilling criterion is sometimes used as a synonym for updating criterion in the literature.

3. *Training time for the metamodel:* Training time is also an issue in case of single objective optimization but when moving to multiobjective optimization, training or updating time becomes more influential. For example, different metamodels for different objective functions may take a different amount of training data (archive size) which increases the training time when compared to single objective optimization. In the literature, unfortunately, most of the algorithms do not mention the training or updating time for the metamodel. It may happen that training time of the metamodels is substantial and whole aim of reducing computation time is jeopardized.

   In addition to the three challenges above, two challenges exist which are common to both single objective and multiobjective optimization can influence the performance of EA while using metamodel.

1. *Choice of the metamodel:* In the literature, there is very little guidance about the choice of the metamodel for approximation of computationally expensive functions. A metamodel is either selected randomly or due to its popularity in the area with which the problem is associated. For instance, in [72], radial basis neural network was used as a metamodel for approximation of objective functions in coastal aquifer management problem and it was mentioned that neural networks were popular for groundwater applications. Kriging was used in [56] due to its promising nature for building accurate global approximations.In [95], different metamodelling techniques were used for different problems i.e. RBF with linear kernel function for ZDT problems and Kriging for UF [137] problems. The reason mentioned was that RBF was used due to simplicity of the function landscape in ZDT problems and and Kriging was used because functions in UF problems are highly non-linear and test with RBF showed a very low accuracy when used for UF problems. However, the algorithm was developed for black-box computationally expensive problems where function landscape of the problems in question is not known a priori. Moreover, Kriging is most popular technique when compared to others because of its ability to provide uncertainty for the approximated values. To overcome the problem of choosing of a metamodel, an ensemble of metamodels described in Section 2 is also used in the literature [118, 79]. However, there are still some open challenges related to the ensemble of metamodels such as what should be the criterion for choosing different metamodels or how different metamodels can be used simultaneously?

2. *When to update the metamodel:* It is also an important issue to decide when to update the metamodel or in other words, is there any need to update the metamodel. For example, before selecting individuals in step 9, one can check whether the existing metamodel is accurate enough to predict objective function values in the next generation. If so, the metamodel is not updated and the existing metamodel is used for approximation. An offspring population has to be generated before updating the metamodel so that the metamodel accuracy can be measured for the individuals of the offspring population. There may

be a possibility that the metamodel is never updated and the metamodel trained after stage 1 is used in all generations. In that situation, steps 9 and 10 are eliminated from stage 2. In the literature, there is very little focus on the question of when to update the metamodel.

3. *Handling constraints:* While using metamodels in constrained problems, an appropriate set of solutions is needed to train them. Most of the metamodel-based algorithms in the literature are not developed to handle constraints. In addition, few algorithms which are developed to handle constraints assume that feasible solutions are available to train metamodels for constraint function. In many problem, obtaining a feasible solution is not trivial e.g. C1-DTLZ1 problem [55] with very small feasible region. In such instances, using metamodels for constraints can be very challenging. We provide more details of handling constraints while describing different algorithms.

Steps 5, 9 and 10 representing the first three main challenges are written in italics in the function approximation framework to indicate the steps where algorithms using function approximation in multiobjective optimization mainly differ from each other. Other three challenges, selection of a metamodel, when to update the metamodel and size of the data to train the metamodel are not the major differences in the algorithms.

In what follows, 30 algorithms and from the literature are discussed utilizing the three main steps (5,9 and 10) of the function approximation framework and attention is paid to their efficiency in reducing the computation time. These algorithms are classified according to the number of metamodels they have used e.g. single metamodel, multiple metamodels independently or ensemble of metamodels. Among single metamodel based algorithms, Kriging is used more often than other metamodels. As mentioned, the main advantage from Kriging is the uncertainty information besides the predicted objective and/or constraint function values. This uncertainty information can be further utilized in the algorithm. For instance, uncertainty information is used while updating the metamodel in [36, 19, 103, 68].

In contrast to single metamodel based algorithms, some algorithms use metamodels independently. Moreover, some algorithms use ensemble of metamodels to solve the problem of choosing a metamodel. In what follows, three categories, algorithms based on single metamodel, algorithms based on multiple metamodels and algorithms based on ensemble of metamodels, are used to describe different algorithms. It was found that the number of papers in the literature belonging to the function approximation are largely skewed towards the first category, i.e. algorithms based on a single metamodel. Hence a sub-classification based on the usage of metamodel within different algorithms is devised to enhance clarity and readability. Thus we further classify the algorithms using single metamodel into Kriging and non-Kriging based algorithm. Due to a limited number of articles in other categories, such a sub-classification is not used for others. It is also worth mentioning that most of the algorithms are based on dominance based EAs and are generic for using any metamodel. At the end of this section, a comparison is presented based on which metamodel and type of EAs are used, what is the evolution control strategy and what are the characteristics of the optimization problems.

Parameter values used in these algorithms in stages 1 and 2 of the function approximation framework are presented in Table 1, where NA indicates that the parameter is not applicable to the algorithm and not given means that this information is not given in the reference. The table collects information of population size in step 1, size of archive in step 2, prefixed number of generations in step 3 in stage 1 and number of individuals for re-evaluation in step 9 in stage 2, as reported in connection with example problems solved in the papers cited.

3.3 Algorithms based on single metamodel

In this subsection, we discuss algorithms using one metamodel for all objective and/or constraint function. As mentioned, to make the a clear structure of the paper and also due to wide applicability of Kriging models, we further classify this subsection into Kriging based algorithms and non-Kriging based algorithms. Further these algorithms are discussed year wise, i.e. starting from the year 2008.

3.3.1 Kriging based algorithms

In this section, we discuss algorithms using Kriging. A DACE model [110] which is a Kriging based approximation was used in [102]. This algorithm is known as SMS-EGO which is an extension of efficient global optimization (EGO) [64] for multiobjective optimization problems. In step 5, the metamodel is built for each objective function and to update it in step 9, one individual having maximum contribution to the hypervolume is selected for re-evaluation using the original functions. This individual is then added to the archive in step 10 and the metamodel is updated in the next generation. The size of the archive is not fixed in this algorithm.

SMS-EGO was tested on five benchmark problems (2-5 objectives and 3-6 decision variables) and compared with ParEGO [68] and the algorithm proposed in [57]. The unary hypervolume indicator [143], the R2 indicator [44] and the unary epsilon indicator [144] were used as the comparison criteria. The proposed algorithm performed better than the other algorithms

Table 1: Parameter values used in different algorithms: $|N_P|$= population size in EA, $|N_A|$= size of archive in step 10, $|N_I|$= size of the data set to train metamodels in step 5, $|N_U|$= size of the data set for updating the metamodels and maximum

reference	$\|N_P\|$	$\|N_A\|$	$\|N_I\|$	$\|N_U\|$	FE^{max}
[3]	not given	not fixed	not given	4	200-400
[7]	300	300	200	5	2000
[15]	5	not fixed	3-19	10	23-300
[19]	105-275	*11n-1*	*11n-1*	5	250-300
[27]	50-100	not fixed	50-100	not fixed	500-5000
[46]	1000	2000	2000	NA	2000
[49]	100	not fixed	not given	5	not given
[56]	500	not fixed	12	1	66
[72]	40	not fixed	100	1	3100
[76]	20-50	not fixed	16-40	not fixed	58-497
[78]	30	not fixed	30	not fixed	148-901
[79]	100	1000-2000	1000-2000	2	8000-30000
[80]	not given	not fixed	6-18	10	58-242
[81]	140-200	not fixed	30	10-40	140-3600
[85]	300	not fixed	$10n$	10	1000-5000
[90]	100	not fixed	100	not fixed	267-8988
[89]	50	not fixed	50	50	2000
[96]	26	not fixed	150	1	1606
[95]	26-50	not fixed	150	1	NA
[98]	300	not fixed	$10n$	40	1000-2000
[97]	100	not fixed	$10n$	10	200-2000
[101]	100	800	100	1	30000
[102]	not given	not fixed	$11n$-1	1	130
[106]	not given	not fixed	*(n+1)(n+2)/2*	1	200-500
[109]	25	not fixed	150	2	500
[118]	not given	not fixed	1000	not fixed	6000-11900
[119]	NA	not fixed	15-58	1	50-150
[122]	20-60	not fixed	50	1	3000
[136]	300-595	not fixed	$11n$-1	5	200-300
[139]	140	not fixed	6-10	6-10	190-228

in all cases when hypervolume was used as the performance measure. In terms of other comparison criteria, it performed worse than the other algorithms for three problems.

In [78], Kriging was used as a metamodel with a modified version of NSGA-II. In this algorithm which is known as K-MOGA, the metamodel is used for approximating each objective function in step 5. To update the metamodel in step 9, domination status is measured for each individual evaluated using the metamodel. Individuals which change the domination status are re-evaluated using the original functions and added to the archive in step 10. To check the domination status, minimum of minimum distance (MMD) is measured. To calculate MMD, individuals are partitioned into two sets in the decision space, nondominated (x^{nd}) and dominated (x^{d}). MMD is calculated as, $MMD = min\left\{\left\|\hat{f}(x^{nd}) - \hat{f}(x^{d})\right\|\right\}$, where $\hat{f}$ is the predicted objective vector using the metamodel. MMD is then projected to each objective function axis to obtain $MMD\hat{f}_i$ (for i = 1,. . .,k). Thereafter, a threshold $s_i(x) \leq MMD\hat{f}_i$ is specified by using the standard deviation obtained ($s_i(x)$) from the Kriging model for each objective function. The individuals which do not satisfy this threshold are re-evaluated using the original functions. The size of the archive is not fixed in this algorithm.

K-MOGA was tested on five benchmark (two objectives and 3-6 decision variables) and two real-world problems (two objectives and 2-4 decision variables) and compared with MOGA. For comparison, nondominated individuals from both algorithms were visualized and similar solutions were obtained in fewer function evaluations with K-MOGA.

The same algorithm was extended to KD-MOGA in [76] with one additional element. In KD-MOGA, a fixed number of individuals is generated using constrained maximum entropy design after step 9, which is an extension of unconstrained maximum entropy design [25]. These individuals are then added to the population for the next generation.

KD-MOGA was tested on the same set of problems as K-MOGA with one more real-world problem (two objectives, four decision variables and four constraints). It was compared with both EAs (modified version of NSGA-II) and K-MOGA using visualization of the nondominated individuals. A similar performance was obtained in fewer function evaluations with KD-MOGA.

Kriging was used as a metamodel in [56]. This paper is based on an algorithm called combined AASO-AAMO (adaptive approximation in single objective optimization - adaptive approximation in multiobjective optimization) [133] which uses a multiobjective genetic algorithm [133] as an EA. The metamodel is built for each objective and constraint function in step 5. To update the metamodel in step 9, a fixed number of individuals is selected for re-evaluation using the original functions and added to the archive in step 10. The individuals having the best value according to the maxmin distance design criterion [63], i.e. the largest value of minimum distance from individuals in the archive in the decision space, are selected for re-evaluation. The size of the archive is not fixed in this algorithm.

The proposed algorithm was tested on a fatigue design MOP with two objectives, 17 decision variables and 11 constraints. It took 70 hours to complete 25 generations but the efficiency of the algorithm was not compared to any EA without using the metamodel. The authors mentioned that it was practically impossible to do the optimization without using approximation.

In [136], Kriging was used as a metamodel with the decomposition based EA MOEA/D [135]. The algorithm is known as MOEA/D-EGO. In MOEA/D-EGO, after evaluating a fixed number of individuals in steps 1-4, the metamodel is built for the scalarized objective function. Chebyshev scalarizing function [88] is used to convert multiobjective optimization problem into single objective optimization problems. After using the metamodel in step 7, the expected improvement (EI) [64] is calculated for each subproblem. Expected improvement is then maximized (for each subproblem) using MOEA/D-DE [77] for a fixed number of generations. In other words, the scalarized problem is changed into another problem to maximize EI. To update the metamodel in step 9, firstly, all individuals after the local search, which are different from the individuals in the archive (in the decision space) are selected. After this, K-means clustering is used to cluster the weight vectors (used in MOEA/D initially) associated with the individuals selected above. From each cluster, an individual with maximum EI is selected and re-evaluated using the original functions. These individuals are then added to the archive in step 10. Moreover, to reduce the training time, a fuzzy clustering [13] is used for selecting a fixed number of individuals for training or updating the metamodel.

MOEA/D-EGO was tested on 12 benchmark problems (2-3 objectives and 2-8 decision variables) and compared with ParEGO [68] and SMS-EGO [103]. Hypervolume and inverted generational distance (IGD) were used as the comparison criteria. The proposed algorithm outperformed on seven problems when compared to ParEGO and performed similar to SMS-EGO. It was also compared with MOEA/D on two problems (two objectives and 2-8 decision variables) and outperformed with IGD as the performance criterion.

A multiobjective variable-fidelity optimization (VFO) algorithm was proposed in [139] where Kriging was used as a metamodel with NSGA-II. Initially, a simplified or approximated problem (having low fidelity functions) is used to replace the original MOP in stage 1. This simplified problem is then solved for a prefixed number of generations using NSGA-II. A fixed number of nondominated individuals obtained af-

ter this step is re-evaluated using the original functions and stored in an archive. These individuals are selected based on a simple inter-individual distance metric in the objective space (steps 1-4). These individuals are then used to construct a Kriging model for each objective function in step 5 of stage 2. For the following fixed number of generations, individuals generated by crossover and mutation in step 6 are evaluated either using the approximated problem functions with Kriging model or just the approximated problem functions (step 7) depending on the error [40] in the Kriging model. Thus, an adaptive evolution control strategy is used to manage the metamodel. In other words, if the metamodel is not accurate for an objective function, the original function is used to evaluate individuals. To update the metamodel in step 9, a fixed number of nondominated individuals evaluated using the metamodel is selected for re-evaluation using the original functions. The individuals are selected by computing the error from the Kriging model and added to the archive in step 10. The size of the archive is not fixed in this algorithm.

The VFO algorithm was tested on the ZDT1-3 benchmark problems (with two objectives and five decision variables) and a structural engineering problem (with two objectives and six decision variables). For the ZDT problems, the efficiency of the VFO algorithm was not mentioned in terms of computation time or number of function evaluations.

For the structural engineering problem, an exhaustive search was carried out with different values of decision variables. Around 65 million combinations of decision variable values were evaluated using both original and approximated problem functions. Nondominated individuals after this search were identified and nondominated individuals obtained using original functions were used to compare results of different studies mentioned next. Fourteen studies were performed with changes in the values of three parameters: number of prefixed number of generations in step 3 of stage 1, number of generations before updating the metamodel of stage 2 and number of individuals selected for re-evaluation in step 9 of stage 2. Two out of fourteen studies were performed using NSGA-II without using a metamodel with high and low fidelity functions (to be called case 1 and case 2, respectively). Results from these studies were then compared with the results of exhaustive search with different criteria (inverted generational distance, error ratio, crowding distance [28] and span [75]). Case 1 gave the best results and in comparison with case 1, one of the studies out of thirteen gave similar results (a graphical comparison was performed) in fewer function evaluations.

In [81], VFO algorithm was extended where instead of one global metamodel, multiple local metamodels were used. The authors mentioned that local metamodels are used for high dimensional problems in the objective space. K-means clustering is used in the decision space to partition the data and to build multiple local metamodels. Other details are the same as in VFO.

This algorithm was tested on six benchmark problems (two objectives and 2-30 decision variables) and one real-world problem (three objectives and six decision variables). For benchmark problem, nondominated individuals obtained from this algorithm and with VFO and NSGA-II were visualized. The authors mentioned that the proposed algorithm outperformed on these problems. In case of the real-world problem, the same performance criterion was used as in the VFO algorithm. In this case too, the proposed algorithm performed better than NSGA-II and VFO in the same number of function evaluations.

In [119], Kriging was used as a metamodel. In this algorithm, the main focus was to apply different strategy for objective and constraint functions while updating the metamodel. In step 5, a metamodel is built for each objective and constraint function. To update the metamodel in step 9, for objectives, selection is performed using hypervolume based probability of improvement (POI_{hv}) [24] and for constraints, probability of feasibility (POF) [39] is used. After this, $\gamma = POI_{hv} \times POF$ is obtained and a fixed number of individuals with highest γ values is selected for re-evaluation in step 9. These individuals are then added to the archive in step 10. The size of the archive is not fixed in this algorithm.

The proposed algorithm was tested on two real-world problems with two objectives, 2-3 decision variables and 5-7 constraints. This algorithm was implemented using the SUrrogate MOdeling MATLAB Toolbox (SUMO) [42] and was not compared with any other algorithm.

In [90], Kriging models were used with a differential evolution based EA to approximate the objective functions. After building the metamodels in step 5, individuals were generated using differential evolution operator in step 6 and metamodels were then used for approximation in step 7. For each offspring in step 7, uncertainties of the approximated values were compared to its corresponding parent. In other words, if the uncertainty value of offspring was less than that of parent, it would selected and kept in the population. On the other hand, if the uncertainties of both offspring and parent are comparable, both are kept in the population. All the offspring thus selected were re-evaluated with the original function to update the metamodels in step 9 and added the training archive in step 10.

The algorithm was tested on 12 benchmark problem out of which three had constraints. However, it was not mentioned in the article how the constraints were handled. All these 12 problems had two objectives and the number of decision variables for all problems was not mentioned. The algorithm was also tested on a continuous steel casting and an electrocardiography

problem with four variables and 2-3 objectives. The algorithm was compared with differential based EA [108] and an algorithm based on NSGA-II-ANN [30]. The algorithm obtained better performance when measured with hypervolume in 10000 function evaluations.

In [19], an algorithm called K-RVEA was proposed to solve computationally expensive problems with more than three objectives. In this algorithm, the metamodels were updated based on the need of convergence or diversity. Angle penalized distance [17] and uncertainty information from the Kriging models with the help of reference vectors are used to select individuals in step 9 for updating the metamodels. In addition, extra individuals are removed from the archive in step 10 to further reduce the computation time.

The proposed algorithm was tested on DTLZ and WFG benchmark problems with 3-10 objectives and 10 decision variables. It was also compared with ParEGO, MOEA/D-EGO and SMS-EGO using IGD and hypervolume. In addition to benchmark problems, the algorithm was also tested on a free-radical polymerization problem [91] and also compared with the state-of-the-art algorithms. In the given number of function evaluations, the proposed algorithm performed better than other algorithms.

The same algorithm was also extended to handle constraints in [20]. Three different approaches are used to while training metamodels based on the feasibility of solutions. The proposed algorithm was tested on constrained version of DTLZ problems [55] with 3-10 objectives and 10 decision variables. The authors found out that infeasible solutions are having a vital role on the performance of surrogates.

In [109], a high dimensional model representation (HDMR) [121] was used as a metamodel for each objective function. Kriging was further used within HDMR to approximate its component functions. The algorithm was proposed to handle problems with large number of decision variables. In each HDMR model, n component functions were approximated using Kriging, therefore $n \times k$ (n and k represent the number of decision variables and objectives respectively) Kriging models were built and k HDMR models were built. After building the metamodels in step 5, NSGA-II was used to from steps 6-7. To update HDMR models, a prefixed number of individuals were re-evaluated using the original functions in step 9. These individuals were selected by doing clustering in the decision space and then added to the training archive in step 10. In addition, bounds of the decision variables were updated to limit the search space after metamodels were updated.

The algorithm was tested on 17 benchmark problems from ZDT, DTLZ and CEC09 suite [137] biobjective problems with 15-30 decision variables. It was compared with NSGA-II and Kriging based NSGA-II [84] using IGD and performed better than others in 500 function evaluations.

3.3.2 Non-Kriging based algorithms

In this subsection, algorithms using metamodels other than Kriging such as neural network, support vector regression and polynomial regression are discussed. In [122], a multiobjective parallel surrogate-assisted evolutionary algorithm (MOPSA-EA) was proposed, where a feedforward neural network was used as a metamodel with a steady state EA. The metamodel is built for each objective function in step 5. A fixed number of offspring individuals is generated in step 6 using crossover and mutation and evaluated using the metamodel in step 7. The fitness values of offspring individuals are altered as per the fitness values of parents. The authors mentioned that this approach was motivated by fitness inheritance, where the fitness values of offspring depend on the fitness values of parents.

To get the altered fitness values for offspring individuals, the parents which are used for creating the offspring individual are evaluated with the metamodel and the error is calculated between true fitness values and the approximated fitness values (using the metamodel) for parents. For example, if the errors (for a biobjective optimization problem) between the true fitness values and the approximated fitness values for two parents are (a_e, b_e) and (c_e, d_e) and an offspring individual is generated using these parents having fitness values (e, g), then the new altered fitness values for offspring are $(e + w_1 \times a_e + w_2 \times c_e, g + w_2 \times b_e + w_1 \times d_e)$. The weight coefficients w_1 and w_2 are selected based on the influence of parent individuals on an offspring individual during crossover. An individual is selected in step 9 after getting new fitness values for offspring individuals and added to the archive in step 10 to update the metamodel. To select this individual, a nondominated sorting for offspring individuals and individuals in the archive is performed and nondominated individuals in both sets are identified. Let O_{R1} and P_{R1} denote nondominated individuals in offspring and in the archive, respectively. These individuals are combined and individuals in O_{R1} dominating P_{R1} are identified. Among these individuals, individual having the largest Euclidean distance to its closest individual in P_{R1} is selected and added to the archive. A nondominated sorting is performed again for individuals of the archive and the worst individual (having the worst fitness value and the smallest crowding distance) is removed from the archive. The size of the archive is fixed in this algorithm.

The MOPSA-EA was tested on the ZDT1-4 and ZDT6 benchmark problems (2 objectives and not explicit information of the number of decision variables) and on a manufacturing MOP (with two objectives and 11 decision variables). The Y [31], Δ [31], Ω [122] and S [122] metrices were used to compare the proposed algorithm with SMS-EMOA [35], MAES [37] and NSGA-II-ANN [92]. For the same number of function evaluations, MOPSA-EA performed better than other algorithms in Y, Ω and S metrics and in Δ, SMS-EMOA

and MOPSA-EA performed equivalent. As the Pareto front was not known for the manufacturing problem, only the S metric was used as the performance criterion and MOPSA-EA gave better results in S than the other algorithms in the same number of function evaluations.

A quadratic polynomial approximation was used as a metamodel with the EA μ-MOGA [22] in [80]. In this algorithm, the bounds of decision variables are updated after every generation using a trust region algorithm. In step 5, the metamodel is built for each objective and constraint function. To update the metamodel in step 9, a fixed number of uniformly distributed individuals (P_a) from nondominated individuals is re-evaluated using the original functions. The nondominated individuals in the decision space after re-evaluation are stored in a set P_e. The set $P_o = P_a \cap P_e$ is determined which is then used to calculate a reliability index $N(P_o)/N(P_a)$, where $N(P_o)$ and $N(P_a)$ are the numbers of individuals in P_o and the number of nondominated uniformly distributed individuals evaluated using the metamodel, respectively. This reliability index is used to update the bounds of the decision variables in the next generation with a trust region algorithm. The trust region radius is updated according to [4] and the algorithm terminates if the trust region radius is smaller than a predefined limit or after a fixed number of generations (step 8). A Latin hypercube design is used for sampling the decision variables with updated bounds. These individuals with the updated bounds are evaluated with the original functions which are used to update the metamodel. Step 10 is not applicable in this algorithm as individuals after step 9 are not added to the archive.

The proposed algorithm was compared with μ-MOGA using two benchmark problems (two objectives and 2-3 decision variables) and one structural engineering problem (two objectives, three decision variables and one constraint). Using generational distance as a performance metric for the benchmark problems and visual comparison for the structural engineering problem the quality of the obtained set of nondominated solutions within a fixed budget of function evaluations using μ-MOGA with metamodel was better.

A feedforward neural network was used as a metamodel with NSGA-II in [89]. The metamodel is built for each objective function in step 5 if a threshold based on a predicted tolerance is satisfied. This predicted tolerance, which is an indication of the accuracy of the metamodel, is calculated (however, calculation of the predicted tolerance is not detailed in the paper) after every generation. If this predicted tolerance is less than a user-specified tolerance, then the metamodel is used in the next generation. Otherwise, the original functions are used to evaluate individuals. The predicted tolerance is updated after every generation and again a decision is to be made either to use the original functions or the metamodel and thus, an adaptive evolution control strategy is used. Individuals obtained after every generation are added to the archive (steps 9 and 10) and as a result, the size of the archive grows with generations. In this algorithm, the metamodel is not updated after every generation but after every generation, it is checked whether the existing metamodel is sufficient enough to predict function values to the extent of accuracy required.

This algorithm was tested on an iron induration MOP with two objectives, 22 decision variables and three constraints. To check the efficiency of the proposed algorithm, a graphical comparison was presented to the solutions from the algorithm and NSGA-II without using any metamodel. Similar nondominated individuals were obtained in 50% fewer function evaluations and for the same number of function evaluations, better nondominated individuals were obtained.

In [15], an extension of algorithm proposed in [80] was proposed. The main differences include the type of metamodel, stopping criterion in step 8 and selection of individuals for re-evaluation for updating the metamodel in step 9. The algorithm proposed utilizes a radial basis function as a metamodel. In addition to the stopping criteria posed in [80] mentioned earlier, the algorithm also terminates if the bounds of the decision variables are equal to predefined limit and the reliability index is equal to one. In step 9, the individuals are selected using an inherited Latin hypercube design (ILHD) [131] and a local-densifying strategy (to reduce the possibility of an ill-conditioned RBF matrix) for re-evaluation and subsequently updating the RBF.

The proposed algorithm was tested on eight different benchmark problems (two objectives and 1-5 decision variables) and a structural engineering problem (two objectives and five decision variables). The results were compared with μ-MOGA without any metamodel and the algorithm proposed in [80]. In case of benchmark problems, the proposed algorithm obtained better values for spread and convergence metrices [31] in fewer function evaluations. For the structural engineering problem, the proposed algorithm was not tested using μ-MOGA without any metamodel.

In [72], a multiobjective surrogate assisted (MOSA) algorithm was proposed, where a modular neural network (MNN) [71] was used as a metamodel with NSGA-II. A metamodel is built for each objective function in step 5. A fixed number (equal to the population size of NSGA-II) of better performing individuals (the authors did not mention any criterion for defining better performing individuals) is used as the population of NSGA-II and the nondominated individuals (A) after this step are determined. The nondominated individuals after evaluating the offspring individuals (generated in step 6) are compared with A to select one individual for re-evaluation (step 9). To do this, the individuals evaluated using the metamodel are clustered into two sets A^1 and A^2 as shown in Figure 1.

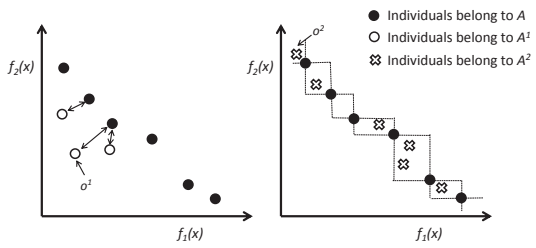


Figure 1: Selection of individual evaluated using the metamodels

The first set (A^1) consists of the individuals which dominate at least one individual of A while the second set (A^2) consists of the individuals that do not dominate nor are dominated. For each offspring in A^1, the Euclidean distances between the offspring and the individuals in A are calculated. As shown in left part of Figure 1, individual (o^1) with the largest distance is selected for re-evaluation using the original functions. In case A^1 is empty, an individual is selected from the set A^2 for re-evaluation. To select the individual, for each offspring individual in A^2, the normalized perimeter for rectangles created between consecutive individuals in A (as shown in the right part of Figure 1) is calculated. An individual located in the rectangle with the largest perimeter (o^2) is selected for re-evaluation using the original functions. In case both A^1 and A^2 are empty, offspring are generated again. However, the authors did not mention about how this algorithm can be used for more than two objectives. The selected individual is added to the archive to update the metamodel in step 10. The size of the archive is not fixed in this algorithm.

The algorithm was used to solve a coastal aquifer management optimization problem with two objectives and eight decision variables. The time required to evaluate the original functions was 26 hours while the time to train the metamodels was 63 minutes. A graphical comparison was presented between MOSA and NSGA-II and the proposed algorithm gave similar results in fewer function evaluations.

In [46], support vector regression was used as a metamodel for approximation of each objective function with NSGA-II. The main focus in this algorithm is to use different basis functions for different kinds of variables such as discrete, continuous and categorical variables. It is to be noted that this algorithm does not use an ensemble of metamodels as only one prediction is obtained for each objective function by using different basis functions for the different kinds of variables. To convert categorical values into real number, dummy coding is used. In step 5, the metamodel is built for each objective function. Steps 9 and 10 are not applicable to this algorithm as the metamodel and the archive are not updated.

The proposed algorithm was tested on one real-world problem (two objectives and 10 decision variables) and compared with NSGA-II. Out of 10 variables, 5 were continuous and 5 were categorical variables. A visualization of nondominated individuals in the objective space was performed to compare the two algorithms. The authors mentioned that the proposed algorithm performed similar to NSGA-II in fewer function evaluations.

In [101], support vector regression was used as a metamodel. This algorithm is known as HO-MOMA where, a metamodel is built for each objective function in step 5. After evaluating new individuals in step 7, a local search is used to optimize the fitness function obtained from the metamodel evaluations. This is done as follows. Firstly, several nondominated fronts are obtained with nondominated sorting. Then, in each front, sorting is performed according to the first objective function values in the ascending order. A reference point is then calculated for each individual in each front using these sorted values. After calculating the reference point, a fitness value for each individual is calculated. Let $\hat{f} = \left\{\hat{f}_1, \hat{f}_2\right\}$ be the predicted objective vector for one individual and $r = \{r_1, r_2\}$ the reference point for that individual. Fitness is then calculated as

$$fitness = \begin{cases} (r_1 - \hat{f}_1)(r_2 - \hat{f}_2) & \hat{f}_1 < r_1 \text{and} \hat{f}_2 < r_2 = 0 \\ (r_1 - \hat{f}_1) & \hat{f}_1 > r_1 \text{and} \hat{f}_2 < r_2 = 0 \\ (r_2 - \hat{f}_2) & \hat{f}_1 < r_1 \text{and} \hat{f}_2 > r_2 = 0 \\ -d(r, \hat{f}) & otherwise, \end{cases} \tag{3}$$

where $d(r, \hat{f})$ is the Euclidean distance between r and $\hat{f}$. This fitness is then optimized using CMA-ES [45] as the local search algorithm. To update the metamodel in step 9, nondominated individuals obtained after the local search are added to the archive in step 10. The size of the archive is fixed in this algorithm and extra individuals are removed from it randomly. The authors mention the study for more than two objectives is a future research.

HO-MOMA was tested on 14 benchmark problems with two objectives and 10-30 decision variables. The algorithm was compared with NSGA-II and ASM-MOMA [99] with hypervolume as the performance criterion. The proposed algorithm performed better than the others in nine out of 12 problems.

In [98, 97], an extreme learning based MOEA/D-DE [77] was used. This algorithm is known as ELMOEA/D-DE and inspired by MOEA/D-RBF. The main focus in this algorithm is to use the metamodel for higher dimensional problems in the decision space. Extreme learning is a single-layer feedforward neural network proposed in [48]. In step 5, the metamodel is built for each objective function. To update the metamodel in step 9, the same procedure is used as in MOEA/D-RBF. In addition, a minimum distance (in the decision space) is maintained between the individuals to be selected for re-evaluation. After adding these individuals in the archive, inferior solutions (in terms of scalarized

single objective problem) are removed. Otherwise, the closest individual in the objective space is replaced by the individual obtained in step 9. This is done to ensure that every new solution is added to the archive for updating the metamodel.

ELMOEA/D-DE in [98] was tested on ZDT problems with 10-60 decision variables and compared with MOEA/D-DE and MOEA/D-RBF.Later in [97], the algorithm was tested on ZDT, DTLZ and WFG problems with 2-5 objectives and 5-60 decision variables and on an airfoil shape optimization problem with two objectives and 12 decision variables. The algorithm was also compared using addictive-Epsilon and R_2 [143] as the performance indicators. For the given number of function evaluations, the proposed algorithm obtained better performance.

In [3], an algorithm called gap optimized multiobjective optimization using response surfaces (GOMORS) was proposed where radial basis functions were used to approximate the objective functions. After building the metamodels in step 5 for each objective function, an EA proposed in [129] was used for optimization from step 6-7. After step 7, an another optimization problem was solved using the same EA by reducing the bounds of the decision variables. This problem was referred as gap optimization problem in the article. In step 9 for updating the metamodels, four criterion were used and one individual corresponding to these criterion was selected and added to the training archive in step 10. Four criterion were based on hypervolume, distance to the individuals in the decision space, distance to the individuals in the objective space and hypervolume in the gap optimization problem. A maximum number of function evaluations was used as the termination criterion.

The algorithm was tested on 11 benchmark problems with 8-24 decision variables and two objectives. In addition, a groundwater remediation problem with 6-24 variables and two objectives was also solved. The algorithm was compared with NSGA-II and ParEGO using hypervolume in 200-400 function evaluations. The proposed algorithms obtained better performance than others in the given number of function evaluations.

In [96], an algorithm called surrogate assisted local search memetic algorithm (SS-MOMA) was proposed where RBF as a metamodel was build on the single objective problem after converting multiobjective optimization problem using a scalarization function. Two common ways of scalarization i.e. Tchebycheff and weighted sum were tested where weights were generated randomly and the reference point in Tchebycheff scalarization was the current individual objective function values. After generating the offspring population in step 6, metamodels were built locally for each individual. For instance, if the offspring population of size 100 was generated, one metamodel for each individual (i.e. 100 in total) was built. As multiobjective optimization problem was converted into single objective using the scalarzing function, SQP algorithm was used to obtain the solutions. All solutions thus obtained were re-evaluated with the original functions and added to the training archive. Afterward, these solutions were combined with the parent individuals and non-dominated sorting [31] was performed to obtain the population for the next generation. In this way, after every generation metamodels were updated with a number equal to the population size used.

The algorithm was tested on three benchmark problems with 15 variables and two objectives. The algorithm was not compared with any other algorithm. However, in 1606 function evaluations, Tchebycheff scalarization performed better than weighted sum in terms of generational distance [28] and one diversity metric used.

In [27], RBF as metamodels were built for every objective and constraint function in step 5. A $(\mu + \lambda)$ evolution strategy mutation operator was used to generate new individuals in step 6 which were then evaluated using the metamodels in step 7. After using metamodels for each objective and constraint functions, feasible solutions were found and a nondominated sorting was performed on these feasible solutions. The best individuals i.e. individuals in the first front were re-evaluated with the original functions to update the metamodels in step 9 and added to the training archive in step 10. Therefore, the maximum number of individuals to be updated was λ. In addition, initial training of metamodels in step 5was performed without considering any feasibility or infeasibility of solutions. However, the authors clearly mentioned that the algorithm is not expected to work well on problems when the feasible region is empty.

The algorithm was tested on 15 benchmark problems with 2-15 decision variables, 2-5 objective functions and 2-13 constraint functions. In addition it was also tested on a manufacturing and robotics problem with 3-7 decision variables, 2-5 objectives and 2-8 constraints. Hypervolume was used to compare the performance of the algorithm against constrained version of $(\mu + \lambda)$ evolution strategy [10] and NSGA-II. The algorithm performed better in 500-5000 function evaluations.

In [106], RBF was used as a metamodel for each objective and constraint function. It was assumed that at least one feasible solution is available to train the metamodels in step 5. To create new individuals in step 6, two different approaches were tested. In the first one, uniform random individuals were generated over the search space and in the second one, individuals were generated by adding Gaussian perturbation centered at the nondominated individual that has the most isolated objective function values. The most isolated individuals is identified by measuring the distance among nondominated individuals. Metamodels were then used to approximate the objective and constraint function values in step 7. Afterward, nondominated so-

lutions with minimum constraint violations were found. Among these individuals, one individual was selected for re-evaluations in step 9 to update the metamodels. One individual was selected by the weighted sum (with equal wights) of two criterion. One criterion was the distance of solutions (in the decision space) obtained in step 7 from the individuals in the training archive and the second criterion was the distance from the nondominated individuals from the last generation in the objective space. Selected individual was then re-evaluated and added to the archive in step 10.

The algorithm was tested on 28 benchmark problems with 2-5 objectives, 2-15 decision variables and 1-11 constraint functions. the algorithm was compared with its two different versions (based on the generation of individuals in step 7), NSGA-II and DirectMultiSearch(DMS) [26] using hypervolume as the performance indicator. Overall, the version where individuals were generated using Gaussian distribution performed better in the given number of function evaluations.

3.4 Algorithms based on multiple metamodels

In this subsection, algorithms using multiple metamodels are discussed. These metamodels are used independently to predict objective and/or constraint functions. In [49], three independent case studies were performed, where three different metamodels (polynomial approximation, Kriging and radial basis function) were used with NSGA-II. The metamodel is built for each objective function in step 5. The nondominated individuals obtained after this step are improved using a local search algorithm with sequential quadratic programming. To perform local search, a variant of ϵ-constraint algorithm [88] is used i.e. one of the objectives is optimized and other objectives are converted to equality constraints. The improved individuals are combined with individuals from step 7 and dominated and duplicated individuals are eliminated. To update the metamodel in step 9, a fixed number of individuals is selected from the remaining individuals using K-means clustering in the objective space for re-evaluation using the original functions. These individuals are then added to the archive in step 10. The size of the archive is not fixed in this algorithm.

This algorithm was used to solve a heat sink MOP with two objectives and three decision variables. Three different studies were performed to compare the results while using different metamodels. In the first one, nondominated individuals were identified while using each metamodel involving steps 1-8 (i.e. without updating the metamodel) of the function approximation framework. Five representative individuals among nondominated individuals were selected using K-means clustering and re-evaluated with the original functions. The proposed algorithm with Kriging gave the least error in objective function values for these five individuals. In the second study, five representative individuals obtained using each metamodel were re-evaluated with other metamodels. For example, individuals obtained while using polynomial approximation were re-evaluated with Kriging and radial basis function. Individuals obtained while using Kriging when re-evaluated with other metamodels gave the least error in objective function values. In the third one, nondominated individuals were obtained utilizing steps 1-10 (i.e. by updating the metamodel in steps 9 and 10) and results were compared graphically in the objective space with the first study. While using Kriging and radial basis function, better results were obtained when compared to results of the first study and while using polynomial approximation, similar results were obtained. However, the proposed algorithm was not tested using NSGA-II without any metamodel and the efficiency of the algorithm was not mentioned in terms of computation time or the number of function evaluations.

In [7], five different radial basis functions (RBFs) with different basis functions were used as metamodels along with the EA MODE-LD+SS [6]. However, these metamodels were used independently for each objective function but individuals from each metamodel evaluation were used to update all metamodels as discussed next. A metamodel is built for each objective function in step 5. In step 9, to select the individuals for updating the metamodel, one individual from each metamodel evaluation is selected for re-evaluation using the original functions and added to the archive in step 10. To select one individual, a set of uniformly distributed weight vectors $(\lambda^1, \lambda^2, \ldots, \lambda^N)$ (where, N is the population size of EA) is defined. Next, from each metamodel evaluation, the individual is selected that minimizes the Chebyshev scalarizing function [88] given by $\max_{i=1,\ldots,k} (\lambda_i{}^j |\hat{f}_i(x^j) - f_i^*|)$. The values of objective functions obtained after step 7 and the minimum value of the objective function in the population of EA at the current generation are represented by $\hat{f}$ and f^*, respectively. The authors mentioned that this updating criterion can balance the accuracy of the metamodel and the diversity among individuals. The size of the archive is fixed in this algorithm and extra individuals are removed from it based on their rank.

This algorithm was tested on five different aerodynamic shape optimization problems with 2-3 objectives and 12 decision variables. Hypervolume was used as the comparison criterion for the results of MODE-LD+SS with and without metamodels. The proposed algorithm gave a better hypervolume in fewer function evaluations for all five optimization problems.

In [95], SS-MOMA described in Section 3.3.2 was tested with different metamodels for different problems i.e. RBF with linear kernel function for ZDT problems and Kriging for UF problems because of different function landscape of the problems. Using a particular technique for a problem with an assumption that

function landscape is known a priori raises the question of applicability of the algorithm to black-box problems. In addition, one modification of achievement scalarzing function used where reference point was replaced by the upper bounds of the objective function values at the current generation was also tested. An another locals search algorithm called random mutation hill climber [1] was also tested after building the metamodels for each objective function. To handle constraints, metamodels were first built for each objective function and after scalarization, a constrained SQP algorithm was used to solve the single objective optimization problem.

The algorithm was tested on seven biobjective benchmark problems with 8-15 decision variables and a biobjective airfoil optimization problem with 16 variables and one constraint. Different versions of the algorithm with different scalrzing functions were compared with each other also with NSGA-II using IGD. Overall a low number of function evaluations were used to obtain a given IGD value when Tchebycheff function with reference point as the individual objective function values was used.

3.5 Algorithms based on ensemble of metamodels

In this subsection, we discuss algorithms using ensemble of metamodels. As defined in Section 2, either the weights are given to predicted output of the metamodel or a metamodel is selected based on its accuracy. In [79], an ensemble of metamodels (Kriging, polynomial regression and radial basis function) was used with the proposed generalized surrogate multiobjective memetic algorithm (GS-MOMA). In this algorithm, the offspring population is generated in step 6 before building the metamodel. A fixed number of individuals, equal to $n + (n + 1)(n + 2)/2$, (where n is the number of decision variables) is selected from the archive to build an ensemble of metamodels and a separate polynomial regression metamodel in step 5. The individuals are selected in the decision space using the Euclidean distance between the individuals in the offspring population and the individuals in the archive. The individuals which are closer to offspring individuals are used to build the metamodels.

The ensemble fitness values of offspring individuals are calculated as $F_{ens}(x) = \sum_{j=1}^{m} \omega_j \overline{F^j}(x)$ where ω_j is the weight coefficient for fitness values $\overline{F^j}(x)$ of the jth metamodel (step 7). The weight coefficient is assigned based on the accuracy of the metamodels which is calculated using root mean square error. A local search algorithm (sequential quadratic programming) is also used for single objective optimization of $\overline{F^j}(x)$ to improve the individuals evaluated with the metamodel. The best found individuals after this step and the local search algorithm are combined with the parent population and a selection mechanism is used to select individuals for the next generation. To update the metamodels, an offspring population is generated and individuals are selected from the initial archive based on the mechanism used to build the metamodel. Steps 9 and 10 are not applicable in this algorithm and the size of the archive is fixed.

This algorithm was tested on six benchmark problems (labeled as MF problems in the paper) with 2-3 objectives and with 10-50 decision variables. Three performance criteria namely generalized distance [128], maximum spread [140] and hypervolume ratio [127] were used for comparison between GS-MOMA and NSGA-II without a metamodel. GS-MOMA performed better in all performance criteria for all problems in the same number of function evaluations.

In [118], a surrogate assisted simulated annealing (SASA) algorithm was proposed, where an ensemble of metamodels (quadratic polynomial and radial basis function) was used with constrained Pareto simulated annealing (C-PSA) as an EA. In this algorithm, one of two different metamodels is selected to evaluate individuals based on their accuracy which is calculated using the root mean square error. A fixed number of recently evaluated individuals in stage 1 is used to create the metamodels in step 5. In what follows, for each objective function, either one of the metamodels or the original functions are used to evaluate the offspring individuals in step 7 and thus, an adaptive evolution control strategy is used. In other words, if none of the metamodels is accurate enough (accuracy is compared with a predefined parameter), the original functions are used. To select the individuals for updating the metamodel in step 9, nondominated individuals after step 7 (in case a metamodel is used) are re-evaluated with the original functions and added to the archive in step 10. Dominated individuals are removed from the archive and the remaining individuals are used to update the metamodels. The size of the archive is fixed here and clustering is used via a linkage algorithm [53] to remove extra individuals.

The SASA algorithm was tested on eight benchmark problems [28] with two objectives, 10 decision variables and 1-2 constraints. The hypervolume and displacement metric [12] were used to compare SASA with NSGA-II and the infeasibility driven evolutionary algorithm (IDEA) [105]. For the same number of function evaluations, SASA performed better than the other algorithms for seven problems and for one problem, IDEA performed the best in both performance criteria.

In [85], a similar algorithm to MOEA/D-EGO was proposed, where radial basis function was used as the metamodel. This algorithm is known as MOEA/D-RBF, where the metamodel is built for each objective function instead of scalarized objective function. An ensemble of metamodels (RBF with three different basis functions) is used for each objective function in step 5. A weighted sum of predictions from each metamodel

is used in the ensemble of metamodels and the weights are decided based on the prediction error of the metamodel. To update the metamodel in step 9, the following procedure is adopted.

In MOEA/D let, $W = \{w^1, \ldots, w^N\}$ are a uniformly distributed set of weight vectors and R is the number of points to be re-evaluated in step 9. An another set of uniformly distributed weight vectors $W^R = \{w^{1R}, \ldots, w^{NR}\}$ is defined such that size of W^R ¡ size of W as shown in Figure 2. For each $w^{iR} \in W^R$, a neighborhood is defined $B_R(w^{iR}) = \{w^1, \ldots, w^{Na}\}$, such that $w^1, \ldots, w^{Na} \in W$ are the Na closest weight vectors from W to w^{iR}. After this, from each neighborhood, an individual which minimize the single objective optimization problem used in MOEA/D is selected for re-evaluation and added to the archive in step 10. In this algorithm, a penalty boundary intersection approach [135] is used for decomposition of the MOP into single objective optimization problems. The size of the archive is fixed in this algorithm and the same procedure mentioned above is applied to eliminate extra individuals from the archive.

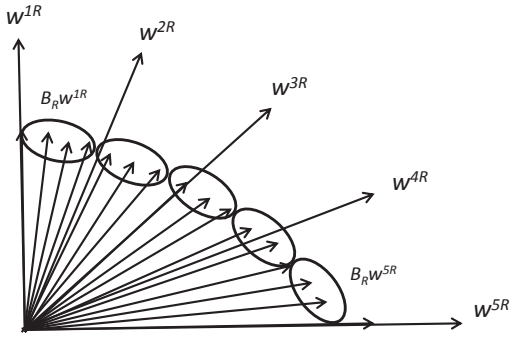


Figure 2: Association of weight vectors from W to W_R

MOEA/D-RBF was tested on five benchmark (two objectives and 8-30 decision variables) and one real-world problem (two objectives and 11 decision variables). It was compared with MOEA/D (for both benchmark and real-world problems) and MOEA/D-EGO (only for benchmark problems) with hypervolume as the comparison criterion. The proposed algorithm obtained better performance in 5 out of six problems in the same number of function evaluations.

3.6 Comparison of function approximation based algorithms

In what follows, a comparison of function approximation based algorithms discussed so far is presented in Figure 3. This figure represents the number of papers with respect to the metamodel (upper part of the figure), EA (lower part of the figure) and evolution control strategy used. The references for these three criteria are mentioned inside the bars of the figure. One should note that some algorithms mentioned in this section were not tested on computationally expensive MOPs. These algorithms are still cited here as they have been proposed for considering the computational burden in any MOP.

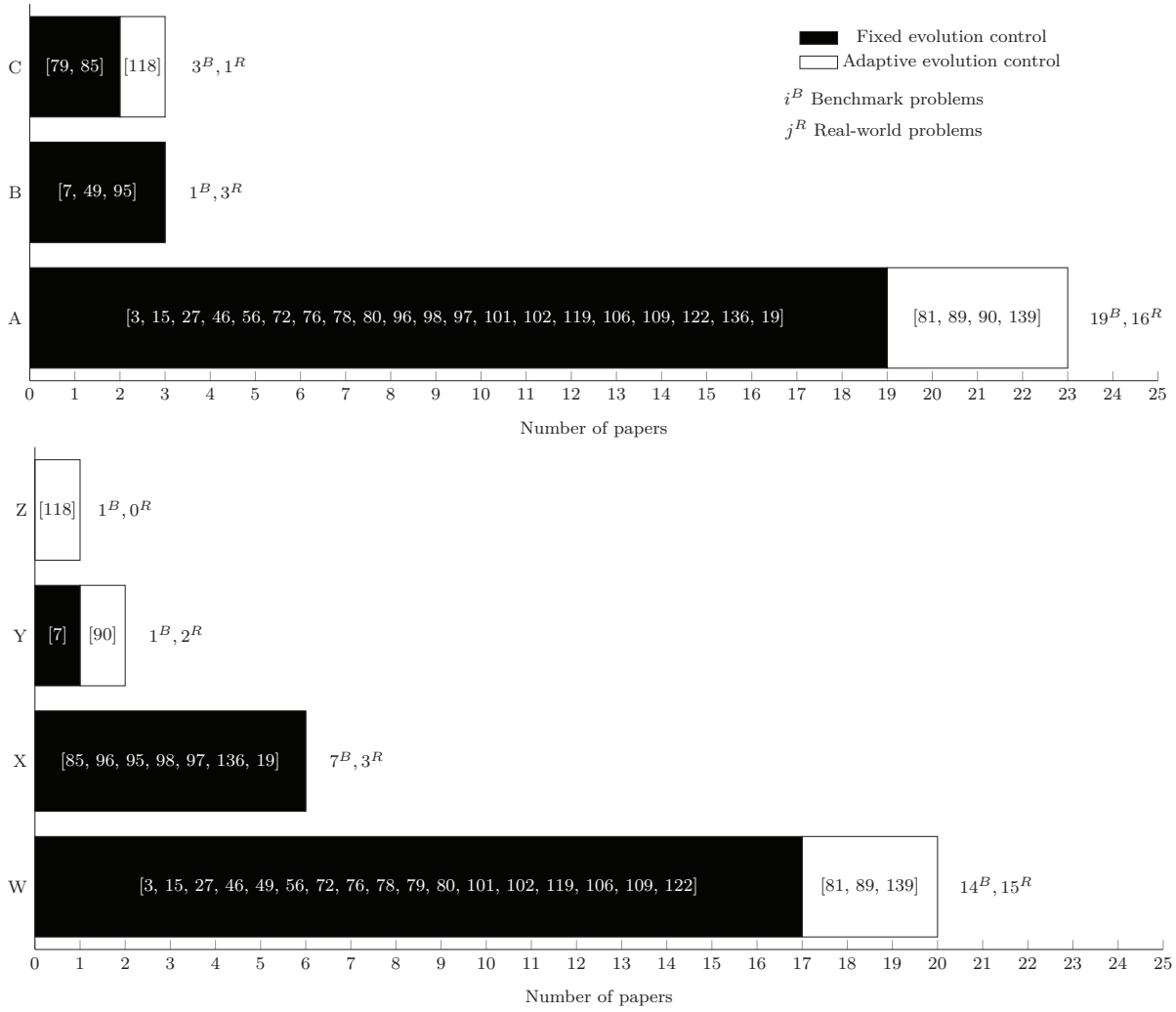
The number of papers which used benchmark and real-world problems is also mentioned in Figure 3. Seven algorithms [79, 98, 101, 103, 118, 122, 136] were tested on benchmark problems, seven algorithms [7, 46, 49, 56, 71, 89, 119] on real-world problems and seven algorithms [15, 76, 78, 80, 81, 85, 139, 19] on both. The efficiency of the different algorithms in terms of computation time or number of function evaluations reduced is very important, especially in the case of real-world problems. In some cases, the authors mentioned that it was practically difficult to do optimization without approximations due to high computation time and the algorithm was not compared with an EA without any metamodel.

As far as selection of a metamodel is concerned, as mentioned in Section 3.1, there is no general rule or correlation between a metamodel for approximation and a particular problem to be solved. As shown in Figure 3, single metamodel is used more than multiple metamodels and ensemble of metamodels. In algorithms using single metamodel, Kriging [56, 78, 76, 81, 103, 119, 136, 139] has been used more than other metamodels such as neural networks [15, 72, 89, 98, 122], support vector regression [46, 101] and polynomial regression [80].

In [139], Kriging was used as a metamodel and the authors mentioned that within different metamodels studies to date, perhaps the most common type of metamodel used is the Kriging model. In [15], radial basis functions (RBFs) were used and the authors mentioned that RBF was a kind of approximation having a very good approximation accuracy. In [7], the authors mentioned that RBFs were very powerful functions to represent complex fitness landscapes. In addition, the authors mentioned that Kriging had a strong mathematical basis, and is probably one of the most powerful interpolation algorithms currently available.

Managing the metamodels or evolution control is also very important as it affects the performance of the metamodel used. As shown in Figure 3, the fixed evolution control strategy was used more than the adaptive evolution control strategy. As mentioned in Section 2, using an adaptive evolution control strategy depends on the accuracy of the metamodel used and using a fixed evolution control strategy implies that either the metamodel is accurate enough for approximation or the metamodel accuracy is not important or not checked. There are only five algorithms [81, 89, 90, 118, 139] where adaptive evolution control was used. For instance in [81], after using Kriging models, uncertainty of the approximated values was compared with a predefined value and if that uncertainty was acceptable, then only Kriging models were further used otherwise original function were used. A similar strategy was followed in [89], where accuracy of neural networks was

Figure 3: Comparison of different algorithms considering metamodel (upper chart), EA (lower chart) and evolution control strategy used and characteristics of optimization problem considered, [A,B,C] = [single, multiple, ensemble] of metamodels based algorithms; [W,X,Y,Z] = [Dominance, Decomposition, DE, SA] based algorithms



measured after every generation and compared with a predefined value. In [90], uncertainties of offspring was compared with the uncertainties of parents and based on the comparison, the decision was made to use Kriging models or original functions. In [118, 139], after training the metamodels, metamodels and original functions were used and an error in approximation was measured and if the error was acceptable, only then metamodels were used further for approximation.

Moreover, the training time for different metamodels was not considered in many papers though training time can be substantially high in some cases, particularly, when the metamodel approximation accuracy is important. Among EAs, dominance based algorithms are more widely used than other algorithms e.g. decomposition based or indicator based. Moreover, in dominance based EAs, NSGA-II was used more often than other EAs.

3.7 Problem and fitness approximation

In addition to function approximation, problem and fitness approximations are also used in the literature to reduce the computation time in multiobjective optimization problems. In problem approximation, the original problem is replaced by a simpler problem which is faster to solve. The main goal in problem approximation is to reduce the computational complexity of the problem. In case of computational fluid dynamics or structural analysis, the governing equations can be modified to reduce the computation time. For example, replacing 3-dimensional Navier-Stokes equations by 2-dimensional Euler equations [73] reduces the computational complexity of the problem. In [73] and [74], Euler equations were used instead of Navier-Stokes equations to solve aerodynamic shape optimization problems. A similar approach was followed in [87, 94], where 2-dimensional Navier-Stokes equations were used for solving aerodynamic shape optimization problems.

Recently, data-driven optimization algorithms [132, 19] are also proposed to solve problems where an analytical forms or simulations models for the objective functions are not available and some data is available obtained through some physical experiments. Therefore, to obtain Pareto optimal solutions, one has to rely upon the data available. In [132], a MOP was formulated using the data available from a trauma system. In addition, the authors found out that accuracy and size of the data used in optimization are conflicting. In contrast, in [18] the data of a small size was used and applied to a blast-furnace problem with eight objectives. In general, algorithms in problem approximation are application-dependent to be tailored to the characteristics of the problem in question.

Two types of fitness approximation were mentioned in the literature [58], fitness inheritance and fitness imitation which are defined in Section 2. In addition to these two types, metamodels are used in the literature in other elements instead of approximating the objective function values e.g. approximation of the rank, classification of individuals into nondominated and dominated set, distance of the nondominated individuals etc. Therefore, we summarize all algorithms using such kinds of approximations under fitness approximation.

Fitness inheritance was originally proposed in [120] to improve the performance of genetic algorithms. Two types of fitness inheritance were proposed in that paper. In the first one, known as averaged inheritance, the fitness values of the offspring were calculated by taking the average of the fitness values of the parents. In the second one, known as proportional inheritance, weighted average of the fitness values of the parents were assigned to offspring and weights were assigned according to common elements between offspring and parents. This algorithm was tested on OneMax problem [2] and the algorithm with fitness inheritance gave better performance (a graphical comparison was performed) than without inheritance in fewer function evaluations.

In what follows, some papers are cited here which use the concept of fitness inheritance. One should note that all these algorithms were not tested on computationally expensive MOPs. In [16], analytical algorithms for computing convergence time and population size were proposed for using the fitness inheritance in MOPs. In [33], the fitness inheritance was used with a binary genetic algorithm (GA) for the ZDT benchmark problems. The authors found a similar performance of the binary GA with and without using the fitness inheritance for problems having convex and continuous Pareto fronts. In case of nonconvex and discontinuous Pareto fronts, the binary GA without inheritance performed better and the authors mentioned that fitness inheritance can only be applied to problems having convex and continuous Pareto fronts. In contrast, in [107], the authors found out that the fitness inheritance can also be applied to problems having nonconvex and discontinuous Pareto fronts.

In [82], a metamodel was used to predict the class for the individuals. This class was the indication that individuals were either nondominated or dominated. This algorithm is known as Pareto-SVM where one class SVM is used for the classification of individuals in the decision space into nondominated and dominated. In addition, support vector regression is used to predict those individuals (after classification in the decision space) to a target value in the objective space. Aggregate surrogate metamodel terminology is used in the paper as two metamodels are used, one in the decision space and another in the objective space. Instead of binary classification in the decision space, all nondominated individuals mapped to the single value ρ with tolerance ϵ and all dominated individuals are mapped to $]-\infty, \rho-\epsilon[$. In this way, the individuals belonging

to $[\rho + \epsilon, +\infty[$ are in the unexplored region. In step 9, an updating criterion for the metamodel is inspired from EA PESA-II [23]. Firstly, individuals obtained after step 7 are added to the archive and duplicate individuals are removed. Then, the objective space is partitioned into a fixed number of equal sized hyperboxes and one individual or nondominated individual (if the box contains nondominated) is selected from each box randomly. These individuals are re-evaluated using the original functions. The size of the archive is fixed in this algorithm. In addition, to generate offspring in step 6, firstly, a fixed number of individuals (more than the population size) is generated using the variation operators. Then a fixed number of individuals is selected based on their distance to the current nondominated individuals in the decision space.

The proposed algorithm was tested on eight benchmark problems with two objectives and 10-30 decision variables. Hypervolume was used as the performance criterion for comparison of Pareto-SVM with $(\mu+\lambda)-S-$NSGA-II [35] and $\mu\times(1+\lambda)-$MO-CMA-ES [50]. Pareto-SVM obtained similar performance in fewer function evaluations.

In [83], Pareto-SVM was extended to a rank based aggregate surrogate model. In this algorithm, instead of two classes, various classes were obtained in the decision space based on the rank of individuals. In this way, nondominated individuals at the current generation are not constrained to the bounds. A pairwise dominance relation was used to classify individuals. For example, two possibilities exist, if an individual dominates another individual or is dominated by it. The case for nondominated individuals is mentioned as the future work. Other details are the same as in Pareto-SVM.

The proposed algorithm was tested on the same benchmark problems. It was compared with $(\mu+\lambda)-S-$NSGA-II, $\mu\times(1+\lambda)-$MO-CMA-ES and Pareto-SVM with hypervolume as the performance criterion. In comparison to Pareto-SVM, it performed similar and in comparison to other two algorithms, it performed better in fewer function evaluations.

In [99], instead of predicting objective functions, a metamodel was used to predict the distance to the current nondominated individuals in the decision space. Three metamodels, linear regression, support vector regression and multilayer perceptron were used independently. This algorithm is known as ASM-MOMA, where the metamodel is built for the distance to the nondominated individuals in step 5. In addition, a local search algorithm is used to improve the individuals obtained after step 7. To update the metamodel in step 9, nondominated individuals at the current generation are added to the archive in step 10. The size of the archive is fixed in this algorithm and extra individuals are removed from it randomly.

ASM-MOMA was tested on four benchmark problems (two objectives and 15 decision variables) and compared with NSGA-II and IBEA [141] with hypervolume as the performance criterion. The proposed algorithm obtained similar performance in fewer function evaluations than other algorithms while using linear regression as the metamodel.

In [100], ASM-MOMA was extended to higher-dimensional problems (in the objective space) with two different additional elements. Instead of using one global metamodel, multiple local metamodels were used. This algorithm is known as LAMM-MMA where the training data in the decision space is partitioned into different sets based on the distance of individuals to the current nondominated individuals. Multiple local metamodels are then built using this data in step 5. Other details are the same as in ASM-MOMA. This algorithm was tested on four benchmark problems with 5-15 objectives and 20 variables and compared with IBEA and ASM-MOMA with hypervolume as the performace criterion. LAMM-MMA performed better than IBEA and similar to ASM-MOMA in the same number of function evaluations.

In [9], a metamodel was used to predict the contribution to the hypervolume instead of the objective functions. This algorithm is known as NN-SS-IBEA where a metamodel (neural network) is built for the contribution to the hypervolume of individuals in step 5. After evaluating the new individuals in step 7, one individual having the maximum contribution to the hypervolume is re-evaluated in step 9 and added to the archive in step 10. The size of the archive is fixed in this algorithm and extra individuals are eliminated from the archive based on their hypervolume contribution.

The proposed algorithm was tested on 12 benchmark (2-3 objectives and 8-30 decision variables) and one real-world problem (two objectives and 11 decision variables). The algorithm was compared with MOEA/D-RBF [85] and IBEA [141]. The hypervolume was used as the comparison criterion. In case of benchmark problems, NN-SS-IBEA performed better in eight out of 12 problems with the same number of function evaluations. In case of the real-world problem, the proposed algorithm performed better than others for a given number of function evaluations.

A similar algorithm to [83] was proposed in [11], where nondominated individuals were also used during classification. In this algorithm, a metamodel is built for three classes while doing pairwise dominance comparison. Here, three possibilities exist, one solution dominates another, one solution is dominated by another or both solutions are nondominated. Ten different classification algorithms were used independently. The metamodel is updated in step 9 after a fixed number of generations with the previously evaluated individuals in step 7. The information on the size of the archive is not mentioned.

The proposed algorithm was tested on three benchmark problems (two objectives and 5-20 decision variables). However, this algorithm was not compared

with any other algorithm, but the results of ten classification algorithms were compared using the training time and the accuracy as the comparison criteria. The authors mentioned that SVM, classification trees, k-neural network and quadratic discriminant analysis were the most preferred among other metamodels as these metamodels found the knee region in the objective space.

In [114], a metamodel was used to predict the rank of individuals in the objective space. In this algorithm, the metamodel is used to create boundaries between fronts in the objective space. As shown in Figure 4, θ_1 and θ_2 represent the two boundaries created using individuals of three different fronts. The ranks of offspring individuals are then predicted with an indicator function mentioned in the paper with boundaries created (steps 5-7). For example, individuals that lie below θ_1 are of rank 1, individuals between θ_1 and θ_2 are of rank 2 and individuals above θ_2 are of rank 3. The first rank individuals are re-evaluated using the original functions in step 9 and added to the archive in step 10. The size of the archive is not fixed in this algorithm.

This algorithm was tested on 19 benchmark problems (the authors did not mention explicitly the number of objectives and decision variables) and compared with NSGA-II, SPEA2 [142] and MOEA/D [135]. Three performance criteria (generational distance, inverted generational distance and hypervolume [34]) were used to compare the proposed algorithm and EAs for the same number of function evaluations. Out of 19 problems, the proposed algorithm performed better than the other algorithms in 16, 14 and 12 problems in generational distance, inverted generational distance and hypervolume, respectively. In the rest of the problems, MOEA/D and NSGA-II performed better than the proposed algorithm.

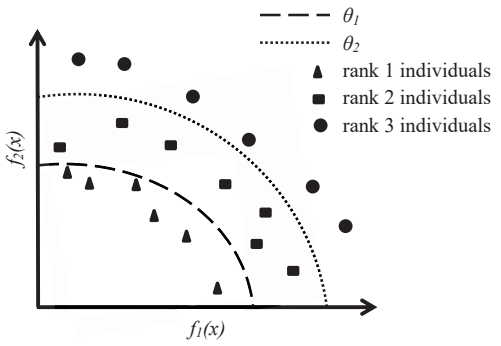


Figure 4: Assignment of ranks to individuals

In [124], a study was performed to compare four metamodelling techniques and three different scalarizing functions using two different approaches of approximation. Therefore, for each approximation, 12 different experiments were performed on different problems. A data set of prefixed size was used for training and validation. The error after the validation was used to compare different studies. As no algorithm was proposed in the current study, other steps of the function approximation were not applied. Four metamodels were based on Kernel ridge regression (KRR), Kriging, Generalized linear models and Tree-based regression. Three different scalarizing functions tested were based on Tchebychef (TCH), Boundary intersection (PBI) and weighted sum (WS). As mentioned, two different approaches of approximation were used, in the first one, the metamodel was used for the scalarizing funcion and in the second one, the metamodel was used to approximate the rank of individuals.

In the study, experiments were conducted on DTLZ suite with 4-10 objectives and 8-19 decision variables. In both types of approximation used, Kriging performed the best. In comparing different scalarizing functions, TCH performed the best in the first type of approximation and WS in the second type of approximation. Developing a algorithm was considered as a future work.

4 Discussion

In this section, we first summarize promising elements that we found in the algorithms proposed in the literature. Furthermore, we also discuss ways which can be used to design enhanced EAs to handle computationally expensive MOPs. Finally, we discuss the main issues we have observed in the algorithms in the literature with respect to using an approximation in an EA and the numerical settings used to test their efficacy.

4.1 Promising elements in the literature

We discuss here promising elements with respect to the choice of the metamodel to be used, building/updating the metamodels and using different types of approximations together to reduce the computation time. Furthermore, we discuss the potential of using hybrid algorithms with function approximation based algorithms to enhance the rate of convergence of EAs near Pareto optimal solutions. Also, one can use these elements while designing an algorithm for using an approximation with EA to reduce the computation time.

Often when metamodels are used, the type of metamodel to be used is a random choice. It is often difficult to know a priori the best metamodel to be used in the solution process. An ensemble of metamodels (see e.g. [79] and [118]) could be an easy fix, where the algorithm is not limited to one single metamodel. In [79], weights are assigned to the fitness values predicted by each metamodel and the weighted sum is then used to get the final fitness value. In [118], one metamodel having the highest accuracy is used for evaluating individuals of the EA population. Moreover, in [7], the authors present an improved way of using multiple metamodels that focus on different parts of the Pareto front. These metamodels are updated using some solutions

Table 2: algorithm reference, problems used, dimensions of the problems and metamodel used in different algorithms ref : reference, $problem$: multiobjective problem(s), $var/obj/cons$: number of decision variables, objectives and constraints, $metamodel$: metamodel used

ref	$problem$	$var/obj/cons$	$metamodel$
[7]	Aerodynamic shape optimization	12/2-3/0	NN
[9]	Benchmark	8-30/2-3/0	NN
	Aerodynamic shape optimization	11/2/0	NN
[11]	Banchmark	5-20/2/0	several classification algorithms
[15]	Benchmark	1-5/2/2	NN
	Design of vehicle door	5/2/2	NN
[19]	Benchmark	10/3-10/0	Kriging
	Free-radical polymerization	4/3/0	Kriging
[46]	Rigid frame design	10/2/0	SVR
[49]	Microchannel heat sink	3/2/0	PR, Kriging, NN
[56]	Fatigue strength assessment for ship	17/2/11	Kriging
[72]	Coastal aquifer management	8/2/0	NN
[76]	Benchmark	3-6/2/0	Kriging
	Cabinet problem	2/2/0	Kriging
	Gear train problem	4/2/0	Kriging
	Distillation column	4/2/4	Kriging
[78]	Benchmark	3-6/2/0	Kriging
	Cabinet problem	2/2/0	Kriging
	Gear train problem	4/2/0	Kriging
[79]	Benchmark	10-50/2	PR, Kriging, NN (ensemble)
[80]	Benchmark	2-3/2/2	PR
	Car front floor forming	3/2/1	PR
	Car sheet metal forming	3/2/0	PR
[81]	Benchmark	2-30/2/0	Kriging
	Stiffed panel problem	6/3/0	Kriging
[82]	Benchmark	10-30/2/0	SVR,SVM
[83]	Benchmark	10-30/2/0	SVR,SVM
[85]	Benchmark	8-30/2/0	NN (ensemble)
	Aerodynamic shape optimization	11/2/0	NN (ensemble)
[89]	Iron induration process	22/2/3	NN
[98]	Benchmark	10-60/2/0	NN
[99]	Benchmark	15/2/0	LR,SVR,NN
[100]	Benchmark	20/5-15/0	LR,SVR,NN
[101]	Benchmark	10-30/2/0	SVR
[102]	Benchmark	3-6/2-5/0	Kriging
[114]	Benchmark	-	SVM
[118]	Benchmark	10/2/2	PR,NN (ensemble)
[119]	Nowacki bean problem	2/2/5	Kriging
	Double folded stub microwave filter	3/2/7	Kriging
[122]	Benchmark	-/2/0	NN
	Automobile production planning	11/2/0	NN
[132]	Trauma system	-/2/2	PR
[136]	Benchmark	2-8/2-3/0	Kriging
[139]	Benchmark	5/2/0	Kriging
	Stiffed panel problem	6/2/0	Kriging

that were more and less accurate. Selection of these solutions enables the algorithm proposed in [7] to converge faster and explore the search space for promising candidate Pareto optimal solutions.

One more promising element observed in some of the algorithms in the literature is to use a combination of different approximations. In [122], fitness inheritance (a type of fitness approximation) is used with function approximation to reduce the computation time. Problem approximation is used with function approximation in [139] to decrease the computational complexity of the problem in question and the numbers of function evaluations. Moreover, the use of multilevel approaches [41, 65] which depend on the problem to be solved needs more attention from both researchers and practitioners. In these approaches, either different solvers were used to solve governing equations (e.g. Navier-Stokes equations) or different problems were solved at different levels. For more details about multilevel approaches, see [41].

In addition, metamodels have been used in the literature to predict the quality of solutions instead of the objective functions. For instance, in [9], the metamodel was used to predict the hypervolume contribution of the individuals. In [99, 100], the metamodel was used to predict the distance from the current nondominated individuals. Moreover, in [11, 82, 83], the metamodel was used to classify the individuals into different classes based on their dominance relations. In [114], the metamodel was used to predict rank of individuals by creating boundaries between individuals in the objective space. This way of using a metamodel other than function approximation could be affective as instead of actual objective functions, quality of individuals (e.g. using hypervolume or rank) is predicted.

Hybrid EAs with a combination of global and local searches are used to enhance the rate of convergence towards the Pareto front [51, 52, 116, 117]. However, such algorithms are not commonly used together with function approximation to handle computationally expensive MOPs. In the literature e.g. in [49, 79, 125], a local search is used with function approximation. This way of using local search algorithm with function approximation can inherit advantages of both hybrid algorithms and function approximation based algorithms, i.e. fast convergence and reducing the numbers of computationally expensive function evaluations.

4.2 Issues with the present algorithms

Let us discuss next the main issues we have observed in the algorithms in the literature related to using an approximation in an EA and the numerical settings used to test their efficacy. These main issues include: 1. type of test problems used (benchmark or real-world), 2. dimensions of the test problems both in objective and decision spaces considered, 3. scarce use of constrained problems for numerical testing, 4. neglecting the training time for fitting the metamodel used when reporting results, 5. less focus on the accuracy of the metamodel, 6. not well explored updating criterion of the metamodel, 7. not well structured ensemble of metamodels, and 8. less emphasis to algorithms that consider problem information when available. To augment the discussion on the issues enumerated above, we present in Table 2 an overview of the test problems considered in the literature with their dimensions both in objective and decision spaces and the type of metamodel used. Issues 1-3 and 4-8 address the shortcomings in numerical testing and solution algorithms used, respectively.

In Table 2, it can be clearly seen that both benchmark and real-world problems have been widely used by researchers to test their proposed EAs. Benchmark problems available in the literature are designed to be simple to implement, fast to evaluate, have known global optimal solutions, pose varied challenges to EAs such as several locally optimal solutions etc. Benchmark problems solved are not computationally expensive and used just to test the efficiency of a particular algorithm. For more details about the characteristics of benchmark problems, see [28, 21]. On the other hand, real-world problems either are actual problems considered in the industry or an emulation of them. Often, real-world problems are computationally expensive to evaluate, have uncertainties in their input and output variables, which in turn does not allow a thorough testing of an EA in practice using several real-world problems. Thus, it could be a good practice, if a numerical test setting involving several benchmark and a few real-world problems could be used to thoroughly test EAs. However, in the literature very few researchers have adopted this practice. One reason for this could be attributed to the lack of open availability of real-world problems to all researchers.

Real-world problems in industries often involve a large number of objectives, a large decision space and a large number of constraints. It can be seen in Table 2 that most of the problems considered in the literature are usually low dimensional both in decision and objective spaces. Additionally, only six algorithms referred in Table 2 consider constraints in the problems for testing in addition to bounds for the decision variables. Thus, significant attention is needed towards issues 2 and 3 and more constrained benchmark and real-world problems need to be included in the numerical test settings. The number of objectives considered does not usually exceed three and the maximum number of decision variables considered is under 60. In fact, only three algorithms were used on more than three objective functions, four algorithms were used for three objective functions and the rest of the algorithms were used for solving biobjective optimization problems.

Furthermore, it is worth noting the connection between the type of metamodels used and the number of decision variables associated with the problem. In one instance, neural network was used when the number of

decision variables was high (60). Next, in the decreasing order of popularity are algorithms using ensemble of metamodels, Kriging, support vector regression and polynomial approximation (including linear regression) for handling high dimensional (decision space) problems. What is missing in the literature is a study with each of the algorithms about the efficacy of using different metamodels when the dimensions of the decision space increases, especially in algorithms where the type of the metamodel to be used within the algorithm is not binding. Hence, further research towards dimensions and types of metamodels is needed.

The issues 4-7 concern the use of metamodels. The major concern in the literature is how to alleviate the computational cost of the MOP. However, the training time needed to build/update a metamodel is completely neglected when reporting the results. This is mainly based on the assumption by the researchers that the computation time needed to compute objective and constraint function values is significantly higher as compared to the building/updating time of the metamodel used. This assumption may not hold in all cases and, specifically, when a large amount of data is used to build/update the metamodel.

Most of the algorithms discussed in this survey used a fixed evolution control strategy, i.e. the accuracy of the metamodel is not considered as an important issue. In this survey, we defined the adaptive evolution control, where if the metamodel is not accurate, the original functions are used for evaluating individuals. A desired accuracy for a metamodel can be predefined (e.g. in terms of a parameter) or changed by the user during the solution process. The accuracy can be calculated by different statistical measurements such as root mean square error, coefficient of determination, analysis of variance (ANOVA) etc. [123]. As the accuracy of the metamodel is not considered in many algorithms in the literature and this is a valid future research direction.

The updating criterion of the metamodel can influence the quality of nondominated solutions. Considering the function approximation framework mentioned in Section 3, selected individuals from step 7 are re-evaluated with the original functions. The number and selection criterion of individuals to be used for re-evaluation from step 7 is not well explored in the literature, however, few algorithms considered both exploration and exploitation while updating the metamodel. We consider this as an important issue as selection of these individuals affects the updating time and accuracy of the metamodel, which may finally affect the quality of the set of nondominated solutions generated by the algorithm.

Although we consider an ensemble of metamodels to be a very promising element in the literature, yet, we consider it as an issue as well. In most of the algorithms proposed in the literature, only the accuracy of the metamodel is used as the criterion to select a particular metamodel. However, there can be other criteria that can be considered in addition to the accuracy of the metamodel, e.g. diversity among individuals, training time of the metamodel etc. Considering these criteria for structuring an ensemble of metamodel is a future research direction.

Finally, the information about the MOP such as a different computation time required for evaluating different objective and/or constraint functions must be considered by the EA. Here we suggest two types of problem information that can be considered: firstly, if any objective function is not computationally expensive, a metamodel need not be created for that function. However, many real-world problems involve black box functions and therefore, an explicit information about them can be difficult to obtain. Secondly, sometimes performing an experiment to calculate the value of an objective function may take less time or may give better results when compared to using a numerical model to evaluate the same. For example in [69], a closed-loop optimization was performed, where values of objective functions are obtained by performing real experiments and selection while crossover and mutation operations are performed by using an EA. For more details about closed-loop evolutionary multiobjective optimization, see [69]. Thus, both types of problem information can lead to significant savings in computation time.

5 Conclusions

In this paper, we have presented a survey of different algorithms to reduce the computation time while solving computationally expensive multiobjective optimization problems. Altogether, 45 algorithms were found in the literature published in years 2008-2016, which were classified based on the type of approximation used, i.e. problem, function or fitness. We found that function approximation or using surrogates is the most common approach for reducing the computation time. Different algorithms were summarized based on the steps of a unified function approximation framework involving an evolutionary algorithm. Additionally, six important challenges were identified, involving using the metamodel, updating the metamodel, training time, type of the metamodel to be used, when to update the metamodel and constraint handling for implementing the proposed approximation framework. Subsequently, the first three important challenges were used in the survey to highlight the differences among different algorithms.

A proper management of the metamodel makes an algorithm efficient to solve a computationally expensive problem. As different algorithms used different strategies to manage the metamodels and solved problems with different characteristics, it is difficult to generalize the efficiency of a method. However, the efficiency of an algorithm can be attributed by 1. number of func-

tion evaluations used 2. dimensions (both in objective and decision spaces) of problems solved 3. characteristics of the problems solved e.g. non-linearity in the decision space, multimodality, disconnected Pareto front. As metamodel based algorithms are generally developed for black-box problems where characteristics of the problems to be solved is not known a priori, therefore, one can measure the efficiency of an algorithm by its ability to provide meaningful solutions in least number of function evaluations. We also provided the dimensions and the number of function evaluations used to solve a particular problem.

In this survey, we summarized several algorithms based on the steps of a unified function approximation framework. We also compared these algorithms with respect to different criteria such as type of metamodel and evolutionary algorithm used, type of problem (benchmark or real-world) solved and evolution control. Some of the major findings were: 1. Kriging and neural networks were the most commonly used metamodels, 2. most of the algorithms were based on dominance based evolutionary algorithms, 3. most of the algorithms solved the problems with maximum three objectives, 4. most of the algorithms were not developed to handle constraints, 5. number of decision variables was also limited especially when using Kriging, 6. only few algorithms used ensemble of metamodels, 7. many algorithms were tested only on benchmark problems which were not at all computationally expensive

We discussed problem and fitness approximation based algorithms with respect to their potential towards reducing the computational complexity and the number of function evaluations. We also identified some promising elements and issues among algorithms in the literature. Promising elements involve using an ensemble of metamodels, hybrid algorithms with function approximation based algorithms and different types of approximations together. We plan to address several issues observed related to shortcomings in numerical testings and algorithms in our future research.

Acknowledgements
The research of Tinkle Chugh was funded by the COMAS Doctoral Program (from Computing and Mathematical Sciences at the University of Jyväskylä) and FiDiPro Project DeCoMo (funded by TEKES, The Finnish Funding Agency for Innovation) and the research of Dr. Karthik Sindhya was funded by by SIMPRO project funded by Tekes: Finnish Funding Agency for Innovation

References

[1] E. Aarts and J. Lenstra, editors. *Local Search in Combinatorial Optimization*. Princeton University Press, 2003.

[2] D. Ackley. *A Connectionist Machine for Genetic Hillclimbing*. Kluwer Academic Publishers, Boston, 1987.

[3] T. Akhtar and C. Shoemaker. Multi objective optimization of computationally expensive multimodal functions with RBF surrogates and multirule selection. *Journal of Global Optimization*, 64:17–32, 2015.

[4] N. Alexandrov, J. Jr., R. Lewis, and V. Torczon. A trust-region framework for managing the use of approximation models in optimization. *Structural Optimization*, 15:16–23, 1998.

[5] R. Allmendinger, M. Emmerich, J. Hakanen, Y. Jin, and E. Rigoni. Surrogate-assisted multicriteria optimization: Business case, complexities and prospective solutions. *Multi-Criteria Decision Analysis*, to appear, 2016.

[6] A. Arias-Montano, C. Coello, and E. Mezura-Montes. MODE-LD+SS: A novel differential evolution algorithm incorporating local dominance and scalar selection mechanisms for multiobjective optimization. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pages 1–8. IEEE, 2010.

[7] A. Arias-Montano, C. Coello, and E. Mezura-Montes. Multi-objective airfoil shape optimization using a multiple-surrogate approach. In *Proceedings of the IEEE Congress on Evolutonary Computation*, pages 1–8. IEEE, 2012.

[8] H. Aytug and S. Sayin. Using support vector machines to learn the efficient set in multiple objective discrete optimization. *European Journal of Operational Research*, 193:510–519, 2009.

[9] N. Azzouz, S. Bechikh, and L. Said. Steady state IBEA assisted by MLP neural networks for expensive multi-objective optimization problems. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 581–588. ACM, 2014.

[10] T. Bäck. *Evolutionary algorithms in Theory and Practice*. New York: Oxford University Press, 1996.

[11] S. Bandaru, A. Ng, and K. Deb. On the performance of classification algorithms for learning Pareto-dominance relations. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pages 1139–1146. IEEE, 2014.

[12] S. Bandyopadhyay, S. Saha, U. Maulik, and K. Deb. A simulated annealing-based multiobjective optimization algorithm: AMOSA. *IEEE Transactions on Evolutionary Computation*, 12:269–283, 2008.

[13] J. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms*. Kluwer, 1981.

[14] J. Branke and C. Schmidt. Faster convergence by means of fitness estimation. *Soft Computing*, 9:13–20, 2005.

[15] G. Chen, X. Han, G. Liu, C. Jiang, and Z. Zhao. An efficient multi-objective optimization method for black-box functions using sequential approximate technique. *Applied Soft Computing*, 12:14–27, 2012.

[16] J.-H. Chen, D. E. Goldberg, S.-Y. Ho, and K. Sastry. Fitness inheritance in multiobjective optimization. In W. B. Langdon, E. Cantú-Paz, K. E. Mathias, R. Roy, D. Davis, R. Poli, K. Balakrishnan, V. Honavar, G. Rudolph, J. Wegener, L. Bull, M. A. Potter, A. C. Schultz, J. F. Miller, E. K. Burke, and N. Jonoska, editors, *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 319–326. Morgan Kaufmann, 2002.

[17] R. Cheng, Y. Jin, M. Olhofer, and B. Sendhoff. A reference vector guided evolutionary algorithm for many-objective optimization. *IEEE Transactions on Evolutionary Computation*, 20:773–791, 2016.

[18] T. Chugh, N. Chakraborti, K. Sindhya, and Y. Jin. A data-driven surrogate-assisted evolutionary algorithm applied to a many-objective blast furnace optimization problem. *Material and Manufacturing Processes*, to appear, 2016.

[19] T. Chugh, Y. Jin, K. Miettinen, J. Hakanen, and K. Sindhya. A surrogate-assisted reference vector guided evolutionary algorithm for computationally expensive many-objective optimization. *IEEE Transactions on Evolutionary Computation*, 2016.

[20] T. Chugh, K. Sindhya, K. Miettinen, J. Hakanen, and Y. Jin. *On Constraint Handling in Surrogate-Assisted Evolutionary Many-Objective Optimization*, pages 214–224. Springer International Publishing, Cham, 2016.

[21] C. Coello, G. Lamont, and D. Veldhuizen. *Evolutionary Algorithms for Solving Multi-objective Problems*. Springer, New York, 2nd edition, 2007.

[22] C. Coello and G. Pulido. A micro multi-objective genetic algorithm for multi-objective optimizations. In E. Zitzler, L. Thiele, K. Deb, C. Coello, and D. Corne, editors, *Proceedings of the Evolutionary Multi-Criterion Optimization*, pages 126–140. Springer, Berlin, Heidelberg, 2001.

[23] D. Corne, N. Jerram, J. Knowles, and M. Oates. PESA-II: Region-based selection in evolutionary multiobjective optimization. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 283–290. Morgan Kaufmann, 2001.

[24] I. Couckuyt, D. Deschrijver, and T. Dhaene. Fast calculation of multiobjective probability of improvement and expected improvement criteria for Pareto optimization. *Journal of Global Optimization*, 60:575–594, 2014.

[25] C. Currin, M. Mitchell, M. Morris, and D. Ylvisaker. A Bayesian approach to the design and analysis of computer experiments. Technical report, Oak Ridge National Laboratory, 1998.

[26] A. L. Custodio, J. Madeira, A. Vaz, and L. N. Vicente. Direct multisearch for multiobjective optimization. *SIAM Journal on Optimization*, 21:1109–1140, 2011.

[27] R. Datta and R. Regis. A surrogate-assisted evolution strategy for constrained multi-objective optimization. *Expert Systems with Applications*, 57:270–284, 2016.

[28] K. Deb. *Multi-Objective Optimization using Evolutionary Algorithms*. Wiley, Chichester, 2001.

[29] K. Deb, K. Miettinen, and S. Chaudhuri. Toward an estimation of nadir objective vector using a hybrid of evolutionary and local search approaches. *IEEE Transactions on Evolutionary Computation*, 6:821–841, 2010.

[30] K. Deb and P. Nain. An evolutionary multiobjective adaptive meta-modelling procedure using artificial neural networks. In S. Yan, Y.-S. Ong, and Y. Jin, editors, *Proceedings of the Evolutionary Compitation in Dynamic and Uncertain Environments*, pages 297–322. Springer, 2007.

[31] K. Deb, A. Prarap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6:182–197, 2002.

[32] J. Dennis and V. Torczon. Managing approximation models in optimization. In N. Alexandrov and N. Hussaini, editors, *Proceedings of the Multidisciplinary Design Optimization: State-of-the-Art*, pages 330–347, 1995.

[33] E. Ducheyne, B. Baets, and R. Wulf. Is fitness inheritance useful for real-world applications? In C. Fonseca, P. Fleming, E. Zitzler, L. Thiele, and K. Deb, editors, *Proceedings of the Evolutionary Multi-Criterion Optimization*, pages 31–42. Springer, Berlin, Heidelberg, 2003.

[34] J. Durillo, A. Nebro, and E. Alba. The jmetal framework for multi-objective optimization: Design and architecture. In *Proceedings of the IEEE*

Congress on Evolutionary Computation, pages 1–8. IEEE, 2010.

[35] M. Emmerich, N. Beume, and B. Naujoks. An EMO algorithm using the hypervolume measure as selection criterion. In C. Coello, A. Aguirre, and E. Zitzler, editors, *Proceedings of the Evolutionary Multi-Criterion Optimization*, pages 62–76. Springer, Berlin, Heidelberg, 2005.

[36] M. Emmerich, A. Deutz, and J. Klinkenberg. Hypervolume-based expected improvement: Monotonicity properties and exact computation. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pages 2147–2154. IEEE, 2011.

[37] M. Emmerich, K. Giannakoglou, and B. Naujoks. Single- and multiobjective evolutionary optimization assisted by Gaussian random field metamodels. *IEEE Transactions on Evolutionary Computation*, 10:421–439, 2006.

[38] M. Emmerich, A. Giotis, M. Özdemir, T. Bäck, and K. Giannakoglou. Metamodel-assisted evolution strategies. In J. Merelo-Guervós, P. Adamidis, H.-G. Beyer, H.-P. Schwefel, and J.-L. F.-V. nas, editors, *Proceedings of the Parallel Problem Solving from Nature-PPSN VII*, pages 361–370. Springer, Berlin, Heidelberg, 2002.

[39] A. Forrester and A. Keane. Recent advances in surrogate-based optimization. *Progress in Aerospace Sciences*, 45:50–79, 2009.

[40] S. E. Gano, J. E. Renaud, J. Martin, and T. Simpson. Update strategies for Kriging models used in variable fidelity optimization. *Structural and Multidisciplinary Optimization*, 32:287–298, 2006.

[41] K. Giannakoglou and I. Kampolis. Multilevel optimization algorithms based on metamodel- and fitness inheritance-asisted evolutionary algorithms. In Y. Tenne and C.-K. Goh, editors, *Computational Intelligence in Expensive Optimization Problems*, pages 61–84. Springer, Berlin, Heidelberg, 2010.

[42] D. Gorissen, I. Couckuyt, P. Demeester, T. Dhaene, and K. Crombecq. A surrogate modelling and adaptive sampling toolbox for computer based design. *Journal of Machine Learning Research*, 11:2051–2055, 2010.

[43] L. Gräning, Y. Jin, and B. Sendhoff. Individual-based management of meta-models for evolutionary optimization with application to three-dimensional blade optimization. In S. Yang, Y.-S. Ong, and Y. Jin, editors, *Evolutionary Computation in Dynamic and Uncertain Environments*, pages 225–250. Springer, Berlin, Heidelberg, 2007.

[44] M. Hansen and A. Jaskiewicz. Evaluating the quality of approximation to the non-dominated set. Technical report, Technical University of Denmark, 1998.

[45] N. Hansen and A. Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9:159–195, 2001.

[46] M. Herrera, A. Guglielmetti, M. Xiao, and R. Coelho. Metamodel-assisted optimization based on multiple kernel regression for mixed variables. *Structural and Multidisciplinary Optimization*, 49:979–991, 2014.

[47] D. Horn, T. Wagner, D. Biermann, C. Weihs, and B. Bischl. Model-based multi-objective optimization: Taxonomy, multi-point proposal, toolbox and benchmark. In A. Gasper-Cunha, C. H. Antunes, and C. Coello, editors, *Evolutionary Multi-criterion Optimization*, pages 64–78. Springer, 2015.

[48] G. Huang, Q. Zhu, and C. Siew. Extreme learning machine: A new learning scheme of feedforward neural networks. In *Proceedings of the International Joint Conference on Neural Networks*, pages 985–990. IEEE, 2004.

[49] A. Husain and K.-Y. Kim. Enhanced multiobjective optimization of a microchannel heat sink through evolutionary algorithm coupled with multiple surrogate models. *Applied Thermal Engineering*, 30:1683–1691, 2010.

[50] C. Igel, N. Hansen, and S. Roth. Covariance matrix adaptation for multi-objective optimization. *Evolutionary Computation*, 15:1–28, 2007.

[51] H. Ishibuchi, Y. Hitotsuyanagi, N. Tsukamoto, and Y. Nojima. Use of heuristic local search for single-objective optimization in multiobjective memetic algorithms. In G. Rudolph, T. Jansen, S. Lucas, C. Poloni, and N. Beume, editors, *Proceedings of the Parallel Problem Solving from Nature-PPSN X*, pages 743–752. Springer, Berlin, Heidelberg, 2008.

[52] H. Ishibuchi, Y. Hitotsuyanagi, Y. Wakamatsu, and Y. Nojima. How to choose solutions for local search in multiobjective combinatorial memetic algorithms. In R. Schaefer, C. Cotta, J. Kolodziej, and G. Rudolph, editors, *Proceedings of the Parallel Problem Solving from Nature-PPSN XI*, pages 516–525. Springer, Berlin, Heidelberg, 2010.

[53] A. Jain and R. Dubes. *Algorithms for Clustering Data*. Prentice-Hall, 1998.

[54] A. Jain, M. Murty, and P. Flynn. Data clustering: A review. *ACM Computing Surveys*, 31:264–323, 1999.

[55] H. Jain and K. Deb. An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part II: Handling constraints and extending to an adaptive approach. *IEEE Transactions on Evolutionary Computation*, 18:602–622, 2014.

[56] B.-S. Jang, D.-E. Ko, Y.-S. Suh, and Y.-S. Yang. Adaptive approximation in multi-objective optimization for full stochastic fatigue design problem. *Marine Structures*, 22:610–632, 2009.

[57] S. Jeong and S. Obayashi. Efficient global optimization (EGO) for multi-objective problem and data mining. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pages 2138–2145. IEEE, 2005.

[58] Y. Jin. A comprehensive survey of fitness approximation in evolutionary computation. *Soft Computing*, 9:3–12, 2005.

[59] Y. Jin. Surrogate-assisted evolutionary computation: Recent advances and future challenges. *Swarm and Evolutionary Computation*, 1:61–70, 2011.

[60] Y. Jin, M. Olhofer, and B. Sendhoff. On evolutionary optimization with approximate fitness functions. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 786–793. Morgan Kaufmann, 2000.

[61] Y. Jin, M. Olhofer, and B. Sendhoff. A framework for evolutionary optimization with approximate fitness functions. *IEEE Transactions on Evolutionary Computation*, 6:481–494, 2002.

[62] Y. Jin and B. Sendhoff. Reducing fitness evaluations using clustering techniques and neural network ensembles. In K. Deb, editor, *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 688–699. Springer, Berlin, Heidelberg, 2004.

[63] M. Johnson, L. Moore, and D. Ylvisaker. Minimax and maximin distance designs. *Journal of Statistical Planning and Inference*, 26:131–148, 1990.

[64] D. Jones, M. Schonlau, and W. Welch. Efficient global optimization of expensive black-box functions. *Journal of Global Optimization*, 13:455–492, 1998.

[65] I. Kampolis and K. Giannakoglou. A multilevel approach to single and multiobjective aerodynamic optimization. *Computer Methods in Applied Mechanics and Engineering*, 197:2963–2975, 2008.

[66] A. Keane. Statistical improvement criteria for use in multiobjective design optimization. *AIAA Journal*, 44:879–891, 2006.

[67] H.-S. Kim and S.-B. Cho. An efficient genetic algorithm with less fitness evaluation by clustering. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pages 887–894. IEEE, 2001.

[68] J. Knowles. ParEGO: a hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems. *IEEE Transactions on Evolutionary Computation*, 10:50–66, 2006.

[69] J. Knowles. Closed-loop evolutionary multiobjective optimization. *IEEE Computational Intelligence Magazine*, 4:77–91, 2009.

[70] J. Knowles and H. Nakayama. Meta-modeling in multiobjective optimization. In J. Branke, K. Deb, K. Miettinen, and R. Slowinski, editors, *Multiobjective Optimization: Interactive and Evolutionary Approaches*, pages 245–284. Springer, Berlin, Heidelberg, 2008.

[71] G. Kourakos and A. Mantoglou. Pumping optimization of coastal aquifers based on evolutionary algorithms and surrogate modular neural network models. *Advances in Water Resources*, 32:507–521, 2009.

[72] G. Kourakos and A. Mantoglou. Development of a multi-objective optimization algorithm using surrogate models for coastal aquifer management. *Journal of Hydrology*, 479:13–23, 2013.

[73] V. Lattarulo, P. Seshadri, and G. Parks. Optimization of a supersonic airfoil using the multi-objective alliance algorithm. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 1333–1340. ACM, 2013.

[74] D. Lee, L. Gonzalez, J. Periaux, and K. Srinivas. Robust design optimisation using multi-objective evolutionary algorithms. *Computers & Fluids*, 37:565–583, 2008.

[75] S. Lee, P. Almon, W. Fink, A. Petropoulos, and R. Terrile. Comparison of multi-objective genetic algorithms in optimizing q-law low-thrust orbit transfers. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 25–29. ACM, 2005.

[76] G. Li, M. Li, S. Azarm, S. Hashimi, T. Ameri, and N. Qasas. Improving multi-objective genetic algorithm with adaptive design of expriments and online metamdeling. *Structural and Multidisciplinary Optimization*, 37:447–461, 2009.

[77] H. Li and Q. Zhang. Multiobjective optimization problems with complicated Pareto sets, MOEA/D and NSGA-II. *IEEE Transactions on Evolutionary Computation*, 12:284–302, 2009.

[78] M. Li, G. Li, and S. Azarm. A kriging metamodel assisted multi-objective genetic algorithm for design optimization. *Journal of Mechanical Design*, 130:1–10, 2008.

[79] D. Lim and Y. Jin. Generalizing surrogate-assisted evolutionary computation. *IEEE Transactions on Evolutionary Computation*, 14:329–354, 2010.

[80] G. Liu, X. Han, and C. Jiang. A novel multi-objective optimization method based on an approximation model management technique. *Computer Methods in Applied Mechanics and Engineering*, 197:2719–2731, 2008.

[81] Y. Liu and M. Collette. Improving surrogate-assisted variable fidelity multi-objective optimization using a clustering algorithm. *Applied Soft Computing*, 24:482–493, 2014.

[82] I. Loshchilov, M. Schoenauer, and M. Sebag. A mono surrogate for multiobjective optimization. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 471–478. ACM, 2009.

[83] I. Loshchilov, M. Schoenauer, and M. Sebag. Dominance-based Pareto-surrogate for multi-objective optimization. In K. Deb, A. Bhattacharya, N. Chakroborty, S. Das, J. Dutta, S. Gupta, A. Jain, V. Aggarwal, J. Branke, S. Louis, and K. Tan, editors, *Proceedings of the Simulated Evolution and Learning*, pages 230–239. Springer, Berlin, Heidelberg, 2010.

[84] C. Luo, K. Shimoyama, and S. Obayashi. Kriging model based many-objective optimization with efficient calculation of expected hypervolume improvement. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pages 1187–1194. IEEE, 2014.

[85] S. Martinez and C. Coello. MOEA/D assisted by RBF networks for expensive multi-objective optimization problems. In C. Blum, editor, *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 1405–1412. ACM, NY, 2013.

[86] M. Mckay, R. Beckman, and W. Conover. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 42:55–61, 2000.

[87] T. Mengistu and W. Ghaly. Aerodynamic optimization of turbomachinery blades using evolutionary methods and ANN-based surrogate models. *Optimization and Engineering*, 9:239–255, 2008.

[88] K. Miettinen. *Nonlinear multiobjective optimization*. Kluwer, Boston, MA, 1999.

[89] K. Mitra and S. Majumder. Successive approximate model based multi-objective optimization for an industrial straight grate iron ore induration process using evolutionary algorithm. *Chemical Engineering Science*, 66:3471–3481, 2011.

[90] M. Mlakar, D. Petelin, T. Tusar, and B. Filipic. GP-DEMO: differential evolution with multiobjective optimization based on gaussian process models. *European Journal of Operational Research*, 243:347–361, 2015.

[91] A. Mogilicharla, T. Chugh, S. Majumder, and K. Mitra. Multi-objective optimization of bulk vinyl acetate polymerization with branching. *Materials and Manufacturing Processes*, 29:210–217, 2014.

[92] P. Nain and K. Deb. A multi-objective optimization procedure with successive approximate models. Technical Report 2005002, KanGAL, Indian Institute of Technology Kanpur, India, 2005.

[93] H. Nakayama, K. Inoue, and Y. Yoshimori. Approximate optimization using computaional intelligence and its application to reinforcement of cable-stayed bridges. In *Proceedings of the Integrated Intelligent Systems for Engineering Design*, pages 289–304. IOS press, 2006.

[94] A. Oyama, Y. Okabe, K. Shimoyama, and K. Fujii. Aerodynamic multiobjective design exploration of a flapping airfoil using a navier-stokes solver. *Journal of Aerospace Computing, Information, and Communication*, 6:256–270, 2009.

[95] P. Palar, T. Tsuchiya, and G. Parks. A comparative study of local search within a surrogate-assisted multi-objective memetic algorithm framework for expensive problems. *Applied Soft Computing*, 43:1–19, 2016.

[96] P. S. Palar, T. Tsuchiya, and G. Parks. Comparison of scalarization functions within a local surrogate assisted multi-objective memetic algorithm framework for expensive problems. In *IEEE Congress on Evolutionary Computation (CEC)*, pages 862–869, 2015.

[97] L. Pavelski, M. Delgado, C. Almeida, R. Gonaslves, and S. Venske. Extreme learning surrogate models in multi-objective optimization based on decomposition. *Neurocomputing*, 180:55–67, 2016.

[98] L. Pavelski, M. Delgado, C. Almeida, R. Goncalves, and S. Venske. ELMOEA/D-DE: Extreme learning surrogate models in multi-objective optimization based on decomposition and differential evolution. In *Proceedings of the Brazilian Conference on Intelligent Systems*, pages 318–323. IEEE, 2014.

[99] M. Pilát and R. Neruda. ASM-MOMA: Multiobjective memetic algorithm with aggregate surrogate model. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pages 1202–1208. IEEE, 2011.

[100] M. Pilát and R. Neruda. Improving many-objective optimizers with aggreagte meta-models. In *Proceedings of the 11th International Conference on Hybrid Intelligent Systems*, pages 555–560. IEEE, 2011.

[101] M. Pilát and R. Neruda. Hypervolume-based local seacrh in multi-objective evolutionary optimization. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 637–644. ACM, 2014.

[102] W. Ponweiser, T. Wagner, D. Biermann, and M. Vincze. Multiobjective optimization on a limited budget of evaluations using model-assisted S-metric selection. In *Proceedings of the Parallel Problem Solving from Nature-PPSN X*, pages 784–794. Springer, Berlin, Heidelberg, 2008.

[103] W. Ponweiser, T. Wagner, and M. Vincze. Clustered multiple generalized expected improvement: A novel infill sampling criterion for surrogate models. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pages 3515–3522. IEEE, 2008.

[104] S. Qasem, S. Shamsuddin, S. Hashim, M. Darus, and E.A.-Shammari. Memetic multiobjective particle swarm optimization-based radial basis function network for classification problems. *Information Sciences*, 239:165 – 190, 2013.

[105] T. Ray, H. Singh, A. Isaacs, and W. Smith. Infeasibility driven evolutionary algorithm for constrained optimization. In E. Mezura-Montes, editor, *Constraint-Handling in Evolutionary Optimization*, pages 145–165. Springer, Berlin, Heidelberg, 2009.

[106] R. Regis. Multi-objective constrained black-box optimization using radial basis function surrogates. *Journal of Computational Science*, 16:140–155, 2016.

[107] M. Reyes-Sierra and C. Coello. A study of fitness inheritance and approximation techniques for multi-objective particle swarm optimization. In *Proceedings of the EEE Congress on Evolutionary Computation*, pages 65–72. IEEE, 2005.

[108] T. Robic and B. Filipic. DEMO: Differential evolution for multiobjective optimization. In *Proceedings of Evolutionary Multi-criterion Optimization*, pages 520–533. Springer, 2005.

[109] P. Roy and K.Deb. High dimensional model representation for solving expensive multi-objective optimization problems. Technical Report COIN Report Number 2016012, Michigan State University, 2016.

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

[111] L. Santana-Quintero, A. M. no, and C. Coello. A review of techniques for handling expensive functions in evolutionary multi-objective optimization. In Y. Tenne and C.-K. Goh, editors, *Computational Intelligence in Expensive Optimization Problems*, pages 29–59. Springer, Berlin, Heidelberg, 2010.

[112] M. Sasena, P. Papalambros, and P. Goovaerts. Exploration of metamodeling sampling criteria for constrained global optimization. *Engineering Optimization*, 34:263–278, 2002.

[113] K. Sastry, D. E. Goldberg, and M. Pelikan. Don't evaluate, inherit. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 551–558. Morgan Kaufmann, 2001.

[114] C.-W. Seah, Y.-S. Ong, I. W. Tsang, and S. Jiang. Pareto rank learning in multi-objective evolutionary algorithms. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pages 1–8. IEEE, 2012.

[115] K. Shimoyama, K. Sato, S. Jeong, and S. Obayashi. Updating kriging surrogate models based on the hypervolume indicator in multi-objectie optimization. *Journal of Mechanical Design*, 135:1–7, 2013.

[116] K. Sindhya, K. Deb, and K. Miettinen. Improving convergence of evolutionary multi-objective optimization with local search: a concurrent-hybrid algorithm. *Natural Computing*, 10:1407–1430, 2011.

[117] K. Sindhya, K. Miettinen, and K. Deb. A hybrid framework for evolutionary multi-objective optimization. *IEEE Transactions on Evolutionary Computation,*, 17:495–511, 2013.

[118] H. Singh, T. Ray, and W. Smith. Surrogate assisted simulated annealing (SASA) for constrained multi-objective optimization. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pages 1–8. IEEE, 2010.

[119] P. Singh, I. Couckuyt, F. Ferranti, and T. Dhaene. A constrained multi-objective surrogate-based optimization algorithm. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pages 3080–3087. IEEE, 2014.

[120] R. Smith, B. Dike, and S. Stegmann. Fitness inheritance in genetic algorithms. In *Proceedings of the ACM Symposium on Applied Computing*, pages 345–350. ACM, 1995.

[121] I. Sobol. Global sensitivity indices for nonlinear mathematical models and their monte carlo estimates. *Mathematics and Computers in Simulation*, 55:271–280, 2001.

[122] A. Syberfeldt, H. Grimm, A. Ng, and R. John. A parallel surrogate-assisted multi-objective evolutionary algorithm for computationally expensive optimization problems. In *Proceedings of the IEEE World Congress on Computational Intelligence*, pages 3177–3184. IEEE, 2008.

[123] Y. Tenne and S. Armfield. Metamodel accuracy assessment in evolutionary optimization. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pages 1505–1512. IEEE, 2008.

[124] G. Toscano and K. Deb. Study of the approximation of the fitness landscape and the ranking process of scalarizing functions for many-objective problems. Technical Report COIN Report Number 2016018, Michigan State University, 2016.

[125] A. Turco. Metahybrid: Combining metamodels and gradient-based techniques in a hybrid multiobjective genetic algorithm. In C. Coello, editor, *Proceedings of the Learning and Intelligent Optimization*, pages 293–307. Springer, Berlin, Heidelberg, 2011.

[126] H. Ulmer, F. Streichert, and A. Zell. Evolution strategies assisted by gaussian processes with improved preselection criterion. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pages 692–699. IEEE, 2003.

[127] D. Veldhuizen. *Multiobjective evolutionary algorithms: Classifications, analyses, and new innovations*. PhD thesis, Graduate School of Engineering of the Air Force Institute of Technology Air University, Dayton, 1999.

[128] D. Veldhuizen and G. Lamont. Evolutionary computation and convergence to a Pareto front. In *Proceedings of the Genetic Programming*, pages 221–228. Morgan Kaufmann, 1998.

[129] J. Vrgut and B. Robinson. Improved evolutionary optimization from genetically adaptive multimethod search. In *Proceedings of the National Academy of Sciences of the United States of America*, volume 104, pages 708–711, 2007.

[130] T. Wagner, M. Emmerich, A. Deutz, and W. Ponweiser. On expected-improvement criteria for model-based multi-objective optimization. In R. Schaefer, C. Cotta, J. Kolodziej, and G. Rudolph, editors, *Proceedings of the Parallel Problem Solving from Nature-PPSN XI*, pages 718–727. Springer, Berlin, Heidelberg, 2010.

[131] G. Wang. Adaptive response surface method using inherited latin hypercube design points. *Journal of Mechanical Design*, 125:210–220, 2003.

[132] H. Wang, Y. Jin, and J. Jansen. Data-driven surrogate-assisted multi-objective evolutionary optimization of a trauma system. *IEEE Transactions on Evolutionary Computation*, 20:939–952, 2016.

[133] B. Yang, Y.-S. Yeun, and W.-S. Managing approximation models in multiobjective optimization. *Structural and Multidisciplinary Optimization*, 24:141–156, 2002.

[134] R. Yuan and B. Guangchen. Comparison of neural network and kriging method for creating simulation-optimization metamodels. In B. Yang, W. Zhu, Y. Dai, L. T. Yang, and J. Ma, editors, *Proceedings of the 8th IEEE International Symposium on Dependable, Autonomic and Secure Computing*, pages 815–821. IEEE, 2009.

[135] Q. Zhang and H. Li. MOEA/D: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, 11:712–731, 2007.

[136] Q. Zhang, W. Liu, E. Tsang, and B. Virginas. Expensive multiobjective optimization by MOEA/D with gaussian process model. *IEEE Transactions on Evolutionary Computation*, 14:456–474, 2010.

[137] Q. Zhang, A. Zhau, S. Zhao, P. Suganthan, W. Liu, and S. Tiwari. Multiobjective optimization test instances for the CEC 2009 special session and competition. Technical Report CES-487, University of Essex/ Nanyang Technological University, Essex, U.K./Singapore, 2009.

[138] Y. Zheng, B. Julstrom, and W. Cheng. Design of vector quantization codebooks using a genetic algorithm. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pages 525–529. IEEE, 1997.

[139] J. Zhu, Y.-J. Wang, and M. Collette. A multiobjective variable-fidelity optimization method for genetic algorithms. *Engineering Optimization*, 46:521–542, 2013.

[140] E. Zitzler. *Evolutionary algorithms for multiobjective optimization: methods and applications*. PhD thesis, Swiss Federal Institute of Technology Zurich, 1999.

[141] E. Zitzler and S. Kunzli. Indicator-based selection in multiobjective search. In X. Yao, E. Burke, J. Lozano, J. Smith, J. Merelo-Guervós, J. Bullinaria, J. Rowe, P. Tiňo, and H.-P. S. A. Kabán, editors, *Proceedings of the Parallel Problem Solving from Nature-PPSN VIII*, pages 832–842. Springer, Berlin, Heidelberg, 2004.

[142] E. Zitzler, M. Laumanns, and L. Thiele. SPEA2: Improving the strength Pareto evolutionary algorithm for multiobjective optimization. In K. Giannakoglou, editor, *Proceedings of the Evolutionary Methods for Design, Optimisation and Control with Application to Industrial Problems (EUROGEN 2001)*, pages 95–100. CIMNE, 2002.

[143] E. Zitzler and L. Thiele. Multiobjective optimization using evolutionary algorithms-a comparative case study. In A. Eiben, editor, *Proceedings of the Parallel Problem Solving from Nature-PPSN V*, pages 292–301. Springer Heidelberg, 1998.

[144] E. Zitzler, L. Thiele, M. Laumanns, C. Fonseca, and V. Fonseca. Performance assessment of multiobjective optimizers: An analysis and review. *IEEE Transactions on Evolutionary Computation*, 8:117–132, 2003.

PII

A SURROGATE-ASSISTED REFERENCE VECTOR GUIDED EVOLUTIONARY ALGORITHM FOR COMPUTATIONALLY EXPENSIVE MANY-OBJECTIVE OPTIMIZATION

by

Tinkle Chugh, Yaochu Jin, Kaisa Miettinen, Jussi Hakanen, Karthik Sindhya

IEEE Transactions on Evolutionary Computation, to appear,
doi:10.1109/TEVC.2016.262230

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2016.2622301, IEEE Transactions on Evolutionary Computation

A Surrogate-assisted Reference Vector Guided Evolutionary Algorithm for Computationally Expensive Many-objective Optimization

Tinkle Chugh, Yaochu Jin, *Fellow, IEEE*, Kaisa Miettinen, Jussi Hakanen, Karthik Sindhya

***Abstract*—We propose a surrogate-assisted reference vector guided evolutionary algorithm for computationally expensive optimization problems with more than three objectives. The proposed algorithm is based on a recently developed evolutionary algorithm for many-objective optimization that relies on a set of adaptive reference vectors for selection. The proposed surrogate-assisted evolutionary algorithm uses Kriging to approximate each objective function to reduce the computational cost. In managing the Kriging models, the algorithm focuses on the balance of diversity and convergence by making use of the uncertainty information in the approximated objective values given by the Kriging models, the distribution of the reference vectors as well as the location of the individuals. In addition, we design a strategy for choosing data for training the Kriging model to limit the computation time without impairing the approximation accuracy. Empirical results on comparing the new algorithm with the state-of-the-art surrogate-assisted evolutionary algorithms on a number of benchmark problems demonstrate the competitiveness of the proposed algorithm.**

***Index Terms*—multiobjective optimization, reference vectors, surrogate-assisted evolutionary algorithms, model management, Kriging, computational cost**

I. Introduction

Many industrial optimization problems have multiple objectives to be optimized and these objectives are typically conflicting in nature, i.e. improvement in one objective will lead to deterioration of at least one of the other objectives. Such problems are known as multiobjective optimization problems (MOPs). In this paper, we consider MOPs in the following form :

$$\begin{aligned} &\text{minimize } \{f_1(x), \ldots, f_k(x)\} \\ &\text{subject to } x \in S \end{aligned} \tag{1}$$

with $k(\geq 2)$ objective functions $f_i(x)$: $S \rightarrow \Re$. The vector of objective function values is denoted by $f(x) = (f_1(x), \ldots, f_k(x))^T$. The (nonempty) feasible space S is a subset of the decision space $\Re^n$ and consists of decision vectors $x = (x_1, \ldots, x_n)^T$ that satisfy all the constraints. The image of the feasible region S in the objective space $\Re^k$ is called the feasible objective set denoted by Z. The elements of Z are called feasible objective vectors denoted by $f(x)$ or $z = (z_1, \ldots, z_k)^T$, where $z_i = f_i(x), i = 1, \ldots, k$, are the objective function values. As the objectives are conflicting, there typically does not exist a single optimal solution, but multiple so-called Pareto optimal solutions. The set of all optimal solutions in the objective space is called the Pareto front and in the decision space the Pareto set.

A large number of optimization methods have been reported in the literature. These methods can be classified into two main different fields, namely, multiple criteria decision making (MCDM) [33] and evolutionary multiobjective optimization (EMO) [10], [13]. Methods either find a representative set of Pareto optimal solutions or the most preferred solution for a decision maker and they differ in the way the solutions are obtained. For instance, in the MCDM community, an MOP is often scalarized into a single objective optimization problem. Moreover, a decision maker is usually involved in the solution process to identify preferred solutions. On the other hand, EMO algorithms work with a population of candidate solutions and often aim to find a set of solutions representing the whole Pareto front. The decision maker is involved usually after a set of Pareto optimal solution is found [42].

Evolutionary algorithms (EAs) have become popular in past decades due to several advantages. For example, they have the ability to handle different kinds of decision variables e.g. binary, integer, real or mixed and they do not assume any convexity or differentiability of objective functions and/or constraints involved. Despite of these advantages, EAs are often criticized because of slow convergence and a large number of function evaluations needed before an acceptable solution can be found. For instance, in aerodynamic optimization, one function evaluation involving computational fluid dynamics simulations may take substantial amount of time and it will become computationally prohibitive to use an EA to solve aerodynamic optimization problems.

One of the popular approaches to reduce computation time in evolutionary optimization is to introduce approximations, especially function approximation. Computational models for functional approximation are often known as surrogates and EAs using objective values estimated by surrogates are often referred to as surrogate-assisted evolutionary algorithm (SAEAs). A surrogate is also known as a metamodel in the literature, which can in part replace the computationally expensive objective functions. For more details about SAEAs, see [9], [24], [43].

Although numerous algorithms have been proposed in using surrogates in an EA, many challenges remain. One is the choice of the surrogate, as different types of surrogate techniques exist in the literature, e.g. Kriging, neural networks, support vector regression and polynomial approximation. Nevertheless, there is no simple rule for choosing the right type

This work was supported by the FiDiPro project DeCoMo funded by the Finnish Funding Agency for Innovation (TEKES). *(Corresponding author: Yaochu Jin)*

[1]Tinkle Chugh, Yaochu Jin, Kaisa Miettinen, Jussi Hakanen and Karthik Sindhya are with the Faculty of Information Technology, University of Jyvaskyla, Finland. Yaochu Jin is also with the Department of Computer Science, University of Surrey, Guildford, United Kingdom. Email: [first-name.last-name]@jyu.fi, yaochu.jin@surrey.ac.uk.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2016.2622301, IEEE Transactions on Evolutionary Computation

of surrogates for approximating the given computationally expensive objective or constraint functions. A second challenge is how to use a surrogate, i.e. what to approximate using the surrogate. The most conventional way is to approximate the objective or constraint functions. In addition, a surrogate can be used to estimate the rank of individuals [31], or some other quality measure, e.g. distance to the nondominated solutions [38], [39], [40], or hypervolume [2]. The third challenge is the computational cost for constructing the surrogate, which is often neglected in SAEAs. In practical, training a surrogate may take a substantial amount of time, and the main aim of reducing computation time will be jeopardized. The fourth challenge is how to update the surrogate i.e. how to choose individuals in the current population to be re-evaluated using the original functions. Different ways exist in the literature for selecting the individuals, e.g. selecting a set of best solutions [25] or nondominated solutions [17] according to the surrogate and selecting a set of representative solutions [27]. If the Kriging model is used, one can select solutions that maximize the expected improvement [47], the probability of improvement [12] and hypervolume improvement [18]. Selection of individuals to be re-evaluated is also called updating criterion or infilling criterion. The fifth challenge is to determine when the surrogate needs to be updated. For instance, it may be possible that a surrogate is accurate enough and does not need to be updated even if new training samples are available.

In SAEAs, which individuals are to be re-evaluated using the original objective functions, how to update the surrogate and when to update the surrogate are referred to as model management, which is also known as evolution control [26]. In [26], two methods were mentioned for managing the surrogate, i.e., fixed evolution control and adaptive evolution control. In fixed evolution control, updating the surrogate is based on a prefixed criterion, while in adaptive evolution control, a surrogate is updated based on its performance.

As pointed out in [9], little work has been reported on using SAEAs for solving computationally expensive problems having more than three objectives. During the years 2008-2015, only three algorithms [6], [38], [41] have been tested on multi-objective benchmark problems with more than three objectives. While many industrial problems, e.g., optimization of the controller of a hybrid car [36], involve more than three computationally expensive objectives, surrogate-assisted evolutionary optimization of many-objective problems has not attracted much attention in the evolutionary computation community and SAEAs developed so far cannot be directly extended to many-objective optimization. Therefore, our work is an effort to fill this gap.

Apart from the challenges resulting from the large number of objectives, it is notoriously difficult to achieve high-quality surrogates for large scale optimization problems due to the curse of dimensionality. For this reason, the number of decision variables SAEAs have handled is by far up to 50. According to a recent survey [9], only seven SAEAs have been tested on optimization problems with more than 20 decision variables and six of them were benchmarks. Note that SAEAs using neural networks as the surrogate were tested on more than 20 variables, while SAEAs using Kriging models as the surrogate have been used to solve problems with up to eight decision variables. This can be attributed to the fact that the computational time for training the Kriging model will become too long when the number of training samples increases [35].

This paper focuses on developing an efficient SAEA for solving computationally expensive many-objective optimization problems. One of the major reasons limiting the applicability of existing algorithms to many-objective optimization is the lack of an efficient surrogate management method suited for the evolutionary algorithm used. In SAEAs when managing the surrogates, individuals should be selected by taking into account of both convergence and diversity. To select such individuals, surrogates need to be seamlessly embedded into the evolutionary algorithm. Most existing SAEAs are dominance based and thus are not well suited for handling many objectives. Therefore, the major contribution of the paper is to propose an efficient algorithm to manage the surrogates for handling a large number of objectives. To this end, we adopt the reference vector guided evolutionary algorithm (RVEA) [8] for many-objective optimization to be used as an evolutionary algorithm. Two sets of reference vectors adaptive and fixed, together with uncertainty information from the Kriging models as well as the location of the individuals are exploited for surrogate management. To limit the computation time for training the Kriging models, a strategy for choosing training samples is proposed so that the maximum number of training data is fixed.

The rest of the paper is organized as follows. In Section II, we provide a relatively detailed description of RVEA as well as Kriging models so that the paper is self-contained. The Kriging assisted RVEA, called K-RVEA is introduced in Section III. Section IV presents the numerical results of K-RVEA on benchmark problems and compared them with a few state-of-the-art SAEAs. Finally, conclusions are drawn and future work briefly discussed in Section V.

II. Background

In this section, we first summarize main components of RVEA, which we use as the underlying evolutionary algorithm. Next, we present a brief description of Kriging, including a discussion about its advantages and disadvantages over other surrogate models.

A. Reference vector guided evolutionary algorithm

Two major difficulties in solving problems with high number of objectives are convergence to the Pareto front and maintaining a good diversity between solutions. Several evolutionary algorithms have been proposed for solving many-objective optimization problems, by, for instance, using a revised dominance relationship, decomposing the multi-objective optimization into several single objective optimization problems, an indicator-based objective function, or using reference points. For more details about these algorithms and challenges in solving problems with more than three objectives, see [23], [29], [45].

RVEA is an EMO algorithm most recently developed for many-objective optimization [8]. While MOEA/D [50] and

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2016.2622301, IEEE Transactions on Evolutionary Computation

NSGA-III [14] use a set of weights and reference points, respectively, RVEA adopts a set of reference vectors. The main difference between RVEA and the MOEA/D and NSGA-III lies in its selection strategy. In RVEA, selection is based on a criterion known as angle penalized distance (APD), which is used to manage both convergence and diversity. It has been shown [8] that APD is better scalable to the increase in the number of objectives in maintaining a balance between convergence and diversity. APD relies on a set of reference vectors, which partitions the objective space into a number of subspaces, where selection of individuals is performed independently. The main components of RVEA are presented in Algorithm 1.

Algorithm 1 RVEA

Input: t_{max} = maximum number of generations; N = number of reference vectors; $V_0 = \{v_{01}, v_{02}, \dots v_{0N}\}$ a set of unit reference vectors;

Output: nondominated solutions from population $P_{t_{max}}$

1: Create an initial population P_0 of size N randomly and set generation counter t = 0
2: **while** $t < t_{max}$ **do**
3: Generate offspring Q_t
4: Combine parent and offspring populations, $L_t = P_t \cup Q_t$
5: Select parents (P_{t+1}) for the next generation
6: Update reference vectors (V_{t+1})
7: **end while**

RVEA uses elitism and offspring generation strategies similar to other state-of-the-art EMO algorithms such as NSGA-II [15] and NSGA-III [14]. RVEA distinguishes itself with NSGA-III in its selection strategy and the adaptation of reference vectors, which are Steps 5 and 6 in Algorithm 1. In the following, we present the four main components of RVEA, i.e., generation of reference vectors, assignment of individuals to reference vectors, selection and adaptation of reference vectors.

1) Generation of reference vectors: RVEA uses a set of reference vectors in the objective space to guide the search process. To generate a uniformly distributed set of reference vectors, first a set of uniformly distributed reference points (p) is generated on a unit hyperplane using the canonical simplex-lattice design method [11], [7].

$$\begin{cases} p_i = (p_i^1, p_i^2, \dots, p_i^k), \\ p_i^j = \left\{ \frac{0}{H}, \frac{1}{H}, \dots \frac{H}{H} \right\}, \sum_{j=1}^{k} p_i^j = 1 \end{cases} \quad (2)$$

where $i = 1, 2, \dots, N$ with N being the number of uniformly distributed points, k is the number of objectives and H is a positive integer used in simplex-lattice design. An example is shown in Figure 1 for a biobjective optimization problem, where the dots represent the reference points generated on a unit hyperplane. The corresponding reference vector is then obtained by projecting the reference points from the hyperplane to a hypersphere

$$v_i = \frac{p_i}{||p_i||}. \quad (3)$$

where $||p_i||$ represents the L_2-norm of p_i. As a result, these

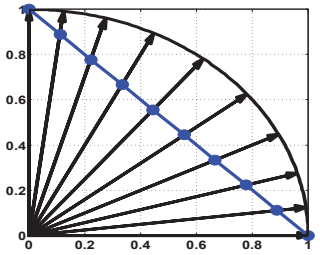


Figure 1. An illustrative example of reference vectors for a biobjective optimization problem.

reference vectors partition the objective space into a number of subspaces. In the next subsection, we describe how the individuals in a population are assigned to these reference vectors and how the population is partitioned into different subpopulations.

2) Assignment of individuals to reference vectors: After the generation of the reference vectors, individuals are assigned to them as follows. First, objective values of all individuals at the current generation are are translated, i.e. $\bar{f}_i^j = f_i^j - f_i^*$, where f_i^j represents the objective value of f_i for the j^{th} individual and f_i^* the minimum objective values of f_i at the current generation. We denote the translated objective vector by $\bar{f} = (\bar{f}_1, \bar{f}_2, \dots, \bar{f}_k)$. Translation of objective functions ensures that the initial point of reference vectors is always the origin and all the translated objective values are inside the positive orthant. After the translation, the acute angle is measured between an individual and all the reference vectors. For instance, let us consider the situation shown in Figure 2, with two reference vectors v_i and v_{i+1}, and three individual $\bar{f}^1$, $\bar{f}^2$ and $\bar{f}^3$. As the angle θ_i^1 between the individual $\bar{f}^1$ and the reference vector v_i is less than the angle θ_{i+1}^1 between the individual and the other reference vector v_{i+1}, this individual is assigned to the first reference vector v_i. Similarly, $\bar{f}^2$ and $\bar{f}^3$ will be assigned to reference vector v_i and v_{i+1}, respectively. Therefore, an individual is assigned to a reference vectors if and only if the angle between it and the reference vector is minimum among all reference vectors. In this way, assignment of individuals to the reference vectors partitions the population into subpopulations. Other notations in Figure 2 are used in the next subsection.

3) Selection mechanism in each subpopulation: After the generation of reference vectors and the assignment of individuals them, one individual is selected from each subpopulation (Step 5 in Algorithm 1). The selection criterion consists of two subcriteria that are meant for managing convergence and diversity, respectively. Convergence is taken care by the distance between the translated objective vector and the origin. As selection is performed in each subpopulation independently, let us take the subpopulation corresponding to reference vector v_i. In the illustrative example shown in Figure 2, two individuals $\bar{f}^1$ and $\bar{f}^2$ are assigned to the reference vector v_i and the distance between them and the origin is denoted by $||\bar{f}^1||$ and $||\bar{f}^2||$, respectively. As the aim is to find solutions closer to the origin, individual $\bar{f}^1$ is preferred over individual $\bar{f}^2$ and

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2016.2622301, IEEE Transactions on Evolutionary Computation

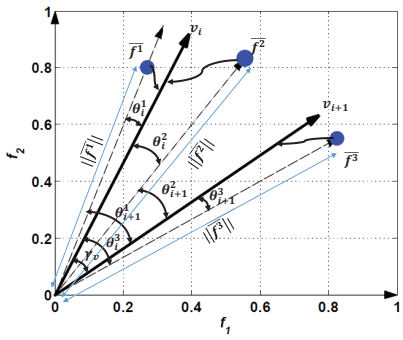


Figure 2. An illustration for assignment of individuals to a reference vector assignment and calculation of the selection criteria.

will therefore be selected for this subpopulation.

In RVEA, diversity is accounted by the angle between the translated objective vector and the reference vector the individual is assigned to. The individual with the smallest angle is preferred over other individuals. For instance, for reference vector v_i in Figure 2, individual $\bar{f}^1$ with the angle θ_i^1 is preferred over individual $\bar{f}^2$ as $\theta_i^1 < \theta_i^2$.

To combine the two subcriteria for convergence and diversity, the following angle penalized distance (APD) is defined:

$$d^j = (1 + P(\theta^j)) \cdot ||\bar{f}^j||, \tag{4}$$

where $||\bar{f}^j||$ is the distance from the translated objective vector corresponding to the j^{th} individual to the origin, and θ^j is the angle between the j^{th} individual and the reference vector it is assigned to. In APD, $P(\theta^j)$ is the penalty function defined as follows:

$$P(\theta_i) = k \cdot (\frac{t}{t_{max}})^{\alpha} \cdot \frac{\theta^j}{\gamma_v}, \tag{5}$$

where γ_v is defined as the smallest angle between the reference vectors v_i and its closest neighboring reference vector v_j i.e. $\gamma_v = min_{i \in \{1,\ldots,N\}, i \neq j} \langle v_i, v_j \rangle$. The angle γ_v is used to normalize the angles and is important when the distribution of the reference vectors is either too dense or too sparse. As highly dense or sparse reference vectors may generate a very small or a very large angle between the individual and the reference vector, normalization of the angle can be helpful to alleviate this problem. Parameter α is used to change the rate of the penalty function $P(\theta)$. The rate of the penalty function is used to emphasize convergence at the early stage and diversity at the later stage of the evolutionary search process. For instance, at the early stage of solution process, convergence is preferred to push the individuals closer to the Pareto front. Once individuals have converged to the Pareto front, diversity is then preferred by distributing individuals along the Pareto front. Therefore, the rate of the penalty function depends on the current generation number t, the maximum number of generations t_{max} and parameter α used in (5). As careful empirical studies for setting the parameter α have been performed in [8], we use the same parameter setting in this work. After calculating APD for all individuals in each subpopulation, one individual with the minimum APD value is selected from each subpopulation for the next generation.

4) Adaptation of reference vectors: In order to find a set of uniformly distributed nondominated individuals as close to the Pareto front as possible. For some optimization problems, e.g. those in the WFG test suite [22] where objective functions are scaled to different ranges, a uniformly distributed set of reference vectors is not best suited for getting uniformly distributed individuals. To tackle this issue, one possible solution is to adapt the reference vectors. In RVEA, reference vectors are adaptive. In other words, they change their position according to the location of individuals in the objective space. The adaptation of reference vectors for the next generation (v_{t+1}) is applied in the following way:

$$v_{t+1,i} = \frac{v_{0,i} \circ (z_t^{max} - z_t^{min})}{||v_{0,i} \circ (z_t^{max} - z_t^{min})||}, \tag{6}$$

where $\circ$ represents the Hadamard product [1] that multiplies two matrices of the same size element-wise, $v_{0,i}$ is the uniformly generated reference vector in the initialization phase in RVEA and z_t^{max} and z_t^{min} are the maximum and minimum values of each objective function in the t^{th} generation, respectively. The adaptation of reference vectors ensures that a set of uniformly distributed nondominated individuals will be obtained even for optimization problems whose objective functions have different ranges.

B. Kriging

We use Kriging, also known as Gaussian process to approximate each objective function. As per the survey on computationally expensive multiobjective optimization problems [9], Kriging has been frequently used for surrogate techniques, mainly because it is able to deliver uncertainty information of the approximated values, which is very useful in managing surrogates [24]. In this work, we use uncertainty information from Kriging models to update the surrogates, which will be further discussed in the next section.

Kriging approximates the objective function value of an individual x as

$$y(x) = \mu(x) + \epsilon(x), \tag{7}$$

where μ is the prediction of a regression model $F(\beta, x)$ i.e. $\mu = F\beta$ and $\epsilon(x)$ is a Gaussian distribution of zero mean and the standard deviation σ

$$\epsilon(x) = N(0, \sigma^2). \tag{8}$$

The regression model $F(\beta, x) = \beta_1 g_1(x) + \ldots + \beta_l g_l(x)$ is a linear combination of l chosen functions with coefficients β.

To get an approximated value from (7) for any new input, Kriging model needs to be trained using training samples, which are the pre-evaluated individuals in SAEAs. Let matrix $X = \left[x^1, \ldots, x^{N_I}\right]^T$ represent the training data in the decision space with their corresponding objective vector $y = \left[y^1, \ldots, y^{N_I}\right]^T$, where $i = 1, 2, \ldots, N_I$ represents the number of samples or the size of the training data. One should note that the size of X is $N_I \times n$, where n is the number of decision variables, i.e., $x^i = [x_1, \ldots, x_n]$ for $i = 1, \ldots, N_I$.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2016.2622301, IEEE Transactions on Evolutionary Computation

For any two arbitrary inputs x^i and x^j, the covariance between two random processes $\epsilon(x^i)$ and $\epsilon(x^j)$ is defined by

$$cov[\epsilon(x^i), \epsilon(x^j)] = \sigma^2 \mathbf{R}([R(x^i, x^j)]), \tag{9}$$

where $\mathbf{R}$ is the correlation matrix of size $N_I \times N_I$

$$\mathbf{R} = \begin{bmatrix} R(x^1, x^2) & \cdots & R(x^1, x^{N_I}) \\ \vdots & \ddots & \vdots \\ R(x^{N_I}, x^1) & \cdots & R(x^{N_I}, x^{N_I}) \end{bmatrix} \tag{10}$$

and $R(x^i, x^j)$ is the correlation function between $\epsilon(x^i)$ and $\epsilon(x^j)$. The commonly used correlation function is

$$R(x^i, x^j) = exp(-\sum_{k=1}^{n} \theta_k |x_k^i - x_k^j|^2), \tag{11}$$

where $\theta_i, i = 1, \ldots, N_I$ denote the hyperparameters.

For a new input $\bar{x}$, an approximated value from (7) can be written as

$$\bar{y} = \beta + r^T(\bar{x})\mathbf{R}^{-1}(y - F\beta), \tag{12}$$

where y contains the values of given N_I individuals, $r(\bar{x})$ is the correlation vector of size N_I between the new input $\bar{x}$ and the training data $\left[x^1, \ldots, x^{N_I}\right]$ i.e.

$$r^T(\bar{x}) = \left[R(\bar{x}, x^1), \ldots, R(\bar{x}, x^{N_I})^T\right]. \tag{13}$$

To obtain a new approximated value $\bar{y}$, we need to specify coefficients β and hyperparameters θ. Equation (12) has the generalized least square solution,

$$\beta = (F^T\mathbf{R}^{-1}F)^{-1}F^T\mathbf{R}^{-1}y \tag{14}$$

and the estimated variance σ^2 is given by

$$\sigma^2 = \frac{1}{N_I}(y - F\beta)^T\mathbf{R}^{-1}(y - F\beta). \tag{15}$$

Values of θ are obtained by maximizing the likelihood function

$$\psi(\theta) = -\frac{1}{2}(N_I \ln \sigma^2 + \ln det(\mathbf{R})) \tag{16}$$

where *det*($\mathbf{R}$) is the determinant of the correlation matrix $\mathbf{R}$. After getting hyperparameters θ, coefficients β and the variance σ^2 are calculated from (14) and (15), respectively, which are further used to approximate the objective function value from (12).

While Kriging is a very attractive surrogate model due to its ability to deliver uncertainty information, it also suffers from potentially serious weaknesses resulting from the computational complexity for training surrogates. As indicated in [20], the computational complexity of training Kriging is $O(n^3)$, where n is the number of training samples. The issue of high computational complexity will become worse if the hyperparameters θ are determined by maximize the likelihood function using an optimization algorithm, which has often been done in Kriging assisted SAEAs. For instance, MOEA/D-EGO uses differential evolution [44] and SMS-EGO employs covariance matrix adaptation (CMA-ES) [19] to optimize the hyperparameters, while in ParEGO, the Nelder and Mead algorithm [37] is used.

In this work, we use a modification of Hookes and Jeeves algorithm [21], which is implemented in DACE toolbox [30]. The main reason is that it is not practical to use population based techniques to optimize the hyperparameters due to the prohibitively high computation time thus incurred, since in SAEAs, Kriging models need to be frequently re-trained. We will provide a brief empirical comparison in Section IV.

III. Surrogate-assisted reference vector guided evolutionary algorithm

Model management is crucial to the success of surrogate-assisted evolutionary algorithms [24]. It is mainly concerned with how to use and update surrogates, including choosing individuals to be re-evaluated using the original objective functions. These re-evaluated individuals can then be used as training data for updating (retraining) the surrogate. Both convergence and diversity have to be taken into account in selecting individuals to be re-evaluated, which becomes more difficult for problems with a large number of objectives. In this paper, we focus on selection of training data in such a way that both convergence and diversity are managed given a large number of objectives, which is one major contribution of this paper.

The computation time for training the surrogate depends on the size of the training data set and the type of the surrogate used. The Kriging model is widely used due to its unique property of being able to predict with an error bound. Unfortunately, the computation time for training Kriging models will become prohibitive when the number of training data is large. Therefore, the second contribution of this paper is the proposal of a strategy to choose training samples so that the size of the training data can be kept sufficiently small. To this end, we use an archive to store the training data for updating the Kriging model.

The proposed Kriging-assisted reference vector guided evolutionary algorithm, called K-RVEA, is presented in Algorithm 2. This algorithm consists of three main phases, the initialization phase followed by the phase where a surrogate is used and finally updated in the last phase. Algorithm 2 is composed of two other algorithms, Algorithm 3 and Algorithm 4. Algorithm 3 selects the individuals for re-evaluation, and therefore also for updating the surrogate, while Algorithm 4 manages the training data in archive A_1. In addition to the training archive A_1, a second archive A_2 is used to store non-nondominated solutions as the final solution set.

A. Initialization

In the initialization phase, an initial population is generated e.g. using the Latin hypercube sampling [32]. These individuals are evaluated with the original expensive functions and added to two archives A_1 and A_2. Individuals stored in A_1 are used to build a Kriging model for each objective function.

B. Using the surrogate

In the phase of using the surrogate, we use Kriging models instead of the original functions to calculate objective function values. Kriging models are used for fitness evaluations for a prefixed number of generations without updating them.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2016.2622301, IEEE Transactions on Evolutionary Computation

Algorithm 2 K-RVEA

Input: FE_{max}, maximum number of expensive function evaluations; u = number of individuals to be re-evaluated and to be used for updating Kriging models; w_{max} = prefixed number of generations before updating Kriging models;
Output: nondominated solutions of all evaluated ones from A_2
/*Initialization*/
1: Create an initial population of size N_I using e.g. the Latin hypercube sampling, initialize the number of function evaluations- *FE* = 0, the generation counter for using Kriging models *w* = 1 and a counter for the number of updates, $t_u = 0$. Initialize two empty archives A_1 and A_2, i.e. $|A_1| = |A_2| = \phi$
2: Evaluate the initial population with the original functions and add them to A_1 and A_2, update *FE* = *FE*+N_I, update $|A_1| = |A_1| + N_I$ and $|A_2| = |A_2| + N_I$
3: Train a Kriging model for each objective function by using individuals in A_1
4: **while** $FE \leq FE_{max}$
/*Using the surrogate*/
5: **while** $w \leq w_{max}$
6: Run Steps 3-6 of Algorithm 1 with Kriging models instead of the original functions and update *w*=*w*+1
7: **end while**
/*Updating the surrogate*/
8: Select *u* individuals using Algorithm 3 and re-evaluate them with the original functions and update *FE* = *FE* + u
9: Add individuals from step 8 to A_1 and A_2 and update $|A_1| = |A_1| + u$ and $|A_2| = |A_2| + u$
10: Remove $|A_1| - N_I$ individuals from A_1 using Algorithm 4, update *w* = 1 and $t_u = t_u + 1$ and go to step 3
end while

Empirically, the prefixed number of generations should be set in such a way that it allows the evolutionary algorithm to perform adequate search on the fitness landscape defined by the surrogate, while the search should be terminated if no further improvement in either convergence or diversity can be made. Ideally, the frequency of updating the surrogates can be made adaptive based on their performance, e.g., as done in [26]. However, a rigorous guideline for adapting the frequency still lacks. For simplicity, in this work we adopt a prefixed frequency based on a sensitivity analysis of the performance on the prefixed number of generations.

Once the function evaluations are done, simulated binary crossover and polynomial mutations are applied to generate offspring, similar to [15]. The parent and offspring populations are combined and then the selection criteria in RVEA detailed in Section II are used to select parents for the next generation.

C. Updating the surrogate

After an evolutionary search using the Kriging models for a fixed number of generations, the Kriging models will be updated. As previously mentioned, selection of individuals to be re-evaluated using the original functions, which will also be used for updating the surrogates, is essential for the performance of SAEAs. In this paper, we use information from the underlying evolutionary algorithm, RVEA, and uncertainty information from the Kriging models for selecting individuals to be re-evaluated and then for re-training the surrogate. As mentioned in Section I, selection of uncertain individuals not only helps in finding the unexplored regions but can also improve the quality of the surrogate. Therefore, we select individuals with the maximum uncertainty whenever diversity is needed. If a satisfactory degree of diversity is already achieved, we select individuals with the minimum angle penalized distance, which is one of the selection criteria in RVEA that contributes to convergence. Full details of the strategy for selecting individuals for re-evaluation is given in the next subsection, also summarized in Algorithm 3. The selected individuals are then re-evaluated with the original functions and these data samples are added to both archives A_1 and A_2. To keep a fixed number of individuals in the archive A_1, we will eliminate extra individuals from from A_1 using a strategy to be introduced in the following.

A maximum number of function evaluations is used as the termination criterion of the evolutionary optimization process. After the evolutionary search is complete, nondominated individuals in A_2 are taken as the final solutions.

1) Strategy for selecting individuals for re-evaluation: In RVEA, the reference vectors are adaptive. In this work, we introduce an additional set of fixed reference vectors that are evenly distributed in the objective space. The number of fixed reference vectors is the same as the number of adaptive vectors. For convenience, we denote the fixed reference vectors by V_f and the adaptive ones by V_a.

The fixed reference set is mainly for determining whether diversity or convergence should be prioritized in selecting individuals for re-evaluation. This is done by comparing the number of empty vectors in the fixed reference set during the previous surrogate update, denoted by $|V_f^{ia}|_{t_u-1}$ and that in the current update denoted by $|V_f^{ia}|_{t_u}$. A vector is called empty or non-active if no individual is assigned to this vector. In case $|V_f^{ia}|_{t_u}$ - $|V_f^{ia}|_{t_u-1} > \delta$, where $\delta > 0$ is a small integer, which means the increase in the number of empty fixed reference vectors has exceeded a given threshold, diversity should be prioritized. In this case, individuals for re-evaluation should be selected based on the uncertainty information offered by the Kriging models. By contrast, if $|V_f^{ia}|_{t_u}$ - $|V_f^{ia}|_{t_u-1} < \delta$, which means that the diversity of the population is not the major concern, priority will be given to the convergence criterion. In other words, individuals should be selected using the APD according to the adaptive reference vectors.

The next step is to determine which individuals should be selected according to either the amount of uncertainty or the value of APD. To this end, we divide the active adaptive reference vectors into a given number of clusters (Step 1 in Algorithm 3), from each of which one individual that has either the maximum amount of uncertainty (in case diversity is prioritized) or the minimum APD (in case convergence is to be taken care of) in the corresponding cluster. Thus, the number of clusters is always equal to the number of individuals,

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2016.2622301, IEEE Transactions on Evolutionary Computation

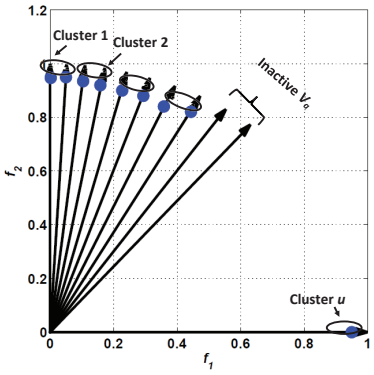


Figure 3. Clustering of active adaptive reference vectors V_a into a predefined number of clusters u.

denoted by u, to be selected for re-evaluation and for updating the surrogate. The process for selecting solutions to be re-evaluation described above is summarized in Algorithm 3.

Algorithm 3 Selection of individuals for updating the surrogate

Input: Sets of adaptive V_a and fixed V_f reference vectors, I = individuals obtained from the latest generation, $|V_f^{ia}|_{t_u-1}$ = number of inactive fixed reference vectors from the previous update, δ = parameter to decide whether to use APD or uncertainty from Kriging models for updating Kriging models, u = number of individuals to update Kriging models
Output: u = Individuals for updating Kriging models
1: Cluster active adaptive reference vectors into $min\{u, |V_a^a|\}$ clusters
2: Identify individuals closest to active adaptive reference vectors within each cluster
3: Assign I to fixed reference vectors and identify the number of inactive fixed reference vectors, i.e. $|V_f^{ia}|$
4: Calculate the change in the number of inactive fixed reference vectors from the previous update i.e. $\Delta V_f = |V_f^{ia}|_{t_u} - |V_f^{ia}|_{t_u-1}$
5: **If** $\Delta V_f \leq \delta$
6: Select one individual from each cluster with minimum APD
7: **else**
8: Select one individual from each cluster with maximum uncertainty
9: **end if**

To elaborate the above selection process, let us consider a few different situations shown in Figure 4 for a biobjective optimization problem, where the fixed reference vectors as well as the individuals (denoted by dots) associated to each vector are shown in updating Kriging models. Figure 4(a) illustrate the fixed reference vectors and the individuals assigned to them during the previous update, i.e., at the counter for updating the surrogate $t_u - 1$. Two different cases at the current update, i.e., at t_u, are shown in Figure 4(b) and Figure 4(c), respectively. In the situation shown in 4(b), the number of inactive fixed reference vectors, denoted by $|V_f^{ia}|_{t_u}$, has decreased, which indicates that diversity is not a concern and convergence should be emphasized. By contrast, Figure 4(c) shows a situation in which the number of inactive fixed reference vectors has increased, which indicates that diversity needs to be prioritized in updating the Kriging models. In the former case shown in Figure 4(b), individuals for re-evaluation are selected based on APD calculated using the reference vector set, while in the latter case, the amount o uncertainty, where uncertainty is calculated using the average of the standard deviations obtained from Kriging models. Selected individuals are re-evaluated with the original functions and the obtained data added to both archives A_1 and A_2. Note that if the number of active adaptive reference vectors is smaller than u, i.e., $|V_a^a| < u$, we group them into $|V_a^a|$ clusters. In the first update of the Kriging models, APD is always used for selecting individuals for re-evaluation.

In the following, we describe the strategy for managing the training data in A_1, when the number of available training data is large than the predefined maximum number defined by the size of A_1.

2) Managing the training data: In order to limit the computation time for re-training the Kriging models, the number of training data in archive A_1 is fixed. To this end, some data need to be discarded if the number of available training data is larger than the archive size. Which data samples should be kept in A_1 becomes critical for the quality of the Kriging model and thus the overall performance of K-RVEA. In this work, the maximum size of the training data, i.e., the size of archive A_1, is set to N_I. The main steps for managing training data archive A_1 are presented in Algorithm 4.

Algorithm 4 Managing individuals in the archive

Input: Archive A_1, adaptive reference vectors V_a, u= individuals selected to update Kriging models, N_I=maximum number of individuals in the archive A_1
Output: Updated individuals in A_1
1: Remove duplicate individuals from the training archive A_1 and update $|A_1|$
2: **If** $|A_1| > N_I$
3: Assign u individuals to V_a and identify inactive adaptive reference vectors, denote these reference vectors by V_a^{ia}
4: Assign $A_1 \setminus u$ individuals i.e. individuals other than recently evaluated ones in the archive to V_a^{ia}
5: Identify active reference vectors from inactive adaptive reference vectors V_a^{ia}
6: Cluster active set of inactive adaptive reference vectors V_a^{ia} into $N_I - u$ clusters
7: Select one individual from each cluster randomly and remove other individuals
11: **end if**

We first add recently evaluated individuals (u) obtained with Algorithm 3 to archive A_1 and remove duplicate data points from it (step 1 in Algorithm 4). If the number of training data is still larger than the archive size, we eliminate some training samples other than the recently evaluated one

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2016.2622301, IEEE Transactions on Evolutionary Computation

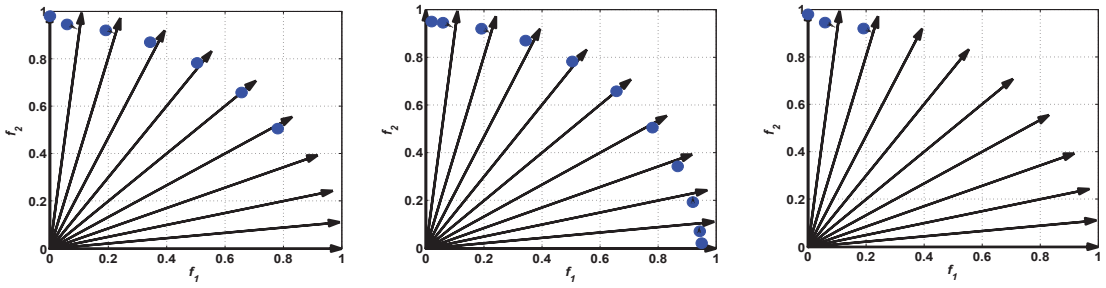


Figure 4. Different cases for fixed reference vectors while updating Kriging models. (a): An example of the fixed reference vectors, (b): An example showing the fixed reference vectors at the current update, where the number of inactive reference vectors has decreased compared to the previous update in (a), (c): An example showing the fixed reference vectors at the current update, where the number of inactive reference vectors has increased compared to the previous update in (a).

(step 2 in Algorithm 4). For this purpose, data points other than the recently evaluated individuals are assigned to the adaptive reference vectors. For instance, consider the situation in Figure 5 for a biobjective optimization problem. In Figure 5(a), individuals (training data) obtained with Algorithm 3 are assigned to the adaptive reference vectors V_a and inactive reference vectors are identified, denoted by V_a^{ia}. Then, we assign $A_1 \setminus u$ data points to these vectors V_a^{ia} and identify active reference vectors from them, as shown in Figure 5(b) (steps 4 and 5 in Algorithms 4). The active adaptive reference vectors are then grouped into $N_I - u$ clusters and data points assigned to these reference vectors in each cluster are identified (step 6 Algorithm 4). We randomly select one data point from each cluster and eliminate the rest of the data. In this way, a fixed number of diverse set of training data is maintained in A_1, thereby to improve the quality of Kriging as much as possible while keeping computation time limited.

IV. Numerical experiments

In this section, numerical experiments are conducted to examine the performance of K-RVEA on the DTLZ benchmark problems [16] for 3, 4, 6, 8 and 10 objectives are presented. The number of decision variables was set to 10. We also compared K-RVEA with representative Kriging based SAEAs, e.g. MOEA/D-EGO [51], SMS-EGO [41], [46], ParEGO [28], and with the original RVEA without using the surrogates. We also included RVEA for comparison to give the reader a sense of how differently an algorithm with and without surrogates performs on computationally expensive problems.

A. Parameter settings

1) Number of individuals to be evaluated using the original objective functions in the initialization phase (from the literature [28], [51]) = maximum number of individuals in the training archive, $N_I = 11n - 1$
2) Number of independent runs = 10
3) Maximum number of function evaluations, FE_{max} = 300
4) Number of individuals to update Kriging models, $|u|$ = 5
5) Number of reference vectors (N): number of reference vectors is determined by the design factors for simplex-lattice design [11] and the number of objectives and listed in the supplementary material
6) Parameter while updating the Kriging models $\delta = 0.05N$
7) Number of generations before updating the Kriging models w_{max} = 20.

For RVEA, a population size of 50 was used and for other algorithms, the same parameters were used as recommended by the authors in the respective articles. Inverted generational distance (IGD) [3] used as the performance measure to compare different algorithms. A Wilcoxon rank sum test was used to compare the results obtained by K-RVEA and the other four algorithms at a significance level of 0.05. Results are collected in Tables I, II and III, where symbol $\uparrow$ indicates that K-RVEA performed statistically better than a compared algorithm, and $\downarrow$ means that other algorithms performed better than K-RVEA, while $\approx$ means that there is no significant difference between the results obtained by K-RVEA and the other algorithm. One should note that for a given number of decision variables, the landscape of the problems is getting easier as the number of objectives increases. This is due to the fact that the number of decision variables for driving convergence decreases with the increase in the number of objectives [22]. Therefore, we have added the WFG suite [22] in our experiments and present the results in the supplementary material.

B. Performance on DTLZ problems

The results obtained with the four compared algorithms over 25 independent runs are collected in Table I. No results are given for ParEGO for more than four objectives as the current implementation given by the authors of ParEGO was limited to four objectives, which is denoted by 'NA' in the tables. The presented results include the minimum, mean and maximum values of IGD. The best values are highlighted.

Before we discuss the results, we want to mention an important issue in measuring the performance of many-objective evolutionary algorithms. IGD and hypervolume are two widely used performance indicators. However, the evaluation result may heavily depends on their parameters, particularly when the number of objectives is large. For instance, the IGD value is very sensitive to the size of the reference set, which has not been explicitly discussed. In [4], [48], 100000 reference points were used irrespective of the number of objectives, while in [52], only 500 reference points were used. In [5],

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2016.2622301, IEEE Transactions on Evolutionary Computation

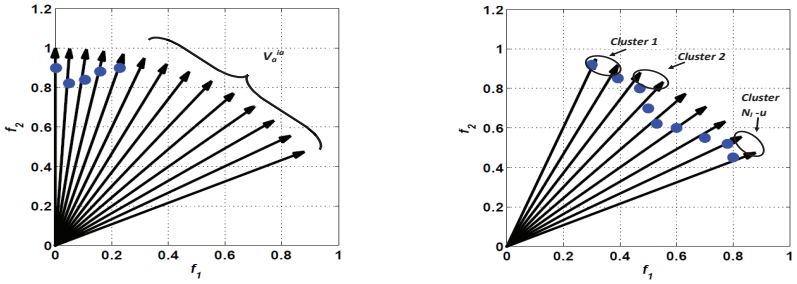


Figure 5. Managing training data in the archive A_1, (a):Assignment of the recently evaluated individuals u to the active adaptive reference vectors V_a, (b):Assignment of $A_1 \setminus u$ individuals to inactive adaptive reference vectors V_a^{ia}.

Table I. Statistical results for IGD values obtained by K-RVEA, RVEA and ParEGO. The best results are highlighted

Problem	k	K-RVEA				RVEA				ParEGO		
		min	mean	max		min	mean	max		min	mean	max
DTLZ1	3	82.03	106.9	125.2	≈	42.65	82.87	115.1	↓	**13.42**	**52.47**	**112.7**
	4	48.23	73.21	101.4	≈	39.65	59.18	97.71	↓	**18.63**	**45.45**	**87.76**
	6	**8.031**	28.83	**35.22**	≈	12.24	**22.94**	36.85		NA	NA	NA
	8	**0.699**	**6.991**	**13.29**	≈	1.250	7.406	15.66		NA	NA	NA
	10	0.198	0.347	**0.655**	≈	**0.193**	**0.339**	1.105		NA	NA	NA
DTLZ2	3	**0.092**	**0.155**	0.262	↑	0.227	0.288	0.335	↑	0.151	0.191	**0.243**
	4	**0.191**	**0.276**	**0.376**	↑	0.280	0.332	0.383	↑	0.289	0.337	0.408
	6	**0.316**	**0.342**	**0.362**	↑	0.375	0.404	0.440		NA	NA	NA
	8	**0.360**	**0.395**	**0.522**	↑	0.466	0.541	0.704		NA	NA	NA
	10	**0.419**	**0.446**	**0.470**	↑	0.539	0.608	0.733		NA	NA	NA
DTLZ3	3	181.5	280.1	353.1	≈	133.7	256.1	347.9	↓	**81.15**	**145.5**	**261.6**
	4	85.56	210.9	314.5	≈	89.95	198.6	306.3	↓	**66.93**	**138.1**	**209.4**
	6	61.61	105.0	**156.4**	≈	**43.54**	**95.97**	157.7		NA	NA	NA
	8	12.36	26.49	43.51	≈	**8.569**	**25.27**	**42.17**		NA	NA	NA
	10	0.781	1.299	2.303	≈	**0.761**	**1.228**	**1.836**		NA	NA	NA
DTLZ4	3	**0.190**	0.448	**0.737**	≈	0.205	**0.399**	0.959	↑	0.387	0.646	0.947
	4	**0.268**	**0.458**	**0.648**	≈	0.320	0.514	0.737	↑	0.505	0.725	0.960
	6	**0.422**	**0.585**	**0.754**	≈	0.503	0.615	0.800		NA	NA	NA
	8	**0.547**	0.635	**0.728**	≈	0.554	**0.628**	0.731		NA	NA	NA
	10	**0.553**	**0.608**	**0.672**	↑	0.599	0.667	0.761		NA	NA	NA
DTLZ5	3	0.050	0.112	0.211	↑	0.201	0.247	0.316	↓	**0.039**	**0.055**	**0.072**
	4	**0.046**	**0.123**	**0.242**	↑	0.149	0.294	0.393	↑	0.090	0.288	0.428
	6	**0.032**	**0.102**	**0.153**	↑	0.159	0.280	0.431		NA	NA	NA
	8	**0.023**	**0.048**	**0.107**	↑	0.104	0.260	0.748		NA	NA	NA
	10	**0.009**	**0.017**	**0.022**	↑	0.224	0.488	0.746		NA	NA	NA
DTLZ6	3	**2.121**	**2.727**	**3.343**	↑	3.651	4.960	5.613	↑	5.030	6.378	6.867
	4	**1.306**	**2.446**	**3.060**	↑	3.027	4.044	5.208	↑	5.652	5.916	6.034
	6	1.133	**1.597**	**2.174**	↑	**1.025**	2.524	3.600		NA	NA	NA
	8	**0.377**	**0.660**	**1.049**	↑	0.247	1.004	1.870		NA	NA	NA
	10	**0.054**	**0.153**	**0.373**	↑	0.140	0.297	0.751		NA	NA	NA
DTLZ7	3	**0.088**	**0.111**	**0.150**	↑	0.400	0.515	0.637	↑	0.621	0.829	1.201
	4	**0.188**	**0.243**	**0.298**	↑	0.532	0.691	0.926	↑	0.719	0.892	1.149
	6	**0.391**	**0.500**	**0.627**	↑	0.889	1.088	1.808		NA	NA	NA
	8	**0.745**	**0.886**	**1.030**	↑	1.162	1.359	1.634		NA	NA	NA
	10	**0.917**	**1.030**	**1.134**	↑	1.343	1.900	3.327		NA	NA	NA

different sizes for the reference set were used for different numbers of objectives. Here, we also use different sizes of the reference set for different numbers of objectives, as we believe more reference points are needed as the number of objectives increases, referring to the Supplementary material for details. Similarly, different criteria for setting the reference point in calculating the hypervolume have been adopted in different papers. It has been found in [49] that choosing a reference point slightly better than the nadir point is able to strike a balance between convergence and diversity of the solution set. Therefore, in this work, we use the worst objective function values of the non-dominated solutions found by all compared algorithms plus a small threshold. The detailed comparative results in terms of hypervolume are provided in the Supplementary material due to the page limit. We must emphasize that how to fairly compare the performance of EAs for many-objective optimization has received little attention in the literature and deserves more research.

As can be seen in Table I, overall, K-RVEA performed the best among all algorithms compared in this study, except on DTLZ1 and DTLZ3. We surmise that DTLZ1 and DTLZ3 have many local Pareto optimal solutions. Both RVEA and K-RVEA try to keep a well-distributed set of solutions because of the distribution in the reference vectors and, therefore, the convergence rate is relatively slow on these problems. Nondominated solutions of the run producing the best IGD values for K-RVEA, RVEA and ParEGO for DTLZ1 are shown in Figure 6. As can be seen, solutions of K-RVEA and RVEA are better distributed than those of ParEGO. However, the IGD values of all three algorithms are high, in other words, the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2016.2622301, IEEE Transactions on Evolutionary Computation

solutions obtained by these algorithms are all far from the true Pareto front. These results indicate that solving such problems requires more function evaluations to reach the Pareto front.

To compare the results with SMS-EGO, we selected a different stopping criterion. As SMS-EGO tries to maximize the expected hypervolume improvement and was implemented in MATLAB, it took about seven hours to complete one run on a computer with the i5 processor and 4GB RAM. The needed large amount of computation time of SMS-EGO was also mentioned in [51], for which reason the authors compared their algorithm with SMS-EGO only on two test problems. In this paper, we allowed SMS-EGO to run for 24 hours with 10 parallel runs and stored all the solutions. The number of function evaluations reached during this time was used as the stopping criterion for comparison with the other algorithms. The results for three and four objectives are obtained with 120 and 115 function evaluations, respectively, which are presented in Table II. These results obtained with a small number of function evaluations show that K-RVEA performed better than the compared algorithms. We did not compare K-RVEA with SMS-EGO for problems within more than four objectives, as SMS-EGO requires even more time for such problems.

The comparison of K-RVEA and MOEA/D-EGO for three objective functions after 300 original function evaluations is shown in Table III. An implementation of MOEA/D-EGO from the authors was available for only two and three objectives. As can be seen in Table III, K-RVEA performed either better or comparably to MOEA/D-EGO for all problems except on DTLZ5. As the Pareto front of DTLZ5 is a curve that covers a small subspace in the objective space, most of the reference vectors in K-RVEA and RVEA are empty, i.e., no solutions are assigned to them. We observed that for both K-RVEA and RVEA, almost 70% of the reference vectors are empty, which makes the solution process to converge slowly to the Pareto front. For such problems, a large number of reference vectors could be helpful while using K-RVEA.

Nondominated solutions from the run with the best IGD values obtained by the compared algorithms on the three-objective DTLZ7 are shown in Figure 7. As can be seen from the figure, nondominated solutions obtained of K-RVEA and RVEA are much closer to the Pareto front than those of ParEGO and MOEA/D-EGO. For DTLZ7 which has a disconnected Pareto front, RVEA and K-RVEA have a good potential to get solutions close to the Pareto front because of the adaptation in the reference vectors. Remind that we did not run SMS-EGO for optimization problems with more than four objectives as the runtime is prohibitive. In addition, a parallel coordinates plot for DTLZ2 with 10 objectives is presented in Figure 8, which we can see that the ranges of solutions obtained by K-RVEA are bigger than those obtained by RVEA. In other words, the solutions obtained by K-RVEA have a better distribution. The reason for a lower density of the solutions obtained by RVEA is due to the small population size. As the maximum number of function evaluations for termination is quite low and the performance of RVEA depends on the maximum number of generations, we reduced the population of RVEA to 50. To more convincingly demonstrate the performance of K-RVEA, we tested and compared the algorithm on the WFG problems [22]. Results in terms of both IGD and hypervolume are provided in the Supplementary material, which show that K-RVEA was able to perform better than the compared algorithms.

As mentioned in Section II.B, the computation time for training Kriging models varies a lot depending on the specific implementation and the number of training, which may become prohibitive large. One contribution of K-RVEA is the development of a strategy to select training samples reference vectors, the number of samples for training Kriging models is kept constant, the computation time for training K-RVEA is remains constant as the number of expensive fitness evaluations increases. To empirically verify this, the run time of the different implementations in the compared SAEAs for training the Kriging model has been investigated. The results over the number of training samples obtained on the 3-objective DTLZ2 are shown in Figure 9. We can observe that the training time of K-RVEA remains constant, as the maximum number of training samples is kept constant, the training time for ParEGO increases slightly. In contrast, the training time of MOEA/D-EGO increases quickly over the number of training samples as a piecewise continues function. As already mentioned, the computation time of SMS-EGO increases dramatically over the number of training samples. Note however that SMS-EGO and K-RVEA are implemented in MATLAB, ParEGO is implemented in C and MOEA/D in Java. Therefore, the absolute times used by the different algorithms may not be directly comparable, although the different behaviours of the change in training time over the number allowed true function evaluations are of more interest.

In what follows, we consider the effect of different parameters in K-RVEA. The parameter δ in Algorithm 2 is used to select individuals for updating surrogates to balance between convergence and diversity. We did a sensitivity analysis to see the effect of δ on the diversity and the performance of the algorithm. As diversity in both RVEA and K-RVEA can easily be measured using reference vectors, we studied the effect of δ by measuring the change in the number of empty reference vectors. We also measured the hypervolume to see the effect on the performance of the algorithm and provide the results in the supplementary material. As expected, increase in the value of δ deteriorates the diversity and thus the hypervolume. This is due to fact that frequency of using uncertainty information from Kriging models decreases with the increase in value of δ. In other words, if the change in the number of the empty reference vectors is smaller than δ, individuals are selected based on convergence, otherwise based on uncertainty of Kriging models. Increase in the value of δ will force the algorithm to select the individuals based on convergence and thus deteriorates the diversity. In this article we fixed the value of δ to be $0.05 \times N$, where N is the number of reference vectors used and adaptation of δ will be our future work.

Apart from δ, two other parameters can also influence the performance of K-RVEA, which are the number of individuals to be selected to update the surrogates and the number of generations (w_{max}) used before updating the surrogates. The first parameter depends on the characteristics of the problem, e.g., multi-modality, and the evolutionary algorithm used. We study

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2016.2622301, IEEE Transactions on Evolutionary Computation

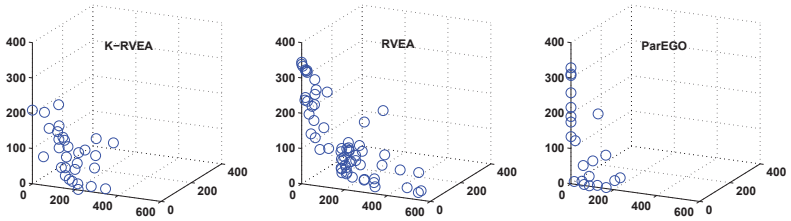


Figure 6. Nondominated solutions obtained with K-RVEA, RVEA and ParEGO of the run with the best IGD value for three-objective DTLZ1, which are all very far away from the true Pareto front.

Table II. Statistical results for IGD values obtained by K-RVEA, RVEA, ParEGO, SMS-EGO and MOEA/D-EGO for three and four objectives with 120 and 115 function evaluations, respectively. The best results are highlighted

Problem	k	K-RVEA				RVEA				ParEGO				SMS-EGO				MOEA/D-EGO		
		min	mean	max		min	mean	max		min	mean	max		min	mean	max		min	mean	max
DTLZ1	3	82.03	130.2	170.8	≈	**66.19**	**129.6**	171.4	≈	100.7	124.2	148.8	↑	85.47	114.2	**148.8**	↑	173.1	267.5	388.2
	4	61.92	90.29	115.7	≈	81.52	102.2	133.3	≈	75.23	99.81	128.5	↑	93.33	130.2	173.7		NA	NA	NA
DTLZ2	3	0.305	0.360	0.408	↑	0.401	0.436	0.494	≈	**0.271**	**0.356**	**0.396**	↑	0.365	0.444	0.522	↑	0.325	0.384	0.456
	4	**0.376**	0.427	**0.464**	↑	0.421	0.463	0.499	≈	0.384	**0.422**	0.474	↑	0.447	0.487	0.532		NA	NA	NA
DTLZ3	3	217.3	**324.3**	**383.7**	≈	236.3	366.6	495.6	≈	232.4	368.3	460.7	≈	220.8	325.3	459.2	≈	**189.7**	339.9	523.41
	4	173.8	302.1	370.5	≈	177.6	**283.1**	359.1	≈	**172.5**	291.9	**357.2**	≈	209.1	314.1	403.8		NA	NA	NA
DTLZ4	3	0.452	0.711	0.979	≈	0.537	0.678	0.967	≈	0.586	0.769	0.911	≈	0.649	0.690	0.722	↓	**0.393**	**0.434**	**0.479**
	4	**0.587**	**0.750**	0.914	≈	0.669	0.803	0.928	≈	0.716	0.819	0.993	≈	0.680	0.746	**0.815**		NA	NA	NA
DTLZ5	3	0.173	0.282	0.360	≈	0.272	0.326	0.397	↑	0.597	0.615	0.638	↑	0.447	0.498	0.546	↓	**0.170**	**0.215**	**0.308**
	4	**0.226**	**0.254**	**0.301**	↑	0.248	0.283	0.316	↑	0.432	0.454	0.484	↑	0.360	0.413	0.465		NA	NA	NA
DTLZ6	3	3.104	3.974	4.629	↑	5.737	6.110	6.462	↑	6.504	6.707	6.825	↓	**1.603**	**1.756**	**2.007**	↑	4.072	5.078	6.624
	4	**2.962**	**3.910**	**4.585**	↑	4.406	5.036	5.629	↑	5.720	5.868	6.024	↑	5.842	5.883	6.010		NA	NA	NA
DTLZ7	3	**0.119**	**0.167**	**0.216**	↑	0.632	0.760	1.264	↑	3.138	5.573	7.417	↓	0.241	0.260	0.277	↑	0.618	1.743	3.830
	4	**0.297**	**0.401**	**0.647**	↑	0.778	1.018	1.219	↑	5.304	7.376	8.921	↓	0.578	0.644	0.706	↑	NA	NA	NA

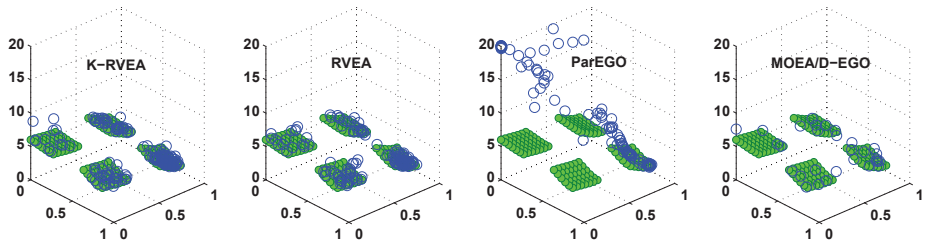


Figure 7. Nondominated solutions obtained by K-RVEA, RVEA, ParEGO and MOEA/D-EGO, denoted by circles, in the run with the best IGD value for three-objective DTLZ7, where the dots represent the Pareto front.

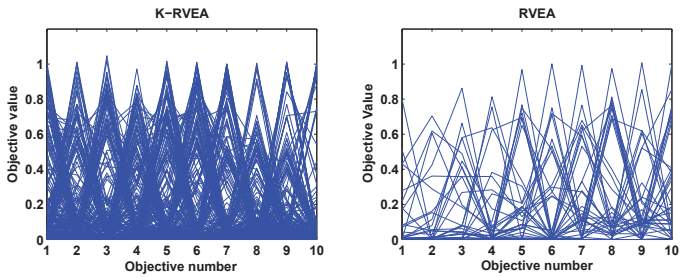


Figure 8. Parallel coordinate plot of the nondominated solutions obtained by K-RVEA and RVEA in the run with the best IGD value on the 10-objective DTLZ2.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2016.2622301, IEEE Transactions on Evolutionary Computation

Table III. Statistical results for IGD values obtained by K-RVEA and MOEA/D-EGO for three objectives after 300 function evaluations. The best results are highlighted

Problem	K-RVEA				MOEA/D-EGO		
	min	mean	max		min	mean	max
DTLZ1	**82.03**	**106.9**	**125.2**	↑	145.9	177.9	224.5
DTLZ2	0.092	0.155	0.262	≈	**0.081**	**0.103**	**0.212**
DTLZ3	181.5	280.1	353.1	≈	**161.5**	**205.9**	**281.8**
DTLZ4	**0.190**	0.448	0.737	≈	0.357	**0.436**	0.574
DTLZ5	0.050	0.112	0.211	↓	**0.035**	**0.046**	**0.071**
DTLZ6	2.121	2.727	**3.343**	≈	**0.491**	**2.551**	4.126
DTLZ7	**0.088**	**0.111**	**0.150**	↑	0.154	0.646	1.254

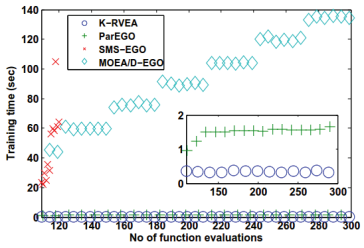


Figure 9. Training time over the number of function evaluations in K-RVEA, ParEGO, SMS-EGO and MOEA/D-EGO on the 3-objective DTLZ2.

the effect of the parameter in the Supplementary material. Our results show that the value of the parameter is problem-specific and an adaptive way of using the parameter is needed. For the second parameter i.e. the frequency of updating the surrogates or when to update the surrogate is very important in surrogate management, although, unfortunately, there is no solid theory for guiding when to update the surrogates. We have performed a sensitivity analysis on the performance of K-RVEA given different prefixed numbers of generations before the Kriging models are re-trained. The results are also included in the Supplementary material.

We also tested K-RVEA, RVEA, ParEGO and MOEA/D-EGO on a three objective real-world polymerization problem [34]. Even though K-RVEA and RVEA have been proposed for more than three objectives, they still performed better in solution quality and computation time than the compared algorithms. Details and results for this problem are given in the supplementary material.

V. Conclusions and Future Research

In this paper, a Kriging-assisted reference vector guided evolutionary algorithm, called K-RVEA, has been proposed for solving computationally expensive optimization problems with more than three objectives, where a Kriging model is used to approximate each objective function. We take care of both convergence and diversity in choosing individuals for re-evaluation with the original expensive objective functions. For this purpose, we introduced a set of fixed, uniformly distributed reference vectors in addition to the adaptive reference vectors in RVEA. In updating the Kriging models, attention is paid to limiting the computation time for training the surrogate by means of selecting training samples according to their relationships to the reference vectors, thereby limiting the number of training data.

We have examined the performance of K-RVEA on benchmark problems with 3, 4, 6, 8 and 10 objectives. We also compared K-RVEA with three state-of-the art SAEAs. Empirical results show that overall the K-RVEA obtained much better performance than the compared algorithms given the same number of function evaluations using the original expensive objective function.

In this paper, the number of decision variables was set to 10, as done in other papers in the literature that use Kriging as the surrogate model. This can be attributed to the factor that for solving optimization problems with a higher number of decision variables, much more training data will be needed, which will not only require more computational resource but also poses more serious challenges to Kriging based surrogate techniques. Nevertheless, it is highly desired that SAEAs can be applicable to optimization problems having a larger number of decision variables, which will be our future work. Another topic for future study is to investigate the performance of the proposed K-RVEA for constrained computationally expensive optimization problems. Finally, in the present work, we fixed the number of generations for updating the surrogates. Although our empirical results indicate that the performance of K-RVEA is relatively insensitive to the frequency of updating the surrogates for the benchmark problems studied in this work, our previous work [26] indicated that it is likely to adapt the frequency of updating the surrogates to further enhance the performance of SAEAs. Consequently, developing an adaptive method for updating the surrogates will also be our future research work.

VI. Acknowledgment

The authors would like to thank R. Cheng, J. Knowles, T. Wagner, Q. Zhang and A. Zhou for providing the codes of the algorithms compared in this work.

References

[1] T. Ando. Majorization relations for hadamard products. *Linear Algebra and its Applications*, 223-224:57–64, 1995.

[2] N. Azzouz, S. Bechikh, and L. Said. Steady state IBEA assisted by MLP neural networks for expensive multi-objective optimization problems. In C. Igel, editor, *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 581–588. ACM, 2014.

[3] P. Bosman and D. Thierens. The balance between proximity and diversity in multiobjective evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 7:174–188, 2003.

[4] P. Chandan, M. Islam, K. Murase, and X. Yao. Evolutionary path control strategy for solving many-objective optimization problems. *IEEE Transactions on Cybernetics*, 45:702–715, 2015.

[5] J. Cheng, G. Yen, and G. Zhang. A many-objective evolutionary algorithm with enhanced mating and environmental selections. *IEEE Transactions on Evolutionary Computation*, 19:592–605, 2015.

[6] L. Cheng, K. Shimoyama, and S. Obayashi. Kriging model based many-objective optimization with efficient calculation of hypervolume improvement. In *In proceedings of IEEE Congress on Evolutionary Computation*, pages 1187–1194. IEEE, 2014.

[7] R. Cheng, Y. Jin, K. Narukawa, and B. Sendhoff. A multiobjective evolutionary algorithm using gaussian process-based inverse modeling. *IEEE Transactions on Evolutionary Computation*, 19:838–856, 2015.

[8] R. Cheng, Y. Jin, M. Olhofer, and B. Sendhoff. A reference vector guided evolutionary algorithm for many objective optimization. *IEEE Transactions on Evolutionary Computation*, 2016 (accepted).

[9] T. Chugh, K. Sindhya, J. Hakanen, and K. Miettinen. Handling computationally expensive multiobjective optimization problems with evolutionary algorithms-A survey. Technical Report Series B, Scientific Computing No. B 4/2015, Department of Mathematical Information Technology, University of Jyvaskyla, 2015.

[10] C. Coello, G. Lamont, and D. Veldhuizen. *Evolutionary Algorithms for Solving Multi-objective Problems*. Springer, New York, 2nd edition, 2007.

[11] J. Cornell. *Experiments with Mixtures: Designs, Models, and the Analysis of Mixture Data*. John Wiley & Sons, 2011.

[12] I. Couckuyt, D. Deschrijver, and T. Dhaene. Fast calculation of multiobjective probability of improvement and expected improvement criteria for Pareto optimization. *Journal of Global Optimization*, 60:575–594, 2014.

[13] K. Deb. *Multi-Objective Optimization using Evolutionary Algorithms*. Wiley, Chichester, 2001.

[14] K. Deb and H. Jain. An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: solving problems with box constraints. *IEEE Transactions on Evolutionary Computation*, 18:577–601, 2014.

[15] K. Deb, A. Prarap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6:182–197, 2002.

[16] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler. Scalable multi-objective optimization test problems. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pages 825–830. IEEE, 2002.

[17] A. G. Di Nuovo, G. Ascia, and V. Catania. A study on evolutionary multi-objective optimization with fuzzy approximation for computational expensive problems. In *Parallel Problem Solving from Nature*, volume LNCS 7492, pages 102–111. Springer, 2012.

[18] M. Emmerich, K. Giannakoglou, and B. Naujoks. Single- and multi-objective evolutionary optimization assisted by Gaussian random field metamodels. *IEEE Transactions on Evolutionary Computation*, 10:421–439, 2006.

[19] N. Hansen. The CMA evolution strategy: A comparing review , pp. 17691776. In J. Lozano, P. Larranaga, I. Inza, and E. Bengoetxea, editors, *Towards a New Evolutionary Computation*, pages 75–102. Springer Berlin Heidelberg, 2006.

[20] J. Hensman, N. Fusi, and N. Lawrence. Gaussian processes for big data. In *Conference on Uncertainty in Artificial Intelligence*, 2013.

[21] R. Hooke and T. Jeeves. Direct search solution of numerical and statistical problems. *Journal of the Association for Computing Machinery (ACM)*, 8:212–229, 1961.

[22] S. Huband, L. Barone, L. While, and P. Hingston. A scalable multi-objective test problem toolkit. In C. Coello, A. H. Aguirre, and E. Zitzler, editors, *Evolutionary Multi-Criterion Optimization*, pages 280–295. Springer, 2005.

[23] H. Ishibuchi, N. Tsukamoto, and Y. Nojima. Evolutionary many-objective optimization: A short review. In *Proceedings of IEEE Congress on Evolutionary Computation*, pages 2419–2426. IEEE, 2008.

[24] Y. Jin. Surrogate-assisted evolutionary computation: Recent advances and future challenges. *Swarm and Evolutionary Computation*, 1(2):61–70, 2011.

[25] Y. Jin, M. Olhofer, and B. Sendhoff. On evolutionary optimization with approximate fitness functions. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 786–793. Morgan Kaufmann, 2000.

[26] Y. Jin, M. Olhofer, and B. Sendhoff. A framework for evolutionary optimization with approximate fitness functions. *IEEE Transactions on Evolutionary Computation*, 6:481–494, 2002.

[27] Y. Jin and B. Sendhoff. Reducing fitness evaluations using clustering techniques and neural network ensembles. In K. Deb, editor, *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 688–699. Springer, Berlin, Heidelberg, 2004.

[28] J. Knowles. ParEGO: a hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems. *IEEE Transactions on Evolutionary Computation*, 10:50–66, 2006.

[29] B. Li, J. Li, K. Tang, and X. Yao. Many-objective evolutionary algorithms: A survey. *ACM Computing Surveys*, 48:13–35, 2015.

[30] S. Lophaven, H. Nielsen, and J. Sondergaard. DACE: a MATLAB Kriging toolbox. Technical report, Technical University of Denmark, 2002.

[31] I. Loshchilov, M. Schoenauer, and M. Sebag. Dominance-based Pareto-surrogate for multi-objective optimization. In K. Deb, A. Bhattacharya, N. Chakroborty, S. Das, J. Dutta, S. Gupta, A. Jain, V. Aggarwal, J. Branke, S. Louis, and K. Tan, editors, *Proceedings of the Simulated Evolution and Learning*, pages 230–239. Springer, Berlin, Heidelberg, 2010.

[32] M. Mckay, R. Beckman, and W. Conover. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 42:55–61, 2000.

[33] K. Miettinen. *Nonlinear Multiobjective Optimization*. Kluwer, Boston, 1999.

[34] A. Mogilicharla, T. Chugh, S. Majumder, and K. Mitra. Multi-objective optimization of bulk vinyl acetate polymerization with branching. *Materials and Manufacturing Processes*, 29:210–217, 2014.

[35] H. Nakayama, K. Inoue, and Y. Yoshimori. Approximate optimization using computational intelligence and its application to reinforcement of cable-stayed bridges. In *Proceedings of the Integrated Intelligent Systems for Engineering Design*, pages 289–304. IOS Press, 2006.

[36] K. Narukawa and T. Rodemann. Examining the performance of evolutionary many-objective optimization algorithms on a real-world application. In *Proceedings of Genetic and Evolutionary Computing*, pages 316–319. IEEE, 2012.

[37] J. Nelder and R. Mead. A simplex method for function minimization. *Computer Journal*, 7:308–313, 1965.

[38] M. Pilát and R. Neruda. ASM-MOMA: Multiobjective memetic algorithm with aggregate surrogate model. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pages 1202–1208. IEEE, 2011.

[39] M. Pilát and R. Neruda. Improving many-objective optimizers with aggregate meta-models. In *Proceedings of the 11th International Conference on Hybrid Intelligent Systems*, pages 555–560. IEEE, 2011.

[40] M. Pilat and R. Neruda. Aggregate meta-models for evolutionary multi-objective and many-objective optimization. *Neurocomputing*, 116:392–402, 2013.

[41] W. Ponweiser, T. Wagner, D. Biermann, and M. Vincze. Multiobjective optimization on a limited budget of evaluations using model-assisted S-metric selection. In G. Rudolph, T. Jansen, S. Lucas, C. Poloni, and N. Beume, editors, *Proceedings of the Parallel Problem Solving from Nature-PPSN X*, pages 784–794. Springer, Berlin, Heidelberg, 2008.

[42] R. Purshouse, K. Deb, M. Mansor, S. Mostaghim, and R. Wang. A review of hybrid evolutionary multiple criteria decision making methods. In *Proceedings of IEEE Congress on Evolutionary Computation*, pages 1147–1154. IEEE, 2014.

[43] L. Santana-Quintero, A. Montano, and C. Coello. A review of techniques for handling expensive functions in evolutionary multi-objective optimization. In Y. Tenne and C.-K. Goh, editors, *Computational Intelligence in Expensive Optimization Problems*, pages 29–59. Springer, Berlin, Heidelberg, 2010.

[44] R. Storn and K. Price. Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, 11:341–359, 1997.

[45] C. von Lücken, B. Barán, and C. Brizuela. A survey on multi-objective evolutionary algorithms for many-objective problems. *Computational Optimization and Applications*, 58:707–756, 2014.

[46] T. Wagner. *Planning and Multi-Objective Optimization of Manufacturing Processes by Means of Empirical Surrogate Models*. PhD thesis, Schriftenreihe des ISF, Vulkan Verlag, Essen, 2013.

[47] T. Wagner, M. Emmerich, A. Deutz, and W. Ponweiser. On expected-improvement criteria for model-based multi-objective optimization. In R. Schaefer, C. Cotta, J. Kolodziej, and G. Rudolph, editors, *Proceedings of the Parallel Problem Solving from Nature-PPSN XI*, pages 718–727. Springer, Berlin, Heidelberg, 2010.

[48] Y. Yazawa, H. Aguirre, A. Oyama, and K. Tanaka. Evolutionary many-objective optimization optimization using ϵ-Hoods and chebyshev function. In *In the proceedings of IEEE Congress on Evolutionary Computation*, pages 1861–1868. IEEE, 2015.

[49] Y. Yuan, H. Xu, B. Wang, and B. Zhang. Balancing convergence and diversity in decomposition-based many-objective optimizers. *IEEE Transactions on Evolutionary Computation*, 2016 (accepted).

[50] Q. Zhang and H. Li. MOEA/D: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, 11:712–731, 2007.

[51] Q. Zhang, W. Liu, E. Tsang, and B. Virginas. Expensive multiobjective optimization by MOEA/D with Gaussian process model. *IEEE Transactions on Evolutionary Computation*, 14:456–474, 2010.

[52] X. Zhang, Y. Tian, and Y. Jin. A knee point-driven evolutionary algorithm for many-objective optimization. *IEEE Transactions on Evolutionary Computation*, 19:761–776, 2015.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2016.2622301, IEEE Transactions on Evolutionary Computation

Tinkle Chugh is a PhD student in the Industrial Optimization Group within Faculty of Information Technology, University of Jyvaskyla, Finland. He received his Master of Technology degree in Chemical Engineering from Indian Institute of Technology Hyderabad, India, in 2012. His research interests include evolutionary computation, surrogate-assisted multi-/many- objective optimization, and machine learning.

Yaochu Jin (M'98-SM'02-F'16) received the B.Sc., M.Sc., and Ph.D. degrees from Zhejiang University, Hangzhou, China, in 1988, 1991, and 1996 respectively, and the Dr.-Ing. degree from Ruhr University Bochum, Germany, in 2001.

He is a Professor in Computational Intelligence, Department of Computer Science, University of Surrey, Guildford, U.K., where he heads the Nature Inspired Computing and Engineering Group. He is also a Finland Distinguished Professor, University of Jyvaskyla, Finland and a Changjiang Distinguished Professor, Northeastern University, China. His main research interests include evolutionary computation, machine learning, computational neuroscience, and evolutionary developmental systems, with their application to data-driven optimization and decision-making, self-organizing swarm robotic systems, and bioinformatics. He has (co)authored over 200 peer-reviewed journal and conference papers and has been granted eight patents on evolutionary optimization.

Dr Jin is the Editor-in-Chief of the IEEE TRANSACTIONS ON COGNITIVE AND DEVELOPMENTAL SYSTEMS and Complex & Intelligent Systems. He was an IEEE Distinguished Lecturer (2013-2015) and Vice President for Technical Activities of the IEEE Computational Intelligence Society (2014-2015). He was the recipient of the Best Paper Award of the 2010 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology and the 2014 IEEE Computational Intelligence Magazine Outstanding Paper Award. He is a Fellow of IEEE.

Kaisa Miettinen is Professor of Industrial Optimization and vice-rector of the University of Jyvaskyla. She received her Ph.D. and M.Sc. degrees at the Department of Mathematics of the University of Jyvaskyla in 1994 and 1988, respectively. Her research interests include theory, methods, applications and software of nonlinear multiobjective optimization including interactive and evolutionary approaches and she heads the Research Group on Industrial Optimization. She has authored about 150 refereed journal, proceedings and collection papers, edited 13 proceedings, collections and special issues and written a monograph Nonlinear Multiobjective Optimization. She is a member of the Finnish Academy of Science and Letters, Section of Science and the Immediate-Past President of the International Society on Multiple Criteria Decision Making. She belongs to the editorial boards of six international journals, the Steering Committee of Evolutionary Multi-Criterion Optimization and is a member of the IEEE Computational Intelligence Society Task Force Group on Multiobjective Evolutionary Algorithms. She has worked at IIASA, International Institute for Applied Systems Analysis in Austria, KTH Royal Institute of Technology in Stockholm, Sweden and at Helsinki School of Economics in Finland.

Jussi Hakanen is a senior researcher with the Faculty of Information Technology at the University of Jyvaskyla, Finland. He received MSc degree in mathematics and PhD degree in mathematical information technology, both from the University of Jyvaskyla. His research is focused on multiobjective optimization with an emphasis on interactive multiobjective optimization methods and computationally expensive problems. He has participated in several industrial projects involving different applications of multiobjective optimization, e.g. in chemical engineering. He has been a visiting researcher in Cornell University, Carnegie Mellon, University of Surrey, University of Wuppertal, University of Malaga and the VTT Technical Research Center of Finland. He also has a title of docent in industrial optimization at the University of Jyvaskyla.

Karthik Sindhya is a post-doctoral researcher with the Faculty of Information Technology, University of Jyvaskyla, Finland. He has a bachelor's and master's degree in chemical engineering and a PhD in the area of multi-criteria optimisation. His PhD thesis on hybrid algorithms was nominated as one of the top three dissertations in multiple criteria decision making during the period of 2010-2013. He was also the recipient of the "Best algorithm award" at IEEE CEC 2007 and best poster award at MCDM 2011 conferences. His research interests include evolutionary algorithms especially evolutionary multi-objective optimisation (EMO) algorithms, handling computationally expensive objective functions that arise in industry and multiple criteria decision making approaches.

Supplementary material - A Surrogate-assisted Reference Vector Guided Evolutionary Algorithm for Computationally Expensive Many-objective Optimization

Tinkle Chugh, Yaochu Jin, *Fellow, IEEE*, Kaisa Miettinen, Jussi Hakanen, Karthik Sindhya

In the supplementary material, we provide the parameter values for the number of reference vectors, size of the reference set to calculate IGD, results on DTLZ problems using hypervolume and on WFG problems with both IGD and hypervolume as quality metrics. In addition, results of K-RVEA, RVEA, ParEGO and MOEA/D-EGO on a free radical polymerization system are also presented. Moreover, a sensitivity analysis of three important parameters used in K-RVEA is also provided.

I. Number of reference vectors

The number of reference vectors (N) in K-RVEA is determined by the design factors (H_1, H_2) for a simplex-lattice design [2] and the number of objectives (k). The parameter values used (from [1]) are listed in Table I for the different numbers of objectives.

Table I: Numbers (N) of reference vectors

k	$(H1, H2)$	N
3	(13,0)	105
4	(7,0)	120
6	(4,1)	132
8	(3,2)	156
10	(3,2)	275

II. Size of reference set to calculate IGD

In this paper, we use a different size of a reference set for different numbers of objectives to calculate IGD values and the sizes are presented in Table II. Test problems DTLZ7 and WFG2 have disconnected Pareto fronts and, therefore the size of the reference set is different from the other problems. In addition, specific number like 10000 for four objectives cannot be generated with the method used [4] for reference set generation and for this reason the number closest to 5000, 10000, 30000, 50000 and 90000 is used for different numbers of objectives.

Table II: Size of reference set to calculate IGD

k	number of reference points		
	DTLZ7	WFG2	Other problems
3	6084	4101	5050
4	10648	10708	10660
6	59049	32191	33649
8	78125	66342	50388
10	262144	115610	92378

III. Performance on the DTLZ suite

Here, we summarize the performances of the different algorithms compared with hypervolume as a performance metric. We used the worst objective function values from non-dominated solutions of all algorithms plus a small threshold as a reference point (f_i^*). For less than eight objectives, a recently proposed method from [7] was used to calculate the hypervolume. For eight and 10 objectives, the Monte Carlo method with 1000000 sample points was used to approximate the hypervolume. The hypervolume values obtained were normalized by dividing with $\prod_{i=1}^{k} f_i^*$ and are shown in Table III.

As can be seen, the performances of different algorithms when measured with the hypervolume are very similar to those with IGD values and overall, K-RVEA performed better than the other algorithms. Note that due to the degenerated Pareto front of DTLZ5, most of the reference vectors in both RVEA and K-RVEA are empty and, therefore, the performance of K-RVEA is worse than that of ParEGO. Results with K-RVEA and MOEA/D-EGO for problems with three objectives are given in Table IV. As can be seen, K-RVEA performed either better or equivalently to MOEA/D-EGO except on DTLZ5. Note that we did not test SMS-EGO here due to its computational overhead, which is also mentioned in the main paper.

Table IV: Statistical results for hypervolume on the DTLZ suite obtained by K-RVEA and MOEA/D-EGO for three objectives. The best results are highlighted

Problem	**K-RVEA**				MOEA/D-EGO		
	min	mean	max		min	mean	max
DTLZ1	**0.319**	**0.539**	**0.719**	↑	0.034	0.151	0.330
DTLZ2	0.881	**0.905**	**0.918**	≈	**0.887**	0.905	0.916
DTLZ3	**0.200**	**0.443**	**0.695**	↑	0.056	0.117	0.234
DTLZ4	**0.811**	**0.887**	**0.947**	≈	0.807	0.872	0.905
DTLZ5	0.728	0.752	0.769	↓	**0.752**	**0.766**	**0.774**
DTLZ6	**0.814**	**0.850**	0.874	≈	0.579	0.737	**0.932**
DTLZ7	**0.455**	**0.467**	**0.476**	↑	0.152	0.280	0.473

IV. Performance on the WFG suite

To be able to show the potential of K-RVEA with different problems, we extended the test set up by also considering the WFG test suite [3]. Besides as in, some of the DTLZ problems the number of decision variables for driving the convergence decreases with the increase in the number of objectives, it is important to test with other types of problems as well. Values of two different types of parameters i.e. position (d)

Table III: Statistical results for hypervolume on the DTLZ suite obtained by K-RVEA, RVEA and ParEGO. The best results are highlighted

Problem	k	K-RVEA				RVEA				ParEGO		
		min	mean	max		min	mean	max		min	mean	max
DTLZ1	3	0.319	0.539	0.719	≈	0.416	0.595	0.749	↓	**0.490**	**0.805**	**0.941**
	4	0.588	0.795	0.911	↑	**0.641**	0.738	0.891	≈	0.595	**0.797**	**0.934**
	6	0.828	0.927	**0.973**	≈	**0.845**	**0.930**	0.964		NA	NA	NA
	8	**0.781**	**0.879**	**0.929**	≈	0.642	0.870	0.929		NA	NA	NA
	10	**0.374**	0.389	**0.393**	≈	0.366	**0.389**	0.393		NA	NA	NA
DTLZ2	3	**0.881**	**0.905**	**0.918**	↑	0.782	0.824	0.858	↑	0.870	0.889	0.902
	4	**0.924**	**0.958**	**0.970**	↑	0.894	0.920	0.939	↑	0.890	0.907	0.926
	6	**0.327**	**0.328**	**0.328**	↑	0.306	0.318	0.325		NA	NA	NA
	8	**0.225**	**0.225**	**0.226**	↑	0.208	0.219	0.223		NA	NA	NA
	10	**0.134**	**0.134**	**0.134**	↑	0.127	0.130	0.132		NA	NA	NA
DTLZ3	3	0.200	0.443	0.695	≈	0.279	0.479	0.651	↓	**0.561**	**0.804**	**0.935**
	4	0.810	0.883	0.964	≈	0.731	0.851	0.932	↓	**0.883**	**0.947**	**0.981**
	6	**0.970**	**0.992**	**0.996**	↑	0.958	0.983	0.991		NA	NA	NA
	8	**0.975**	**0.989**	0.992	≈	0.972	0.988	**0.992**		NA	NA	NA
	10	0.959	**0.961**	0.961	≈	**0.959**	0.960	**0.961**		NA	NA	NA
DTLZ4	3	**0.811**	0.887	**0.947**	≈	0.620	**0.889**	0.938	↑	0.630	0.830	0.929
	4	0.905	0.972	0.990	↓	**0.948**	**0.982**	**0.990**	↑	0.824	0.931	0.979
	6	**0.356**	0.361	0.364	≈	0.356	0.363	**0.364**		NA	NA	NA
	8	0.245	0.245	**0.246**	≈	**0.245**	**0.246**	0.246		NA	NA	NA
	10	**0.134**	0.134	**0.134**	≈	0.134	**0.134**	0.134		NA	NA	NA
DTLZ5	3	0.728	0.752	0.769	↑	0.629	0.678	0.717	↓	**0.749**	**0.801**	**0.829**
	4	0.795	0.819	0.829	↑	0.748	0.772	0.805	↓	**0.807**	**0.924**	**0.974**
	6	**0.191**	**0.195**	**0.197**	↑	0.172	0.185	0.193		NA	NA	NA
	8	**0.087**	**0.088**	**0.090**	↑	0.072	0.084	0.087		NA	NA	NA
	10	**0.031**	**0.031**	**0.031**	↑	0.025	0.027	0.030		NA	NA	NA
DTLZ6	3	**0.814**	**0.850**	**0.874**	↑	0.477	0.549	0.660	↑	0.399	0.423	0.500
	4	**0.891**	0.907	0.932	↑	0.668	0.733	0.779	≈	0.572	**0.978**	**0.991**
	6	**0.534**	**0.550**	**0.564**	↑	0.414	0.441	0.478		NA	NA	NA
	8	**0.252**	**0.272**	**0.288**	↑	0.202	0.230	0.259		NA	NA	NA
	10	**0.069**	**0.077**	**0.079**	↑	0.045	0.066	0.076		NA	NA	NA
DTLZ7	3	**0.455**	**0.467**	**0.476**	↑	0.260	0.316	0.347	↑	0.186	0.218	0.244
	4	**0.455**	**0.483**	**0.508**	↑	0.250	0.318	0.377	↑	0.109	0.162	0.228
	6	**0.465**	**0.543**	**0.572**	↑	0.148	0.294	0.410		NA	NA	NA
	8	**0.343**	**0.416**	**0.480**	↑	0.085	0.198	0.344		NA	NA	NA
	10	**0.317**	**0.370**	**0.440**	↑	0.036	0.085	0.151		NA	NA	NA

and distance (l) used in WFG problems with the number of objectives are shown in Table V.

Table V: Numbers of parameters in WFG problems

k	d	l
3	8	2
4	4	6
6	4	5
8	2	7
10	2	9

Results with K-RVEA, RVEA and ParEGO using IGD and hypervolume are provided in Tables VI and VII, respectively. We used the same settings for the size of the reference set to calculate IGD and for the reference point to calculate hypervolume as mentioned in the previous sections. The problem WFG2 has a disconnected Pareto front and it is interesting to note that K-RVEA handled problems with disconnected Pareto front well and outperformed others with both hypervolume and IGD performance metrics. However in WFG3, the degenerated Pareto front caused the most of the reference vectors to be empty, thereby leading to a bad performance. Problems WFG4-WFG9 possess several challenges to the optimization algorithm in the decision space such as multimodality for WFG4, landscape deception for WFG5 and non-separability for WFG6, WFG8 and WFG9 and as can be seen from the results, K-RVEA performed better than the other algorithms. Results with K-RVEA and MOEA/D-EGO for three objectives are provided in Tables VIII and IX, respectively. For these problems as well, K-RVEA performed similarly or better than MOEA/D-EGO.

Table VIII: Statistical results for IGD values on the WFG suite obtained by K-RVEA and MOEA/D-EGO for three objectives. The best results are highlighted

Problem	K-RVEA				MOEA/D-EGO		
	min	mean	max		min	mean	max
WFG1	**1.543**	**1.842**	**2.012**	≈	1.572	1.975	2.237
WFG2	**0.473**	**0.634**	**0.808**	≈	0.504	0.705	0.837
WFG3	**0.442**	**0.497**	**0.560**	↑	0.518	0.589	0.659
WFG4	**0.363**	**0.566**	**0.687**	↑	0.512	0.555	0.643
WFG5	**0.349**	**0.514**	**0.630**	↑	0.607	0.646	0.686
WFG6	**0.688**	**0.809**	**0.891**	↑	0.794	0.832	0.872
WFG7	0.624	0.669	0.740	≈	**0.617**	**0.653**	**0.715**
WFG8	**0.647**	**0.822**	0.934	≈	0.787	0.844	**0.868**
WFG9	**0.453**	**0.651**	0.885	↑	0.698	0.796	**0.851**

V. Performance on free-radical polymerization of Vinyl acetate

In the free radical polymerization of Vinyl acetate, the main aim is to find the optimal process conditions to obtain a polymer with specific properties. A high molecular weight is usually linked with a high melt strength and a low melt flow index [6]. In addition, a homogeneous distribution of molecular mass polymer is required in short processing time. Polyvinyl acetate has a high molecular weight and often known as wood glue and processed in a batch reactor. This problem

Table VI: Statistical results for IGD values on the WFG suite obtained by K-RVEA, RVEA and ParEGO. The best results are highlighted

Problem	k	K-RVEA				RVEA				ParEGO		
		min	mean	max		min	mean	max		min	mean	max
WFG1	3	**1.543**	1.842	**2.012**	↑	1.775	2.108	2.394	≈	1.697	**1.798**	2.318
	4	**0.940**	2.113	2.504	≈	1.934	**2.018**	**2.225**	≈	1.982	2.137	2.517
	6	**2.394**	**2.645**	**2.958**	↑	2.683	2.822	3.298		NA	NA	NA
	8	**2.699**	**2.941**	**3.403**	↑	2.896	3.074	3.755		NA	NA	NA
	10	**2.990**	**3.255**	**3.781**	↑	3.275	3.529	4.162		NA	NA	NA
WFG2	3	**0.473**	**0.634**	**0.808**	↑	0.595	0.771	0.903	↑	0.652	0.777	0.872
	4	**0.483**	**0.651**	**0.903**	↑	0.601	0.831	1.031	↑	0.914	1.098	1.407
	6	**0.497**	**0.667**	**1.501**	↑	0.919	1.191	1.897		NA	NA	NA
	8	**0.658**	**0.947**	**1.798**	↑	1.040	1.727	2.940		NA	NA	NA
	10	**1.178**	**1.616**	**2.133**	↑	1.595	2.393	3.892		NA	NA	NA
WFG3	3	0.442	0.497	**0.560**	↑	0.557	0.681	1.065	≈	**0.426**	**0.477**	0.568
	4	**0.226**	0.509	0.757	↑	0.440	0.621	0.828	↓	0.368	**0.447**	**0.512**
	6	**0.350**	**0.668**	**0.881**	↑	1.094	1.486	2.572		NA	NA	NA
	8	**0.454**	**0.641**	**0.836**	↑	1.218	1.793	2.793		NA	NA	NA
	10	**0.581**	**0.883**	**1.108**	↑	1.665	3.465	6.425		NA	NA	NA
WFG4	3	**0.363**	**0.566**	0.687	≈	0.533	0.582	**0.657**	≈	0.509	0.569	0.658
	4	**0.622**	**1.064**	1.812	↑	1.066	1.243	**1.527**	↑	1.175	1.494	2.039
	6	**1.527**	**1.632**	**1.852**	↑	2.289	2.632	3.302		NA	NA	NA
	8	**2.749**	**3.080**	**3.861**	↑	4.348	5.275	8.286		NA	NA	NA
	10	**4.257**	**6.790**	**9.212**	↑	6.487	7.741	9.607		NA	NA	NA
WFG5	3	**0.349**	**0.514**	**0.630**	↑	0.589	0.699	0.794	↑	0.586	0.684	0.729
	4	**0.729**	**0.973**	**1.180**	↑	1.016	1.171	1.351	↑	1.120	1.301	1.493
	6	**1.630**	**1.763**	**1.901**	↑	2.211	2.357	2.655		NA	NA	NA
	8	**2.695**	**2.955**	**3.374**	↑	4.145	4.496	5.106		NA	NA	NA
	10	**4.363**	**5.606**	**6.743**	↑	5.895	6.459	7.401		NA	NA	NA
WFG6	3	0.688	0.809	**0.891**	≈	0.758	0.841	0.892	≈	**0.631**	**0.790**	0.942
	4	**0.814**	**1.101**	**1.393**	↑	1.110	1.253	1.410	↑	1.236	1.330	1.418
	6	**1.740**	**1.921**	**2.141**	↑	2.318	2.460	2.802		NA	NA	NA
	8	**3.005**	**3.327**	**3.693**	↑	4.107	4.576	5.488		NA	NA	NA
	10	**4.953**	**5.535**	**6.381**	↑	5.920	6.414	7.163		NA	NA	NA
WFG7	3	0.624	0.669	0.740	≈	0.598	0.667	0.729	≈	**0.563**	**0.645**	0.716
	4	**0.846**	**1.291**	1.814	↑	1.222	1.429	1.910	↑	1.377	1.558	**1.792**
	6	**1.644**	**1.790**	**1.958**	↑	2.381	2.644	3.432		NA	NA	NA
	8	**3.071**	**3.352**	**3.989**	↑	4.519	5.045	5.916		NA	NA	NA
	10	**5.709**	**7.240**	9.140	≈	6.649	7.433	**9.046**		NA	NA	NA
WFG8	3	**0.647**	**0.822**	0.934	↑	0.802	0.893	0.991	≈	0.745	0.843	**0.899**
	4	**1.211**	**1.437**	**1.614**	↑	1.561	1.684	1.876	↑	1.619	1.786	1.954
	6	**2.073**	**2.156**	**2.274**	↑	2.584	2.818	3.167		NA	NA	NA
	8	**3.275**	**3.491**	**3.672**	↑	4.708	5.149	6.108		NA	NA	NA
	10	**5.315**	**5.912**	**6.525**	↑	6.269	7.056	7.998		NA	NA	NA
WFG9	3	**0.453**	0.651	0.885	↑	0.701	0.822	0.953	≈	0.490	**0.640**	**0.869**
	4	**0.693**	**1.086**	**1.434**	↑	1.210	1.349	1.506	↑	0.985	1.232	1.448
	6	**1.686**	**1.949**	**2.849**	↑	2.321	2.568	3.352		NA	NA	NA
	8	**2.918**	**3.532**	**4.353**	↑	4.259	4.853	5.672		NA	NA	NA
	10	**4.469**	**5.893**	**7.690**	↑	6.164	7.209	9.154		NA	NA	NA

Table IX: Statistical results for hypervolume on the WFG suite obtained by K-RVEA and MOEA/D-EGO for three objectives. The best results are highlighted

Problem	**K-RVEA**				**MOEA/D-EGO**		
	min	mean	max		min	mean	max
WFG1	0.366	**0.429**	0.478	≈	**0.373**	0.421	**0.498**
WFG2	**0.732**	**0.782**	**0.852**	↑	0.718	0.751	0.795
WFG3	**0.524**	**0.564**	**0.608**	≈	0.517	0.550	0.571
WFG4	**0.561**	**0.603**	**0.662**	↑	0.517	0.559	0.591
WFG5	**0.530**	**0.604**	**0.656**	↑	0.478	0.525	0.572
WFG6	0.451	**0.490**	**0.555**	≈	**0.455**	0.463	0.467
WFG7	0.500	0.535	**0.582**	≈	**0.522**	**0.536**	0.558
WFG8	**0.491**	**0.525**	**0.558**	↑	0.443	0.451	0.459
WFG9	0.423	**0.523**	**0.634**	↑	**0.428**	0.456	0.496

is computationally expensive and an average computation time for one function evaluation is around 45 minutes. The reason for the high computation time is the slow reaction rate after a certain point of time which is termed as gelation. For more details about kinetics and a study of multiobjective optimization for this polymerization, see [5]. In this study, we optimize:

1) maximize average molecular weight M_w,
2) minimize polydispersity index *PDI* and
3) minimize polymerization time t_{poly}.

Decision variables or the process conditions are monomer concentration, initiator concentration, polymerization time and temperature. Bounds for these decision variables are given in Table X. We ran this process for 250 function evaluations with K-RVEA, RVEA, ParEGO and MOEA/D-EGO.

Table X: Bounds for decision variables

	Lower bound	**Upper bound**
Monomer concentration (mole/lt)	10	14
Initiator concentration (mole/lt)	3.00e-05	1.50e-04
Polymerization time (sec)	10	10000
Temperature (C)	60	80

Nondominated solutions from all four algorithms are shown in Figure 1. As can be seen, solutions from K-RVEA are better distributed than those of RVEA and ParEGO. Solutions from MOEA/D-EGO are also well distributed but some of them are

Table VII: Statistical results for hypervolume on the WFG suite obtained by K-RVEA, RVEA and ParEGO. The best results are highlighted

Problem	k	K-RVEA				RVEA				ParEGO		
		min	mean	max		min	mean	max		min	mean	max
WFG1	3	**0.366**	**0.429**	**0.478**	↑	0.352	0.411	0.468	↑	0.346	0.393	0.415
	4	0.390	0.437	**0.469**	≈	**0.437**	**0.452**	0.466	↑	0.260	0.321	0.377
	6	**0.337**	**0.378**	**0.404**	↑	0.293	0.345	0.375		NA	NA	NA
	8	**0.292**	**0.327**	0.352	≈	0.257	0.307	**0.438**		NA	NA	NA
	10	**0.276**	**0.302**	**0.334**	↑	0.219	0.261	0.293		NA	NA	NA
WFG2	3	**0.732**	**0.782**	**0.852**	↑	0.648	0.694	0.762	↑	0.712	0.761	0.806
	4	**0.650**	**0.738**	**0.896**	↑	0.553	0.673	0.806	↑	0.565	0.637	0.695
	6	**0.683**	**0.845**	**0.913**	↑	0.568	0.707	0.817		NA	NA	NA
	8	**0.765**	**0.900**	**0.941**	↑	0.383	0.697	0.897		NA	NA	NA
	10	**0.661**	**0.835**	0.920	↑	0.334	0.666	**0.927**		NA	NA	NA
WFG3	3	0.524	0.564	**0.608**	↑	0.452	0.500	0.541	≈	**0.568**	**0.577**	0.587
	4	0.527	**0.569**	**0.635**	↑	0.457	0.506	0.543	≈	**0.539**	0.561	0.573
	6	**0.565**	**0.598**	**0.667**	↑	0.448	0.495	0.565		NA	NA	NA
	8	**0.632**	**0.671**	**0.697**	↑	0.449	0.528	0.586		NA	NA	NA
	10	**0.578**	**0.612**	**0.650**	↑	0.259	0.406	0.481		NA	NA	NA
WFG4	3	**0.561**	**0.603**	**0.662**	↑	0.527	0.558	0.607	↑	0.533	0.577	0.620
	4	**0.621**	**0.689**	**0.725**	↑	0.432	0.500	0.564	↑	0.382	0.468	0.545
	6	**0.660**	**0.757**	**0.832**	↑	0.434	0.555	0.659		NA	NA	NA
	8	**0.657**	**0.767**	**0.857**	↑	0.265	0.465	0.586		NA	NA	NA
	10	**0.546**	**0.699**	**0.815**	↑	0.362	0.436	0.533		NA	NA	NA
WFG5	3	**0.530**	**0.604**	**0.656**	↑	0.438	0.479	0.521	↑	0.465	0.488	0.538
	4	**0.560**	**0.621**	**0.685**	↑	0.413	0.459	0.511	↑	0.407	0.448	0.522
	6	**0.573**	**0.675**	**0.746**	↑	0.413	0.485	0.552		NA	NA	NA
	8	**0.548**	**0.721**	**0.782**	↑	0.317	0.431	0.520		NA	NA	NA
	10	**0.542**	**0.671**	**0.777**	↑	0.281	0.402	0.498		NA	NA	NA
WFG6	3	0.451	0.490	0.555	↑	0.403	0.436	0.478	↓	**0.520**	**0.560**	**0.608**
	4	**0.536**	**0.665**	**0.710**	↑	0.390	0.440	0.509	↑	0.428	0.461	0.514
	6	**0.543**	**0.676**	**0.774**	↑	0.400	0.471	0.545		NA	NA	NA
	8	**0.647**	**0.793**	**0.837**	↑	0.334	0.442	0.564		NA	NA	NA
	10	**0.667**	**0.749**	**0.836**	↑	0.288	0.411	0.542		NA	NA	NA
WFG7	3	0.500	0.535	0.582	↑	0.487	0.519	0.562	↓	**0.540**	**0.578**	**0.646**
	4	**0.436**	**0.559**	**0.661**	↑	0.379	0.446	0.527	↑	0.400	0.434	0.480
	6	**0.603**	**0.726**	**0.792**	↑	0.392	0.484	0.573		NA	NA	NA
	8	**0.573**	**0.703**	**0.810**	↑	0.378	0.455	0.564		NA	NA	NA
	10	**0.473**	**0.614**	**0.704**	↑	0.318	0.430	0.511		NA	NA	NA
WFG8	3	**0.491**	**0.525**	**0.558**	↑	0.379	0.426	0.471	↑	0.429	0.469	0.531
	4	**0.456**	**0.490**	**0.554**	↑	0.332	0.371	0.404	↑	0.345	0.388	0.438
	6	**0.501**	**0.540**	**0.606**	↑	0.314	0.398	0.478		NA	NA	NA
	8	**0.488**	**0.591**	**0.684**	↑	0.271	0.329	0.444		NA	NA	NA
	10	**0.444**	**0.561**	**0.665**	↑	0.262	0.326	0.403		NA	NA	NA
WFG9	3	0.423	0.523	0.634	↑	0.386	0.436	0.507	↓	**0.463**	**0.553**	**0.617**
	4	**0.437**	**0.512**	**0.664**	↑	0.400	0.451	0.507	≈	0.420	0.486	0.569
	6	**0.418**	**0.650**	**0.751**	↑	0.394	0.501	0.638		NA	NA	NA
	8	**0.544**	**0.692**	**0.792**	↑	0.299	0.456	0.531		NA	NA	NA
	10	**0.462**	**0.633**	**0.745**	↑	0.336	0.416	0.534		NA	NA	NA

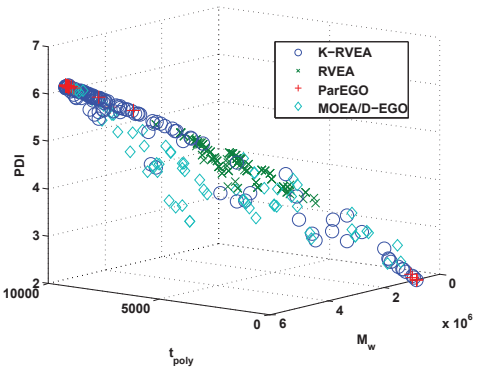


Figure 1: Nondominated solutions from K-RVEA, RVEA, ParEGO and MOEA/D-EGO

Table XI: IGD values obtained with K-RVEA, RVEA and ParEGO

	K-RVEA	RVEA	ParEGO	MOEA/D-EGO
IGD	**2.338e+04**	3.395e+05	7.4873e+05	4.0836e+04

dominated by those from K-RVEA. To compare the results statistically, we combined nondominated solutions from all four algorithms and clustered them into a prefixed number. One individual from each cluster closest to the centroid was selected and all selected individuals were used as the reference set to calculate IGD values. The IGD values thus obtained are given in Table XI. As can be seen, K-RVEA was able to obtain of a better quality solutions than the other algorithms in the given number of function evaluations.

VI. Sensitivity analysis of parameters in K-RVEA

We provide sensitivity analysis of three important parameters used in K-RVEA. The first analysis is for parameter δ used to select individuals based on the needs of convergence and diversity. The second is for the number of individuals (*Nu)* to be selected for updating the surrogates. The third is for the number of generations (w_{max}) used before updating the surrogates.

A. Effect of parameter δ

The parameter δ is effective whenever surrogates are updated. The main motivation for using δ is to reduce the number of empty reference vectors by exploiting the uncertainty information from Kriging models. In K-RVEA, when updating surrogates, the change in the number of empty reference vectors is measured from the previous update and if this change is more than δ, uncertainty is used. In other words, whenever surrogates are updated based on the uncertainty, the number of empty reference vectors is reduced. An increase in the value of δ will decrease the frequency of using uncertainty or decreasing the use of uncertainty from Kriging models will increase the number of empty reference vectors. To elaborate, we used different values of δ i.e. (0.05, 0.3, 0.5, 0.7, 1) $\times$ *N*, where *N* is the number of reference vectors used and measured the number of times uncertainty from Kriging models was used. Results on WFG1 with different numbers of objectives are provided in the first row of Figure 2, where *NK* denotes the frequency of using uncertainty. As can be seen, the increase in the value of δ decreases the frequency of using uncertainty from the Kriging models.

To see the effect on the reference vectors, we measured the change in the number of empty reference vectors (denoted by *NR*) whenever uncertainty was used. The Results are given in the second row of Figure 2 which shows the average change in the number of empty reference vectors (denoted by *NR* in the figure). For instance, if surrogates were updated with uncertainty five times in the solution process and the change in the number of empty reference vectors was (5, 10, 3, 1, 1), the average i.e. four is shown. Moreover, no change was observed at $\delta = 1$ because surrogates were never updated using uncertainty information. In addition, the change was always a positive integer whenever uncertainty information was used, which indicates that the number of empty reference vectors decreased.

To see the effect of δ on the performance of the algorithm, we also measured the hypervolume for different numbers of objectives and present the results in Figure 3. As can be seen, hypervolume decreases with the increase in the value of δ. This is due to the fact that the frequency of using uncertainty information from Kriging models decreases with the increase in the value of δ. We also observed the similar behavior for other problems. In addition, the effect of the parameter δ depends on the number of reference vectors and the problem solved. Using δ in an adaptive way is a future research topic, however, in the current study, we kept it fixed as 0.05 $\times N$.

B. Effect of the number of individuals to be selected to update surrogates

In K-RVEA, surrogates need to be updated by selecting some individuals for re-evaluation using the original functions. These individuals should be selected in such a way that both convergence and diversity are taken into account. The number of individuals to be selected can be important and mainly depends on the problem solved. We performed a sensitivity analysis with numbers 2, 5, 10, 20 and 30 on the problem WFG9 for different numbers of objectives and the results are given in Figure 4. As can be seen, for up to eight objectives, an increase in the number of individuals (denoted by *Nu*) decreased the performance of the algorithm. In contrast, with 10 objectives, the performance was improved with an increase in the number.

As the total number of function evaluations is set as a constant, increasing the number of individuals for updating the surrogate will decrease the frequency of updating the surrogates. For instance, if the total number of function evaluations is 300 and the numbers of individuals for updating the surrogates are 2 and 10, then surrogates are updated 150 and 30 times, respectively. In other words, the number of times surrogates are used with an evolutionary algorithm is bigger in case of a low number of individuals. Using a low number of individuals thus may be helpful to achieve a good approximation of the Pareto front. On the other hand, a low number may not be enough and individuals selected do not necessarily contribute to the performance of the surrogates. In contrast, using a high number will reduce the frequency of using surrogates with an evolutionary algorithm and may be helpful to improve the performance of the surrogates. Therefore, this parameter depends on the type of problem solved and should be adaptive. However, in this paper we kept it fixed as five.

C. Effect of the number of generations before updating the surrogate

The frequency of updating the surrogate or when to update the surrogate is very important in surrogate management, although, unfortunately, there is no solid theory for when to update the surrogates. We, therefore, present here empirical studies on the performance of K-RVEA given different prefixed numbers of generations (w_{max}) before the Kriging models are re-trained. We set $w_{max} = \{10, 20, \ldots, 100\}$. The IGD values of the solution sets obtained by K-RVEA with these different settings for solving DTLZ4 for 3, 4, 6, 8 and 10 objectives are shown in Figure 5. Note that the maximum number of function evaluations is set to 300. However, we examine the performance of K-RVEA for various numbers of function evaluations, which are chosen to be 150, 200, 250, and 300. As can be seen, the performance changes more dramatically with three or four objectives. Nevertheless, there is no clear trend in the change of performance with the change of the number of generations before updating the surrogate. However, consistently good performance is observed for different numners of objectives when this number is set to 20. Although a better performance might be obtained

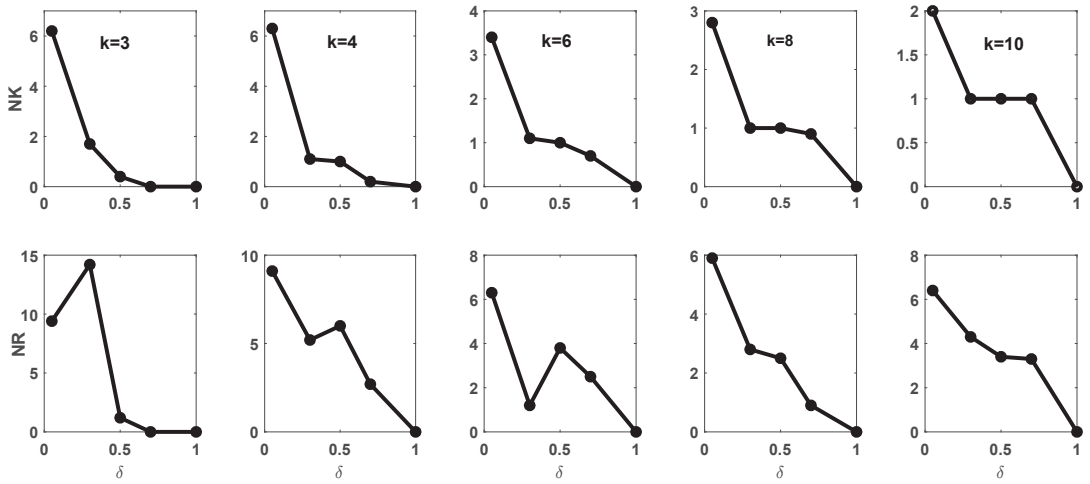


Figure 2: Effect of parameter δ used in K-RVEA on the frequency of using uncertainty information and changes in the number of empty reference vectors

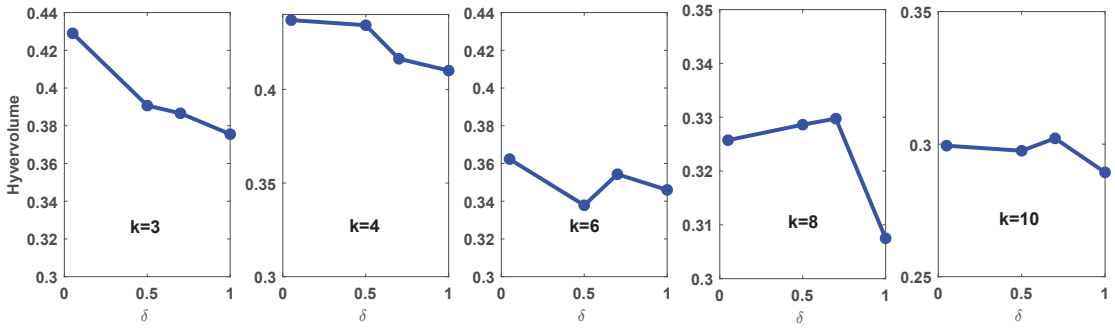


Figure 3: Effect of parameter δ used in K-RVEA on the hypervolume on WFG1

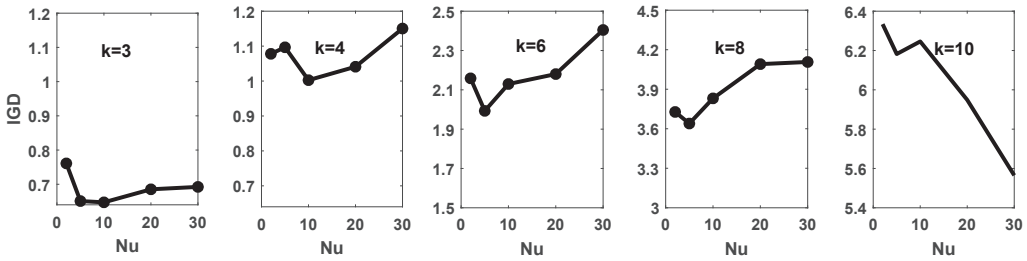


Figure 4: Performance of K-RVEA on WFG9 with the different numbers of individuals to be used for updating the surrogates

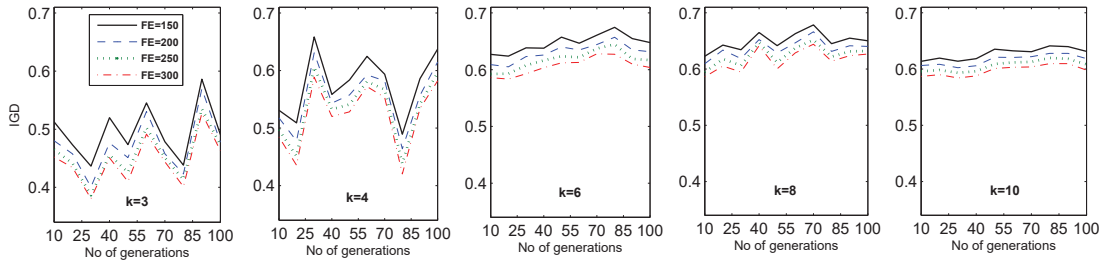


Figure 5: Performance of K-RVEA on DTLZ4 with the prefixed number of generations taken before updating the surrogate for different numbers of function evaluations.

when the number of generations before updating the surrogate is adapted, we set it to 20 in this work.

REFERENCES

[1] R. Cheng, Y. Jin, M. Olhofer, and B. Sendhoff. A reference vector guided evolutionary algorithm for many objective optimization. *IEEE Transactions on Evolutionary Computation*, 2016 (accepted).

[2] J. Cornell. *Experiments with Mixtures: Designs, Models, and the Analysis of Mixture Data*. John Wiley & Sons, 2011.

[3] S. Huband, L. Barone, L. While, and P. Hingston. A scalable multi-objective test problem toolkit. In C. Coello, A. H. Aguirre, and E. Zitzler, editors, *Evolutionary Multi-Criterion Optimization*, pages 280–295. Springer, 2005.

[4] K. Li, K. Deb, Q. Zhang, and S. Kwong. An evolutionary many-objective optimization algorithm based on dominance and decomposition. *IEEE Transactions on Evolutionary Computation*, 19:694–716, 2015.

[5] A. Mogilicharla, T. Chugh, S. Majumder, and K. Mitra. Multi-objective optimization of bulk vinyl acetate polymerization with branching. *Materials and Manufacturing Processes*, 29:210–217, 2014.

[6] S. Thomas. *Measurement and modelling of long chain branching*. PhD thesis, McMaster University, 1998.

[7] L. While, L. Bradstreet, and L. Barone. A fast way of calculating exact hypervolumes. *IEEE Transactions on Evolutionary Computation*, 16:86–95, 2012.

PIII

SURROGATE-ASSISTED EVOLUTIONARY MULTIOBJECTIVE SHAPE OPTIMIZATION OF AN AIR INTAKE VENTILATION SYSTEM

by

Tinkle Chugh, Karthik Sindhya, Kaisa Miettinen, Yaochu Jin, Tomas Kratky, Pekka Makkonen

In Proceedings of IEEE Congress on Evolutionary Computation, IEEE, to appear

Surrogate-assisted evolutionary multiobjective shape optimization of an air intake ventilation system

Tinkle Chugh, Karthik Sindhya, Kaisa Miettinen, Yaochu Jin
University of Jyvaskyla
Faculty of Information Technology
FI-40014 University of Jyvaskyla, Finland
Email: first-name.last-name@jyu.fi

Tomas Kratky
Centre of Hydraulic Research
Lutin, Czech Republic
Email: t.kratky@sigma.cz

Pekka Makkonen
Valtra Inc.,
Suolahti, Finland
Email: pekka.makkonen
@agcocorp.com

***Abstract*—We tackle three different challenges in solving a real-world industrial problem: formulating the optimization problem, connecting different simulation tools and dealing with computationally expensive objective functions. The problem to be optimized is an air intake ventilation system of a tractor and consists of three computationally expensive objective functions. We describe the modeling of the system and its numerical evaluation with a commercial software. To obtain solutions in few function evaluations, a recently proposed surrogate-assisted evolutionary algorithm K-RVEA is applied. The diameters of four different outlets of the ventilation system are considered as decision variables. From the set of nondominated solutions generated by K-RVEA, a decision maker having substance knowledge selected the final one based on his preferences. The final selected solution has better objective function values compared to the baseline solution of the initial design. A comparison of solutions with K-RVEA and RVEA (which does not use surrogates) is also performed to show the potential of using surrogates.**

I. Introduction

Optimization problems involving multiple conflicting objectives are common in engineering design. Typically they have multiple optimal solutions referred to as Pareto optimal solutions with different trade-offs among objectives. A decision maker (DM) who is an expert in the application domain can determine one among them as the final solution based on her/his preference information.

In this article, we focus on addressing three major challenges which often occurr in solving real-world problems:

1) the formulation of the multiobjective optimization problem,
2) modeling of the problem and connecting different pieces of commercial simulation software for numerical evaluation and
3) dealing with computationally expensive objective functions.

The formulation of a real-world optimization problem is usually an iterative task between an expert of the problem domain and an analyst who is familiar with optimization algorithms. In the literature, most of the algorithms developed are tested on benchmark problems, where the formulation of the problem is already known. In case of real-world problems, a meaningful formulation of the optimization problem is not necessarily straightforward and may need several iterations and efforts to be able to verify the appropriateness of the formulation.

We focus on multiobjective shape optimization of a component in the air intake ventilation system of a tractor. The particular component, shown in Figure 1, is used to heat the cabin and defrost the windscreen. Here, the main goal of the DM is to find a design of a cooling pipe ventilation system, where the outflow of air among all outlets is the same and the pressure loss is as low as possible. For considering these conflicting goals, we formulate a multiobjective optimization problem (MOP) involving three objectives. We describe different phases of formulating the objectives and the ones that eventually were selected.

After formulating the optimization problem, the next challenge is to combine different pieces of software to obtain objective function values. In contrast to benchmark problems, where analytical forms of the objective functions are available, real-world problems can have a black-box nature and need different simulation tools or software for function evaluation. For instance, in this study, we do meshing of the geometry in one software and then export it to another one for numerical evaluation. The output generated with the numerical evaluation software is connected with the optimization algorithm. Therefore, the challenge of combining different pieces of software is relevant in real-world problems and usually not addressed in the literature in using evolutionary algorithms.

The next challenge is finding a solution with a limited computation budget. The computation time to find a solution in real-world problems like problems involving computational fluid dynamics (CFD) simulations can be substantial [1]. In such cases, one can afford only few function evaluations.

The problem considered in this article is computationally expensive and involves CFD simulations. For this problem, the computation time for one function evaluation on a computer with Intel Xeon CPU E5-1607 v3 and 32 GB RAM is three to five minutes. To address the challenge of a limited computation budget, one can employ surrogates (or metamodels or response surface approximations) with evolutionary algorithms. Several algorithms have been proposed in the literature, e.g. ParEGO [2], SMS-EGO [3] and MOEA/D-EGO [4] to obtain a set of solutions in few function evaluations. For more details about these algorithms and their main characteristics, see [5], [6].

978-1-5090-4601-0/17/$31.00

Recently, Chugh et al. [7] have proposed a surrogate-assisted algorithm called K-RVEA for computationally expensive optimization problems involving three or more objective functions and demonstrated its efficiency. Kriging models have been popular in solving computationally expensive MOPs [8], [9] and are also used in K-RVEA to alleviate the computational cost of evaluating objective functions. The main focus in K-RVEA is to efficiently manage the surrogates and reduce the training time of the surrogates. The algorithm uses a set of reference vectors for managing the surrogates to obtain a diverse set approximating Pareto optimal solutions. The potential of the algorithm was shown in [7] on several benchmark problems by comparing with the state-of-the-art surrogate-assisted evolutionary algorithms.

We divide the consideration of the shape optimization problem into three phases. In the first phase, we focus on formulating the multiobjective optimization problem. In the next phase, we apply K-RVEA to find a representative set of Pareto optimal solutions. In the final phase, the DM who is the design expert in the automobile industry selects one solution of the set based on his preferences after a careful investigation.

The rest of this article is organized as follows. In Section II, we introduce some relevant characteristics of the air intake ventilation system in a tractor. Next, in Section III, we describe details and challenges occurred in formulating the optimization problem. In Section IV, we describe the modeling and numerical evaluation of objective functions with two pieces of commercial software. We briefly outline the K-RVEA algorithm used for solving the problem in Section V and present the results obtained in Section VI. Finally, in Section VII we draw conclusions.

II. Air intake ventilation system

In this section, we introduce the use of an air intake ventilation system in a tractor. The air intake is a part of a ventilation system in a tractor cabin and plays a crucial role in maintaining a uniform temperature. As can be seen in Figure 1, the component of the system considered in this study consists of four outlets leading to different branches of the ventilation system. Each outlet has its own role in maintaining the temperature. In an ideal scenario, flow rates from all these outlets should be the same to provide the best uniformity in the temperature. However, maintaining an equal flow rate from each outlet is not possible in practice as each outlet has a different shape and consequently, has a different *hydraulic resistance*.

For any component, the *pressure loss* ΔP in a passage is defined as

$$\Delta P = \Delta P_{friction} + \Delta P_{local}, \tag{1}$$

where $\Delta P_{friction}$ is the pressure loss caused by the friction resistance and ΔP_{local} is the pressure loss caused by local resistance. The friction resistance $\Delta P_{friction}$ occurs because of a momentum transfer to the solid walls [10]. Its numerical value can be estimated by Darcy's empirical formula:

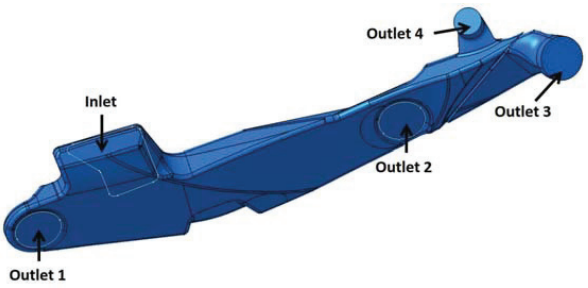


Fig. 1. CATIA 3D model of the component considered

$$\Delta P_{friction} = \bar{f}\frac{l}{D_H}\frac{\rho\bar{u}^2}{2}, \tag{2}$$

where $\bar{f}$ is the Moody friction factor, l and D_H are the length and the hydraulic diameter of the passage, respectively, ρ is the fluid density and $\bar{u}$ is the mean velocity of the flow.

The local resistance ΔP_{local} is due to the dissipation of mechanical energy (by whirls etc.), and can be described by the Weisbach formula:

$$\Delta P_{local} = \zeta\frac{\rho\bar{u}^2}{2}, \tag{3}$$

where ζ is the coefficient of local resistance.

In fact, the coefficients $\bar{f}$ and ζ are functions of both the passage *shape* and the *Reynold's number*. The dimensionless *Reynold's number* Re represents the ratio between momentum-related force and viscous shear force, and is defined in [11] as:

$$Re = \frac{uL\rho}{\eta}, \tag{4}$$

where u is the fluid velocity, ρ the fluid density, η the fluid viscosity and L the so-called *characteristic length*. The values of the *hydraulic resistance coefficients* $\bar{f}$ and ζ can be obtained by multiple means: experimentally or numerically through CFD. For many "standard" shapes, they can also be found in the literature, e.g. in [10].

Generally, the pressure loss increases with an increasing velocity. This is because of the quadratic member $\rho\bar{u}^2$ in (2) and (3). Now that some basic concepts of the ventilation system have been introduced, we can formulate the multiobjective optimization problem to be considered.

III. MOP formulation

In this section, we first discuss the challenges occurred in formulating the multiobjective optimization problem and then present the objective functions formulated.

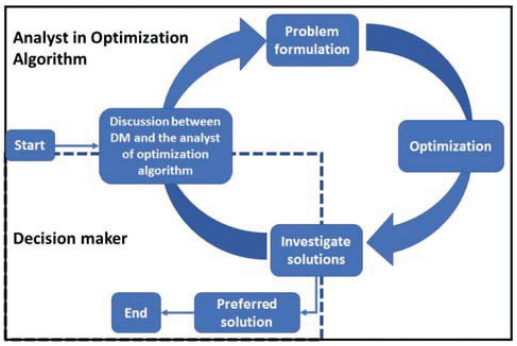


Fig. 2. An illustration of the optimization problem formulation in real-world problems

A. Challenges in formulating the problem

As mentioned in the introduction, formulating a meaningful representation of the optimization problem which reflects the needs and is understandable for the DM is not necessarily straightforward. It is very important to mention this particular challenge as most of the evolutionary algorithms in the literature are tested on benchmark problems, where no effort for the formulation of the optimization problem is needed. A typical illustration of the formulation and solving a real-wold optimization problem is presented in Figure 2. To summarize, in this study, first, a problem was formulated after an initial discussion with the expert who was the DM in the application domain and the optimization analyst. After solving the problem, the solutions were shown to the DM and the problem was revised and reformulated e.g. by adding, removing and modifying the objective functions. After these several iterations, a final formulation was selected.

We finalized the formulation after three iterations with the DM. First, we formulated a five objective optimization problem. Solutions obtained after solving this problem were not providing any extra value in terms of objective functions to the DM. Therefore, we modified the problem to have two objectives. Solutions of this biobjective optimization problem were meaningful to the DM and were also providing better values in terms of objective functions e.g. a good balance in the flow rates from different outlets and low pressure losses. However, the flow rate from one of the outlets was very low because of its small diameter. Therefore, we reformulated the problem to have three objectives by adding an extra objective considering the flow rate from the outlet with the smallest diameter. The DM was satisfied with the solutions and finally selected one solution based on his preferences. Next, we define the objective functions and decision variables of the final formulation of the multiobjective optimization problem.

B. Objective functions and decision variables

As mentioned in Section II, the pressure loss depends on hydraulic resistance and the mean velocity $\bar{u}$. All the outlets

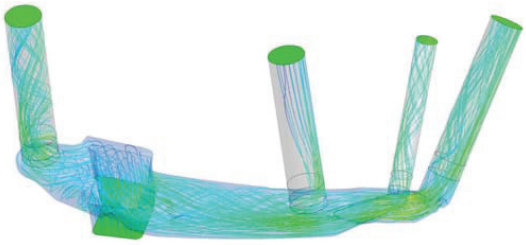


Fig. 3. CFD results of the initial design

open into a surrounding with the same pressure. Therefore, the pressure loss, i.e. the difference of pressures at inlet and outlet, must be the same for all the outlets. If the hydraulic resistances of different outlets are not the same, the velocities and thus the flow rates must be different according to equations (2) and (3). Also, flow rates from outlets 2 and 4 are significantly lower when compared to other outlets. This is mainly caused by the characteristic of the flow in case of outlet 2, and by the small diameter (small diameter leads to a high velocity) of outlet 4. A CFD simulation of the initial design given by the DM is shown in Figure 3. In the following, we denote the solution corresponding to the initial design as the baseline solution.

Maximizing the balance in the flow rates and minimizing pressure losses are two major goals in the optimization. As the flow rates are different from different outlets because of the hydraulic resistances (the higher the resistance, the smaller the flow rate), a balance in the flow rate can be accomplished in two ways:

1) decreasing the high hydraulic resistances or
2) increasing the low hydraulic resistances.

Since the pressure loss is mostly determined by the narrowest parts of the passage, the diameters of the outlets play a key role. A smaller diameter leads to a high hydraulic resistance and vice-versa. Decreasing the hydraulic resistances would be good for both balances in the flow rates and reducing the pressure losses. Therefore, the diameters of the outlets can be used as the decision variables. To ease the solution process, we use the scaling factors of the initial design as the decision variables i.e.

$$x_i = \frac{D_i}{D_i^{(initial)}} \text{ for } i = 1, \ldots, 4, \tag{5}$$

where D_i is the diameter of the i^{th} outlet and $D_i^{initial}$ is the diameter of the i^{th} outlet in the initial design. The diameters of the outlets were also prolonged as shown in Figure 3 for numerical evaluation using CFD simulations. This was easily accomplished using the custom *Tcl* scripts of ANSYS ICEM. Such a simplification is only accurate if changes in diameters are not greater than the original ones. However, for the solution process, we used the following bounds of the decision variables:

$$\begin{aligned} x_i^{lb} &= 0.5 \text{ for } i = 1, \ldots, 4, \\ x_i^{ub} &= 1.5 \text{ for } i = 1, \ldots, 4. \end{aligned} \tag{6}$$

where x_i^{lb} and x_i^{ub} are the lower and upper bounds of the decision variables.

In addition to maximizing the balance in flow rates and minimizing the pressure losses, a third objective considering the flow rate from the outlet 4 was considered because of its smallest diameter. Finally, we finalized the following three objectives:

f_1 : Minimize variance between flow rates at outlets 1 to 3
: Minimize $var(Q_{1,3})$
f_2 : Minimize pressure loss of the air intake
: Minimize $P_{inlet} - P_{outlet}$
f_3 : Minimize the difference between the flow rate at outlet 4 and the average of the flow rates at outlets 1 to 3
: Minimize $avg(Q_{1,3}) - Q_4$,

where Q_k represents the flow rate from the k^{th} outlet, $avg(Q_{1,3})$ the average flow rate value from outlets 1-3 and P_{inlet} and P_{outlet} are the pressure values at the inlet and the outlet, respectively. As already mentioned, decision variables are the scaling factors of the diameters of the initial design given in (5).

For numerical evaluation of the designs, the ANSYS CFX [12] solver was used for CFD simulations. For some standard shapes such as elbows or pipes, it would be easier to use empirical values for hydraulic resistance coefficients. But in this particular case, CFD takes into account both the exact shape of the design and a mutual interaction of the outlets. This allows for a higher precision in the numerical evaluation. Next, we provide a detailed description of modelling and numerical evaluation.

IV. Modelling and Numerical Evaluation

For the CFD analysis, a three dimensional model of the component was built in the modelling software CATIA [13] as shown in Figure 4. The model was then exported to ANSYS ICEM [14] for meshing and a tetra mesh with prism layers was created. For a better numerical stability, the inlet and outlet parts were extended as shown in the figure (the gray zones). However, these extensions could cause differences in the relative flow rates, and therefore they were considered as free-slip walls. The boundary conditions were set as *Mass Flow* at the inlet and *Relative Pressure* at the outlets, with a k-ϵ turbulence model.

The parametric model was realized using ANSYS ICEM, ANSYS CFX and custom scripts. For the sake of simplicity, the original CATIA model and the mesh were fixed in the parametric model. Changing the diameters was instead approximated by changing the diameters of the extended outlet parts. This approximation was easily realized through ICEM Tcl scripts and gave accurate results for $x_i \leq 1$ (because the pressure loss mostly depends on the narrowest part). For $x_i > 1$, the fixed part of the model plays an important role and increasing x_i in such a case does not decrease the pressure loss significantly.

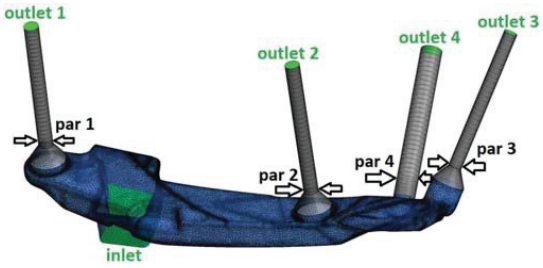


Fig. 4. Parametric CFD model

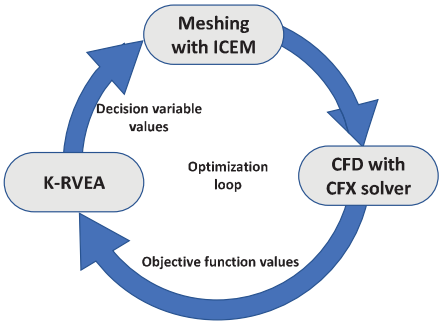


Fig. 5. Illustration of the optimization loop

As mentioned, outlet 4 has the smallest diameter as can be seen in Figure 4 and achieving a high flow rate from this outlet is difficult. Therefore, a special attention was paid towards this outlet while formalizing the objective functions. The mesh generated using ICEM was then exported to CFX for CFD simulations. When using the CFX tools, the values of pressure loss and flow rates were recorded. In order to automate the process during optimization, different scripts were written in Python and Matlab to connect different tools and optimization algorithm.

An illustration of the optimization loop used in this study is shown in Figure 5. First, the meshing is done for a given shape of the component using ANSYS ICEM. After meshing, CFD simulations are performed with the solver ANSYS CFX. Based on this numerical evaluation, K-RVEA produces new values for the decision variables and a new shape is created which then goes for meshing. The loop continues for a prefixed number of evaluations and nondominated solutions of all the evaluated ones are used as the final set of solutions. Next, we provide a brief outline of K-RVEA which was used to solve the problem formulated.

V. Surrogate-assisted evolutionary algorithm

The K-RVEA algorithm [7] has been developed for computationally expensive many-objective optimization problems.

However, the efficiency of the algorithm was also demonstrated on three objective optimization problems in [7]. The algorithm uses elements from its underlying evolutionary algorithm RVEA [15] and Kriging models as surrogates. It consists of three phases as presented in Algorithm 1. Nondominated solutions from all the evaluated ones stored in an archive $A2$ are used as final solutions.

Algorithm 1: K-RVEA

Input: FE^{max}, maximum number of expensive function evaluations

Output: nondominated solutions of all evaluated ones from $A2$

Initialization

1. Create an initial population P generated with some design of experiment technique
2. Initialize the number of function evaluations FE = 0 and two empty archives $A1 = A2 = \phi$
3. Evaluate the population P with the original expensive functions and add them to $A1$ and $A2$, update $FE = FE + |P|$

while $FE \leq FE^{max}$ **do**

4. Train surrogates for each objective function by using individuals in $A1$

Using the surrogates with RVEA

5. Run RVEA with Kriging models to find the individuals to update the surrogates

Updating the surrogates

6. Select individuals from the previous step using a selection strategy and denote the set by I
7. Re-evaluate I with the original expensive functions and update $FE = FE + |I|$, update $A1 = A1 \cup I$ and $A2 = A2 \cup I$
8. Remove extra individuals from $A1$ using management of training archive
9. Go to step 4

In the initialization phase, an initial population is generated e.g. using the Latin hypercube sampling [16]. This population is then evaluated with the original expensive functions and added to two archives $A1$ and $A2$. Individuals in $A1$ are used to build surrogates for each objective function in step 4. The archive $A2$ is used to store all the evaluated solutions.

When using the surrogates with RVEA [15], we use mean values of Kriging models as objective function values. In addition, uncertainty information of the approximated solutions is obtained which is further used in updating the surrogates. Solutions generated are then used to select individuals for re-evaluation with the original expensive objective functions for updating the surrogates.

Updating the surrogates is very important in model management [6], [17]. Individuals for updating the surrogates should be selected in such a way that they enhance both convergence and diversity. In K-RVEA, these two criteria are taken care of with the help of reference vectors and the uncertainty information from the Kriging models. Individuals with a maximum uncertainty are selected whenever diversity is needed. If a satisfactory degree of diversity has already been achieved, individuals with a minimum angle penalized distance are selected, which is one of the selection criteria in RVEA that contributes to convergence. Once these individuals have been selected, they are re-evaluated with the original expensive functions and added to archives A1 and $A2$. As the size of the data set can influence the training time, to decrease the computation time further, some of the individuals are removed from $A1$ in step 8. To be able to do that, reference vectors are used to identify the extra individuals which were not needed for training the surrogates. The individuals in $A1$ are then used to re-train the surrogates in step 4 and the algorithm is run until a maximum number of expensive function evaluations has been reached. For full details about the selection strategy and the management of the training archive, see [7]. Next, we present the results obtained with K-RVEA on the given problem.

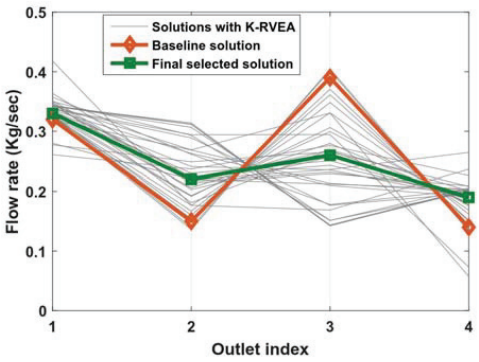


Fig. 7. Flow rates corresponding to nondominated solutions

VI. Results and Discussion

In this section, we report the results and an analysis of the solutions obtained with K-RVEA. Parameters used are given in Table I, where n represents the number of decision variables. Also, note that we ran the optimization solution process only for one run due to a limited computation budget.

TABLE I
Parameter values used

Parameter	Value
Number of function evaluations	200
Size of the training data set	$11n - 1$
Number of reference vectors	105

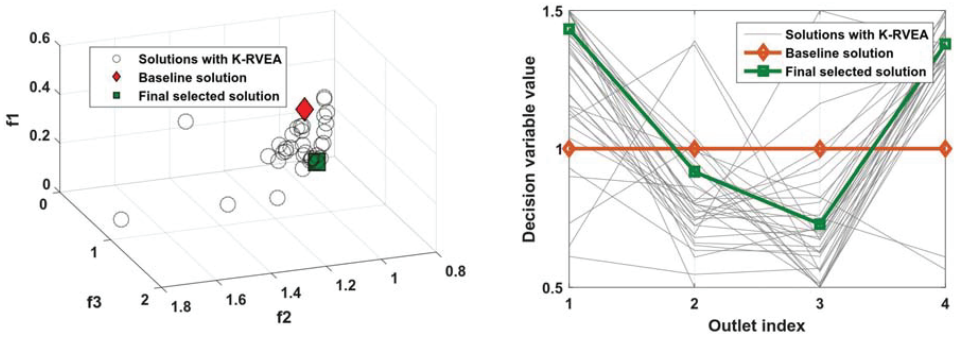


Fig. 6. Nondominated solutions in the objective (normalized) and decision spaces

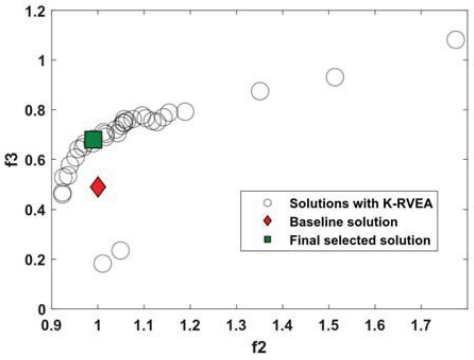


Fig. 8. Projection of nondominated solutions for the second and the third objectives

To protect confidential data, objective function values obtained are normalized in the following way:

$$f_1 : \frac{\sum_{k=1}^{3} |Q_k - avg(Q_{1,3})|}{3 * avg(Q_{1,3})}$$

$$f_2 : \frac{P_{inlet} - P_{outlet}}{P_{original}}$$

$$f_3 : \frac{Q_4}{avg(Q_{1,3})},$$

where $P_{original}$ is the absolute pressure value of the initial geometry.

Nondominated solutions obtained are shown in Figure 6 in the objective and decision spaces. For a given pressure loss, a better balance in the flow rates from outlets 1-3 was obtained. However, pressure losses were high for a high flow rate from the outlet 4. This is due to the fact that outlet 4 has a very small diameter when compared to the other outlets. Solutions are also compared with the baseline solution of the initial geometry as shown in Figure 6. Many solutions obtained with K-RVEA dominate the baseline solution.

As can be seen in the decision space, the algorithm tries to find solutions with a high diameter especially for outlets 1 and 4. This is due to the reason that in the initial geometry, outlet 4 had the smallest diameter and increasing it was the only option of increasing Q_4. The outlet 1 might seem a bit counter-intuitive at first, because in the baseline solution $Q_1 > Q_2$. We assume this is caused by pursuing the minimization of pressure loss and the complexity of the flow, where each outlet can slightly influence the other ones.

We also present the corresponding flow rates from all four outlets in Figure 7. These flow rates are normalized as follows:

$$Q_k = \frac{Q_k}{Q_{input}}, \text{ for } k = 1, \ldots, 4,$$

where Q_{input} is the flow rate at the inlet of the component. It can be seen that for the baseline solution, flow rate values from different outlets were very different from each other and maintaining a uniform temperature was not easily achievable. K-RVEA produced a diverse set of flow rate values as shown in the figure except for outlets 1 and 4. The reason for this is the same as mentioned above i.e. a small diameter for outlet 4 and a low pressure loss from outlet 1.

A projection of the solutions obtained for the second and third objectives is shown in Figure 8. It can be clearly observed the f_2 and f_3 are conflicting objectives, i.e. a high flow rate from outlet 4 causes high pressure losses and consequently, the flow rates from other outlets are decreased. In other words, the diameters for outlets 1-3 decrease less for a high flow rate from outlet 4. For $f_3 \geq 0.46$, all four outlets were active or in other words, all the outlets were fully open. For $f_3 < 0.46$, outlets 1-3 were inactive and pressure losses were increased. This is the reason why two solutions were not contributing to the trade-off between the second and the third objective.

Finally, 40 solutions were shown to the DM and he selected his most preferred solution. The baseline solution and the solution selected by the DM are shown in Figures 6, 7 and 8. As can be seen, the results confirm the original assumptions

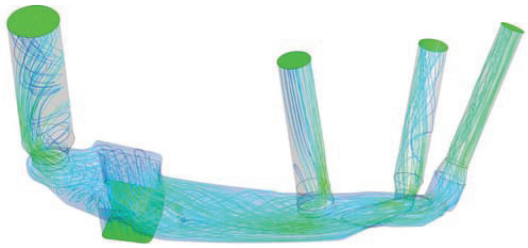


Fig. 9. The design of the final solution selected by the DM

of the necessary of trade-offs between flow rates and pressure loss. Considering the flow rate from outlet 4 as an additional objective did also prove to be a useful decision.

The CFD simulation of the final solution selected by the DM is shown In Figure 9. The solution has a very small variance between flow rates from outlets 1-3 for an equivalent pressure loss compared to the baseline solution. Also, the flow rate from outlet 4 is comparable to flow rates from other outlets. As the balance in flow rates has been increased in the final selected solution, more air flow can be delivered into the cabin using the same fan. This will improve the cooling and heating properties of the cabin which usually improves customer satisfaction.

The DM was happy with the solution obtained. However, out of academic interest, we also compared the solutions of the single run of K-RVEA and RVEA to show the efficiency of using surrogates. We ran RVEA for the same number of function evaluations. The nondominated solutions obtained with both algorithms in the objective space are shown in Figure 10. As can be seen, K-RVEA produced a better distribution of solutions than RVEA. The hypervolume with the number of function evaluations for both algorithms is presented in Figure 11. We used the worst objective function values from nondominated solutions of both algorithms as a reference point (f_i^*) in calculating the hypervolume. The values obtained were normalized by dividing with $\prod_{i=1}^{k} f_i^*$, k being the number of objectives. For a given number of function evaluations, K-RVEA performed better than RVEA in terms of the hypervolume. We also used coverage [18] and inverted generational distance (IGD) [19] as other two performance metrics to compare the results. The values of all three performance metrics after 200 function evaluations are given in Table II. The coverage metric is usually used to compare solutions based on their dominance. For instance, in Table II, 0.5152 means that 51.52 % of the solutions with RVEA are dominated by the ones with K-RVEA. To calculate IGD, we combined nondominated solutions from both algorithms and used this union as the reference set in calculating the values. In all these performance metrics, K-RVEA performed better than RVEA which shows that using surrogates efficiently with evolutionary algorithms can be helpful in getting better quality solutions with a limited computation budget. Comparing K-RVEA with

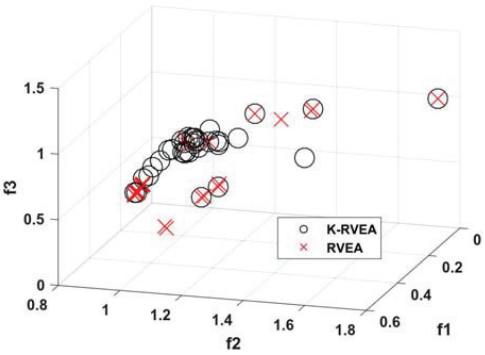


Fig. 10. Nondominated solutions with algorithms K-RVEA and RVEA

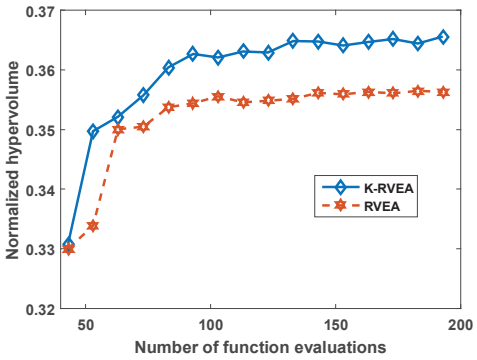


Fig. 11. Normalized hypervolume with the number of function evaluations in a single run

other surrogate-assisted evolutionary algorithms will be our future work.

TABLE II
HYPERVOLUME, COVERAGE AND IGD OF SOLUTIONS WITH K-RVEA AND RVEA

Algorithm	Hypervolume	Coverage	IGD
K-RVEA	**0.3656**	**0.5152**	**0.0881**
RVEA	0.3563	0.2059	0.3320

VII. CONCLUSIONS

A multiobjective shape optimization problem of an air intake ventilation system of a tractor with four outlets was formulated, implemented and solved. Thus, three main challenges in solving real-world problems, formulating the optimization problem, combining optimization and different simulation tools and dealing with computationally expensive objective

functions were emphasized. A description about the modelling of the given problem and solving with a CFD solver was also detailed. To alleviate the computational cost, a recently proposed surrogate-assisted evolutionary algorithm K-RVEA was applied. It was selected as it had shown a very good performance when compared to other surrogate assisted algorithms. A diverse set of solutions representing the balance in flow rates from outlets 1-3, low pressure losses and a high flow rate from outlet 4 were obtained. Among the solutions obtained, a final solution was selected by the DM. The selected solution is significantly better in two objectives and similar in the third objective compared to the baseline solution.

To show the efficiency of K-RVEA, solutions were also compared with its underlying evolutionary algorithm RVEA. K-RVEA performed better than RVEA in terms of hypervolume, coverage and IGD which shows the benefits of using surrogates in dealing with (computationally expensive) real-world problems. Additionally, practitioners in industry usually face the challenges of formulating the optimization problem and connecting different pieces of simulation tools. Therefore, the current approach by focusing on these two challenges and solving the formulated MOP having computationally expensive objective functions may be helpful in providing insight to formulate the optimization problem and solving it with an appropriate algorithm.

In this article, all the nondominated solutions generated were shown to the DM. However, a better way of interacting with the DM with a good visualization of solutions is needed. As the problem is computationally expensive, interacting with the DM may also save the computational resources when not all nondominated solutions are of interest. Thus, developing an interactive algorithm using surrogates will be our future work.

ACKNOWLEDGMENT

This work was funded by TEKES, the Finnish Funding Agency for Innovation under the FiDiPro project DeCoMo. We would also like to thank Valtra Inc. for providing the problem.

REFERENCES

[1] M. Emmerich and B. Naujoks, "Metamodel assisted multiobjective optimisation strategies and their application in airfoil design," in *Adaptive Computing in Design and Manufacture VI*, I. C. Parmee, Ed. Springer London, 2004, pp. 249–260.

[2] J. Knowles, "ParEGO: a hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 10, pp. 50–66, 2006.

[3] W. Ponweiser, T. Wagner, D. Biermann, and M. Vincze, "Multiobjective optimization on a limited budget of evaluations using model-assisted S-metric selection," in *Proceedings of the Parallel Problem Solving from Nature-PPSN X*, G. Rudolph, T. Jansen, S. Lucas, C. Poloni, and N. Beume, Eds. Springer, Berlin, Heidelberg, 2008, pp. 784–794.

[4] Q. Zhang, W. Liu, E. Tsang, and B. Virginas, "Expensive multiobjective optimization by MOEA/D with Gaussian process model," *IEEE Transactions on Evolutionary Computation*, vol. 14, pp. 456–474, 2010.

[5] T. Chugh, K. Sindhya, J. Hakanen, and K. Miettinen, "Handling computationally expensive multiobjective optimization problems with evolutionary algorithms - A survey. Reports of the Department of Mathematical Information Technology, Series B, Scientific Computing no. B 4/2015, University of Jyvaskyla," 2015.

[6] Y. Jin, "Surrogate-assisted evolutionary computation: Recent advances and future challenges," *Swarm and Evolutionary Computation*, vol. 1, no. 2, pp. 61–70, 2011.

[7] T. Chugh, Y. Jin, K. Miettinen, J. Hakanen, and K. Sindhya, "A surrogate-assisted reference vector guided evolutionary algorithm for computationally expensive many-objective optimization," *IEEE Transactions on Evolutionary Computation, to appear*, doi:10.1109/TEVC.2016.262230.

[8] J. Shinkyu, Y. Minemura, and S. Obayashi, "Optimization of combustion chamber for diesel engine using Kriging model," *Journal of Fluid Science and Technology*, vol. 1, pp. 138–146, 2006.

[9] A. Forrester and A. Keane, "Recent advances in surrogate-based optimization," *Progress in Aerospace Sciences*, vol. 45, pp. 50–79, 2009.

[10] I. E. Idelchik, *Handbook of Hydraulic Resistance*. Jaico Publishing House, 2005.

[11] R. A. Granger, *Fluid Mechanics*. Dover Publishing House, 1985.

[12] N. Trev, *CFX: Computational Fluid Dynamics, Ansys, HVAC*. International Book Market Service Limited, 2012.

[13] S. Tickoo, *CATIA V5R17: For Engineers and Designers*. Dreamtech Press, 2008.

[14] *ANSYS ICEM CFD Tutorial Manual*, ANSYS, Inc., 2013.

[15] R. Cheng, Y. Jin, M. Olhofer, and B. Sendhoff, "A reference vector guided evolutionary algorithm for many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 20, pp. 773–791, 2016.

[16] M. Mckay, R. Beckman, and W. Conover, "A comparison of three methods for selecting values of input variables in the analysis of output from a computer code," *Technometrics*, vol. 42, pp. 55–61, 2000.

[17] Y. Jin, "A comprehensive survey of fitness approximation in evolutionary computation," *Soft Computing*, vol. 9, pp. 3–12, 2005.

[18] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach," *IEEE Transactions on Evolutionary Computation*, vol. 3, pp. 257–271, 1999.

[19] P. Bosman and D. Thierens, "The balance between proximity and diversity in multiobjective evolutionary algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 7, pp. 174–188, 2003.

PIV

ON CONSTRAINT HANDLING IN SURROGATE-ASSISTED EVOLUTIONARY MANY-OBJECTIVE OPTIMIZATION

by

Tinkle Chugh, Karthik Sindhya, Kaisa Miettinen, Jussi Hakanen, Yaochu Jin

In 14th International Conference on Parallel Problem Solving from Nature, 2016 Proceedings, Edited by J. Handl et al., 214-224, Springer, 2016

PV

AN INTERACTIVE SIMPLE INDICATOR-BASED EVOLUTIONARY ALGORITHM (I-SIBEA) FOR MULTIOBJECTIVE OPTIMIZATION PROBLEMS

by

Tinkle Chugh, Karthik Sindhya, Jussi Hakanen, Kaisa Miettinen

Evolutionary Multi-Criterion Optimization: 8th International Conference, Proceedings, Part II, Edited by A. Gaspar-Cunha, C. Antunes, C. Coello, Springer, Berlin, Heidelberg, 277-291, 2015