Optimization Letters
https://doi.org/10.1007/s11590-020-01575-2

ORIGINAL PAPER

A comparative study on surrogate models for SAEAs

Mônica A. C. Valadão[1,3] · **Lucas S. Batista**[2]

Received: 31 May 2019 / Accepted: 22 March 2020

Abstract

Surrogate model assisted evolutionary algorithms (SAEAs) are metamodel-based strategies usually employed on the optimization of problems that demand a high computational cost to be evaluated. SAEAs employ metamodels, like Kriging and radial basis function (RBF), to speed up convergence towards good quality solutions and to reduce the number of function evaluations. However, investigations concerning the influence of metamodels in SAEAs performance have not been developed yet. In this context, this paper performs an investigative study on commonly adopted metamodels to compare the ordinary Kriging (OK), first-order universal Kriging (UK1), second-order universal Kriging (UK2), blind Kriging (BK) and RBF metamodels performance when embedded into a single-objective SAEA Framework (SAEA/F). The results obtained suggest that the OK metamodel presents a slightly better improvement than the others, although it does not present statistically significant difference in relation to UK1, UK2, and BK. The RBF showed the lowest computational cost, but the worst performance. However, this worse performance is around 2% in relation to the other metamodels. Furthermore, the results show that BK presents the highest computational cost without any significant improvement in solution quality when compared to OK, UK1, and UK2.

Keywords Kriging · RBF · Surrogate model assisted evolutionary algorithms

Electronic supplementary material The online version of this article (https://doi.org/10.1007/s11590-020-01575-2) contains supplementary material, which is available to authorized users.

✉ Mônica A. C. Valadão
monica.valadao@ict.ufvjm.edu.br

Lucas S. Batista
lusoba@ufmg.br

1 Graduate Program in Electrical Engineering, Universidade Federal de Minas Gerais, Belo Horizonte, MG 31270-901, Brazil

2 Department of Electrical Engineering, Universidade Federal de Minas Gerais, Belo Horizonte, MG 31270-901, Brazil

3 Science and Technology Institute, Universidade Federal dos Vales do Jequitinhonha e Mucuri, Diamantina, MG 39100-000, Brazil

Published online: 04 April 2020

1 Introduction

Evolutionary algorithms (EA) represent an interesting option to deal with real-world optimization problems since they do not depend on problem features, such as continuity, differentiability, smoothness and convexity [1–3]. However, it might not be possible to apply these methods in problems with expensive evaluation functions [4–6]. To address this issue, several approaches integrate metamodels and EAs to reduce the number of evaluation functions required. In such methods, the metamodel is used to give an estimate of the true fitness in specific stages of EAs, e.g., to evaluate all or a subset of candidate solutions on a metamodel (prediction function) in each iteration. These approaches are termed as surrogate model assisted evolutionary algorithms (SAEAs) [5–7].

In SAEAs context, the most frequently used type of metamodel is the Gaussian Process (GP) [5,6,8–10]. It is important to observe, however, that several works point out the GP as design and analysis of computer experiments (DACE) or Kriging [5,7,9,11], but, in fact, they refer to the ordinary Kriging (OK) metamodel. The Kriging model has a more comprehensive definition as can be seen in Sect. 2. Another metamodel that is commonly employed in SAEAs is the RBF, although less frequently used than GP [7,9,10]. From the literature review, a trend in choosing GP metamodels can be observed. This may be justified by the fact that metamodels like Kriging provide an uncertainty measure for the predicted function values. Merit functions are defined from this measure, namely, expected improvement (EI) and lower confidence bound (LCB) [12,13], which can be used to guide the EA towards promising regions of the search space. In several SAEAs that incorporate Kriging metamodels [6,8,9], these merit functions assist the EA on the *evolution control* phase. In general, *evolution control* means a way in which evaluation functions involved will be performed either in the true function or in the prediction function [4,14]. The use of these merit functions can be understood as a mechanism to choose which solutions to evaluate in the original function.

Although the literature mentions different variations of the Kriging to solve optimization problems, many of them [11,15–18] emerged in the Efficient Global Optimization (EGO) context. The EGO is a method developed by [19] to deal with expensive black-box functions. This approach starts building a global Ordinary Kriging metamodel. Then, the metamodel is iteratively updated with solutions obtained via maximization of the expected improvement (EI) function. When dealing with functions with a large number of variables, this approach may demand a large number of evaluations on the original function to build a reliable global metamodel. Hence, the time associated with the metamodel building can be prohibitive, especially if there is a high computational cost on the function evaluation.

It can be noticed that variations of the Kriging model are not usually explored in SAEAs context; frequently, these strategies adopt GP or RBF metamodels. Even though the literature justifies the use of such methods [5–9], an investigation concerning their relative performance (in terms of solution quality and computational cost) in the SAEA context has not been developed yet. Many SAEA-based works state which metamodel to use based on studies that investigate global approximations performed through different metamodels. In [9] a global GP and a local RBF are adopted based

Table 1 Sumary of limitations regarding the justification adopted for the selection of a metamodel

Work	Limitations
[5–9,25,26]	Variations of Kriging metamodel are usually disregarded in the context of SAEAs without any justification
[25]	GP is applied without either a proper justification or analysis. It is generally applied based only on its popularity
[9,26]	Metamodels are often selected based only on their performance for global approximation
[5,6,8]	GP is sometimes selected because it provides an uncertainty measure. In this sense, the way to employ this measure is sometimes based on the EGO context
[7]	RBF is sometimes selected because of its insensitive to problem dimension increasing

mainly on [20–22], which investigate the quality of global approximations built by GP, RBF, response surface methodology (RSM), and artificial neural networks (ANNs) metamodels. Other authors employ the GP due to the uncertainty measure it provides. This is the case for the SAEAs proposed in [5,6,8], which employ the lower confidence bound (LCB) to map promising regions of the search space that have not been explored. These works are based on optimization strategies studied in [12,19,23,24], which share fundamental ideas from EGO, this one involving successive steps that aim at improving the quality of the approximation function. On the other hand, [7] chooses the RBF based on studies that indicate this metamodel as insensitive to problem dimension increasing. Finally, there are authors that adopt the GP because of its popularity [25].

As it is pointed out, there are some drawbacks regarding the strategy used in order to select an appropriate metamodel to integrate within a SAEA framework. Some of these limitations are emphasized in Table 1.

Considering the limitations pointed out above, this work performs an investigative study on commonly adopted metamodels so as to compare ordinary Kriging (OK), first-order universal Kriging (UK1), second-order universal Kriging (UK2), blind Kriging (BK), and RBF metamodels when embedded into a single-objective SAEA Framework. To the best of our knowledge, this analysis represents a first step towards the development of a more adequate way to support the selection of a metamodel to integrate within a SAEA framework. As performance indicators, it is considered the quality of the final solution and the time spent on metamodel building. In this investigation, each metamodel is incorporated into a SAEA framework (SAEA/F), which is used to solve a set of analytical functions of single-objective optimization problems. The SAEA structure employed is simple, but represents a fair framework for the proposed experiments. The results obtained suggest that the OK metamodel presents a slightly better improvement than the others, although it does not present statistically significant difference in relation to UK1, UK2, and BK. It also suggest the OK, UK1, and UK2 as a promising compromise between performance and computational time for training the metamodel. The RBF is indicated as the metamodel with the lowest computational

cost, but with the lowest performance. However, this worse performance is around 2% in relation to other metamodels. Such difference value may not have much practical significance for some real situations. Regarding the BK method, even with the longer computational time required to build the associated metamodel, it does not overcome the OK, UK1, and UK2 performances.

The remainder of this paper is structured as follows. Section 2 is dedicated to an overview of the metamodeling techniques considered in this work. Section 3 presents the SAEA/F adopted for the metamodel comparison. Section 4 explains the experimental design to investigate the influence of each metamodel on SAEA/F performance. Section 5 shows the results obtained for each SAEA/F configuration. Finally, Sect. 6 presents the concluding remarks.

2 Metamodels

In the description of the metamodeling techniques considered in this work, $\mathrm{y} : \mathbb{X} \subset \mathbb{R}^n \longrightarrow \mathbb{R}$ denotes the response function (or the true function) of a given problem. It is also assumed that the evaluations performed on $\mathrm{y}(\cdot)$ are deterministic.

2.1 Kriging model

Consider a sample with N points (or solutions) $\mathbf{X} = [\mathbf{x}^1, \ldots, \mathbf{x}^N]^T$ and the respective responses (or observations) $\mathbf{y} = [\mathrm{y}(\mathbf{x}^1), \ldots, \mathrm{y}(\mathbf{x}^N)]^T$. In the Kriging model, each response is treated as a realization of a random variable (or stochastic process), which is defined as a regression model [12,19,27,28]

$$\mathrm{y}(\mathbf{x}^i) = \mathrm{f}(\mathbf{x}^i) + \epsilon(\mathbf{x}^i), \quad i = 1, \ldots, N \tag{1}$$

where $\mathrm{f}(\mathbf{x}^i) = \mathbf{f}^T\boldsymbol{\alpha} = \sum_{k=0}^{d} \alpha_k \mathrm{f}_k(\mathbf{x}^i)$, $d \leq N-1$, is a linear combination of the regression functions $\mathrm{f}_k(\cdot)$ and α_k, $k = 0, \ldots, d$, are the corresponding coefficients; $\epsilon(\cdot)$ is a random process with mean zero and variance σ^2. Regression functions are usually identified as basis functions and, in the scope of the present study, it is defined as

$$\mathrm{f}_k(\mathbf{x}) = \prod_{r=1}^{n} x_r^{q_r}, \, q_r \in [0, Q], \quad \sum_{r=1}^{n} q_r \leq Q \tag{2}$$

where Q is the highest integer that satisfies $\dfrac{(n+Q)!}{(Q!n!)} < N-1$ [15]. In (1), $\epsilon(\cdot) \sim \mathcal{N}(0, \sigma^2)$ and it is assumed a covariance as

$$Cov(\epsilon(\mathbf{x}^i), \epsilon(\mathbf{x}^j)) = \sigma^2 \mathrm{R}(\mathbf{x}^i, \mathbf{x}^j) \tag{3}$$

where $\mathrm{R}(\cdot,\cdot)$ is a Gaussian correlation function. Here, $\mathrm{R}(\cdot,\cdot,\cdot)$ is of the form

$$\mathrm{R}(\boldsymbol{\theta}, \mathbf{x}, \tilde{\mathbf{x}}) = \prod_{r=1}^{n} \exp(-\theta_r |x_r - \tilde{x}_r|^2) \tag{4}$$

with $\boldsymbol{\theta} \in \mathcal{H}$, $\mathcal{H} = \{[\theta_1, \ldots, \theta_n] \,|\, \theta_r > 0 \,\forall\, r = 1, \ldots, n\}$. Although other correlation functions are mentioned in literature [27,29–31], the expression in (4) is frequently used when applying Kriging model in computer experiments context [4,5,15,27]. The components in $\boldsymbol{\theta}$ are the parameters of the metamodel, which are estimated based on the available sample. Such parameters represent the importance of variables and also influence how correlated the points are. A high correlation is related to low values in $\boldsymbol{\theta}$, while large values in $\boldsymbol{\theta}$ lead to low correlation. More details regarding the effect of $\boldsymbol{\theta}$ parameter in the correlation of sample can be found in [12,13,19].

The optimal choice of $\boldsymbol{\theta}$, accounting the sample data, is defined as the maximum likelihood estimator (MLE), where the likelihood function has the following form

$$L(\boldsymbol{\theta}) = \frac{1}{\sqrt{(2\pi\sigma^2)^N |\mathbf{R}|}} \exp\left(\frac{-1}{2\sigma^2}(\mathbf{y} - \mathbf{F}\boldsymbol{\alpha})^T \mathbf{R}^{-1} (\mathbf{y} - \mathbf{F}\boldsymbol{\alpha})\right) \tag{5}$$

where $\boldsymbol{F} = [\mathrm{f}_k(\mathbf{x}^i)]_{N\times(d+1)}$ and $\boldsymbol{R} = [\mathrm{R}(\boldsymbol{\theta}, \mathbf{x}^i, \mathbf{x}^j)]_{N\times N}$ are, respectively, the regression matrix and a correlation matrix with $i, j \in \{1, \ldots, N\}$ and $k \in \{0, \ldots, d\}$. For a given $\boldsymbol{\theta}$, the expressions

$$\hat{\boldsymbol{\alpha}} = (\mathbf{F}^T \mathbf{R}^{-1} \mathbf{F})^{-1} \mathbf{F}^T \mathbf{R}^{-1} \mathbf{y} \tag{6}$$

and

$$\hat{\sigma}^2 = \frac{1}{N}(\mathbf{y} - \mathbf{F}\hat{\boldsymbol{\alpha}})^T \mathbf{R}^{-1} (\mathbf{y} - \mathbf{F}\hat{\boldsymbol{\alpha}}) \tag{7}$$

provide the respective MLEs of $\boldsymbol{\alpha}$ and σ^2. After replacing (6)–(7) into (5) and then taking the natural logarithm, the optimal $\boldsymbol{\theta}$ is equivalent to

$$\arg\max_{\boldsymbol{\theta}\in\mathcal{H}} \ln L(\boldsymbol{\theta}) \tag{8}$$

where

$$\ln L(\boldsymbol{\theta}) = -\frac{N}{2}\ln(2\pi) - N\ln(\hat{\sigma}) - \ln(|\mathbf{R}|) - \frac{1}{2}. \tag{9}$$

The optimal solution $\boldsymbol{\theta}$ of the problem in (8) provides the correlation matrix $\mathbf{R}$, and subsequently the values of $\hat{\boldsymbol{\alpha}}$ and $\hat{\sigma}^2$ according to (6) and (7). Thus, the predicted evaluation at an untried $\mathbf{x}$ and the mean squared error (MSE) of this prediction are, respectively,

$$\hat{\mathrm{y}}(\mathbf{x}) = \mathbf{f}^T \hat{\boldsymbol{\alpha}} + \mathbf{r}^T \mathbf{R}^{-1} (\mathbf{y} - \mathbf{F}\hat{\boldsymbol{\alpha}}) \tag{10}$$

and

$$\hat{s}^2(\mathbf{x}) = \hat{\sigma}^2 \left[1 - (\mathbf{f}^T(\mathbf{x}) + \mathbf{r}^T(\mathbf{x})) \begin{pmatrix} \mathbf{0} & \mathbf{F}^T \\ \mathbf{F} & \mathbf{R} \end{pmatrix}^{-1} \begin{pmatrix} \mathbf{f}(\mathbf{x}) \\ \mathbf{r}(\mathbf{x}) \end{pmatrix}\right] \tag{11}$$

where $\mathbf{r}^T(\mathbf{x}) = [\mathrm{R}(\boldsymbol{\theta}, \mathbf{x}^1, \mathbf{x}), \ldots, \mathrm{R}(\boldsymbol{\theta}, \mathbf{x}^N, \mathbf{x})]$ is the vector of correlation; $\mathrm{R}(\boldsymbol{\theta}, \mathbf{x}^i, \mathbf{x})$ represents the correlation between $\epsilon(\cdot)$ at the sampled point $\mathbf{x}^i$ and $\epsilon(\cdot)$ at an untried $\mathbf{x}$,

for $i = 1, \ldots, N$. In addition, the Kriging model defined in this way is an interpolation model; Eq. (10) evaluated at a point $\tilde{\mathbf{x}}$ that does belong to the sample, coincides with $y(\tilde{x})$ since $\mathbf{r}^T(\tilde{x})$ matches with a line of the matrix $\mathbf{R}$.

2.1.1 Simple Kriging, ordinary Kriging and universal Kriging

In a general way, variations of the Kriging model are characterized by assumptions made about the mean of each response $y(\mathbf{x}^i)$ in (1). By assuming the mean is known and implementing the necessary modifications on the expressions involved, it provides the Simple Kriging (SK) (more details can be found in [31,32]). When the mean of $y(\mathbf{x}^i)$ is unknown and, in this case, it has the form $\mu = \mathbf{f}^T(\mathbf{x}^i)\boldsymbol{\alpha}$, it provides the Universal Kriging (UK), which coincides with the description presented on Sect. 2.1. Variations of the UK are obtained according to the choice of the basis functions. The case $k = 0$ ($f_0(\cdot) = 1$ and $Q = 0$) becomes the Ordinary Kriging (OK), which is widely known as the DACE (Design and Analysis of Computer Experiment) [11,12,19,25,32]. The variations First-Order Universal Kriging (UK1) and Second-Order Universal Kriging (UK2) are obtained when $Q = 1$ and $Q = 2$, respectively. It is worthy to mention that in [30,33] the DACE is defined in a comprehensive way, allowing the choice between the models OK, UK1 and UK2. Comparisons among OK, UK1, and UK2 models are found in [15,18,34], in which the focus is mainly on the quality of the global approximation. In such works is reported that, for some test functions, using a fixed polynomial set ($Q = 1, 2$ in (2)), UK1 and UK2 may provide a better approximation function in comparison with OK. Thus, it is considered to investigate UK1 and UK2 also in this work, since these methods have not been explored in SAEAs context.

2.1.2 Blind Kriging

The Blind Kriging method, proposed by [28], is another kind of Kriging model that assumes $\mu = \mathbf{f}^T(\mathbf{x}^i)\boldsymbol{\alpha}$ unknown. Unlike OK, BK does not assume the basis functions to be known. Instead, they are selected from a set of candidate functions previously defined. These functions must represent linear effects, quadratic effects, and two-factor interactions. When building a BK metamodel, an OK model is first constructed and, in an iterative process, new basis functions are incorporated into the regression model in (1). Since the vector $\mathbf{f}$ is identified, the predictor and error expressions associated with an untried point are the same as the UK. As indicated in [33], the goal of this model is to select, from the set of basis functions, the ones that represent the major variance in the sample. Notice that it must take into account the basis functions that already exist in the model. It also is pointed out in [33] that the candidate functions may not contain the interaction between the variables exhibited by the sample. Thus, the functions incorporated in the model may not improve the OK initially built. Despite that, it is reported in [28,33] a significant improvement in prediction when using this method. This indeed motivates to investigate BK performance in the context of this work. The configuration of the model considered here is the same as presented in [33]. Further details on BK are found in [28,33,35].

2.2 RBF model

This section provides a brief summary of the Radial Basis Function interpolation structure. To this end, a particular instance of the method, adapted from [13,36], is presented. Consider a sample with N points $\mathbf{X} = [\mathbf{x}^1, \ldots, \mathbf{x}^N]^T$ and their corresponding responses $\mathbf{y} = [\mathrm{y}(\mathbf{x}^1), \ldots, \mathrm{y}(\mathbf{x}^N)]^T$, $\mathrm{y} : \mathbb{X} \subset \mathbb{R}^n \longrightarrow \mathbb{R}$, noise free. The value $\mathrm{y}(\mathbf{x})$ at an untried $\mathbf{x}$ is an approximation defined as

$$\hat{\mathrm{y}}(\mathbf{x}) = \omega_0 + \sum_{i=1}^{N} \omega_i \varphi_i(\mathbf{x}) \tag{12}$$

where $\omega_0 = \frac{1}{N}\sum_{i=1}^{N} \mathrm{y}(\mathbf{x}^i)$, ω_i are the unknown parameters, and each $\varphi_i(\cdot)$ (usually called radial basis function) depends on the Euclidean distance $||\cdot||$ between $\mathbf{x}$ and the sampled $\mathbf{x}^i$ according to the following expression

$$\varphi_i(\mathbf{x}) = \phi(||\mathbf{x} - \mathbf{x}^i||). \tag{13}$$

The function $\phi(\cdot)$ defined here is a Gaussian function $\phi(r) = e^{\frac{-r^2}{2\sigma^2}}$ with $\sigma = 1$. The condition $\mathrm{y}(\mathbf{x}^i) = \hat{\mathrm{y}}(\mathbf{x}^i)$, $i = 1, \ldots, N$, allows to calculate the ω_i according to the expression

$$\mathbf{w} = \boldsymbol{\Psi}^{-1}(\mathbf{y} - \mathbf{1}\omega_0), \tag{14}$$

where $\mathbf{w}_{i,1} = \omega_i$; $\boldsymbol{\Psi}_{i,j} = \varphi_j(\mathbf{x}^i)$ is the interpolation matrix; $\mathbf{1}_{i,1} = 1$; $i, j = 1, \ldots, N$. Hence, the expression in (13) can be rewritten as following

$$\hat{\mathrm{y}}(\mathbf{x}) = \omega_0 + \mathbf{m}\boldsymbol{\Psi}^{-1}(\mathbf{y} - \mathbf{1}\omega_0) \tag{15}$$

where $\mathbf{m} = [\varphi_i(\mathbf{x})]_{1\times N}$. With regards to $\phi(\cdot)$ there are several types of choices, e.g, $\phi(r) = r^\alpha$ with $1 \leq \alpha \leq 3$ (power function), $\phi(r) = r^2 \ln(r)$ (thin plate spline), and $\phi(r) = \sqrt{\sigma + r^2}$ (multiquadric). Some basis functions, unlike the Gaussian, require to replace ω_0 in (12) by a polynomial model to ensure the inverse of the interpolation matrix exists. This correction in (12) demands additional parameters to the model. The σ in Gaussian function is another parameter that can be added to the metamodel. In this case, it must be estimated from the sample instead of considered as a constant. Further details for the general form of RBF are found in [36,37]. Variations of RBF model presented above are found in computer experiment context, either using a different $\phi(\cdot)$ or using more parameters which are estimated using the metamodel sample [7,9,15,26,37]. However, we have decided to use a simple configuration of RBF model in this work, as presented in this section. Although the RBF model, unlike Kriging, does not provide an uncertainty measure for the predicted function values, it has been tried out in some SAEAs [7,9,26]. Its relative low computational cost might explain this fact. Also, it is indicated as insensitive to problem dimension increasing. Since in this work we propose to investigate some commonly adopted metamodels when embedded into a SAEA framework, thus, it is important to consider the RBF one, which also represents some distinct features from the Kriging type.

3 SAEA framework

This section presents the SAEA Framework (SAEA/F) used in this paper to compare the metamodels. The structure of this SAEA shares some characteristics with those used in [5,26], such as, the way the metamodel is used and the population selection which is based only on the true function. On the other hand, the use of subpopulations turns it distinct from them. Moreover, the SAEA/F structure is simple, but it represents a fair framework for the goal of this work. Algorithm 1 shows the pseudocode of SAEA/F. In each iteration of this SAEA, a metamodel is employed to help identify promising solutions to evaluate on the original function that represents the problem; in this case, the chosen solutions are those with the best metamodel fitness. In this sense, $\mathrm{y}(\cdot)$ is the objective function associated with a box-constrained single-objective problem; $\hat{\mathrm{y}}(\cdot)$ is the metamodel function (or the prediction function); the quantities s, p, m and k represent, respectively, the number of initial function evaluations performed in $\mathrm{y}(\cdot)$, the number of solutions of the population to be evolved, the number of solutions in the sample which are used to build the metamodel, and the number of solutions generated for each candidate solution of the current population; the set A is an archive with limited cardinality that is used to store solutions and corresponding evaluations performed in $\mathrm{y}(\cdot)$.

Algorithm 1: SAEA Framework (SAEA/F)

Input: $\mathrm{y}(\cdot)$, Ω, s, p, m, k
1 Generate s random solutions from Ω using the Latin Hypercube Sampling;
2 Evaluate the s generated solutions $\mathrm{y}(\cdot)$;
3 Store in A the s solutions and their corresponding evaluations;
4 Select the p solutions with the best fitness from A to form an initial population P;
5 **while** *stopping criterion is not met* **do**
6 Generate k solutions for each solution in P ($k \cdot p$ new solutions);
7 Use the m newest solutions in A to build a metamodel $\hat{\mathrm{y}}(\cdot)$;
8 Evaluate each of the $k \cdot p$ new solutions on $\hat{\mathrm{y}}(\cdot)$;
9 Select the p solutions with best metamodel fitness (one per subpopulation);
10 Evaluate all of these p solutions in $\mathrm{y}(\cdot)$ and update A;
11 Select the population for the next generation, considering both the current population and the p evaluations performed in $\mathrm{y}(\cdot)$;
12 **end**
Output: Best solution among all the evaluations performed in $\mathrm{y}(\cdot)$

In step 6 of Algorithm 1, for each alternative $\mathbf{x}^i$, $i = 1, \ldots, p$, of the current population, evolutionary operators are applied to generate a subpopulation S^i with k solutions. These new solutions are evaluated in the metamodel according to step 8. Then, in step 9, for each S^i, the solution with the best metamodel fitness is chosen to be evaluated in the original function in step 10. It is expected that the generation of these $k \cdot p$ new solutions keeps the algorithm diversity and, thereby, increases the possibilities of the metamodel to identify solutions that represent some improvement. Depending on the quality of the prediction function $\hat{\mathrm{y}}(\cdot)$, the metamodel built in on step 7 might lead to a poor choice of the solutions to be evaluated in $\mathrm{y}(\cdot)$. Because of this, the sample used to build the metamodel (step 7) is composed of the m newest

solutions from A, aiming to ensure that the evaluations on $\hat{y}(\cdot)$ are carried out regarding solutions which are not far away from the approximated region. Notice that the purpose of the set A is to store solutions which will constitute the metamodel sample. Hence, it is essential to avoid redundant solutions into A. Moreover, the update of A (step 10), should consider for the Kriging model that, at each iteration, a metamodel is built using a normalized sample according to

$$\tilde{M} = \frac{M - \mu(M)}{\sigma(M)} \quad \text{and} \quad \mathrm{y}(\tilde{M}) = \frac{\mathrm{y}(M) - \mu(\mathrm{y}(M))}{\sigma(\mathrm{y}(M))} \tag{16}$$

with $M \subset A$, where the functions $\mu(\cdot)$ and $\sigma(\cdot)$ represent, respectively, the mean and standard deviation; each coordinate direction in $\tilde{M}$ is mean zero and variance 1. Hence, this update must be done in such a way to ensure that the variances of M and $\mathrm{y}(M)$ are non-zero. Notice that (16) removes sensitivity to scale of values and ensures homoscedasticity of observations required in (1). This scaling improves the Kriging model according to [33]. The update also considers, at each iteration, the maintenance of solutions which are near to the current population. To select the new population, in step 11, for each $i = 1, \ldots, p$, it is chosen the solution with the best fitness between the solution i, in the current population, and the solution of the corresponding subpopulation S^i, which has been evaluated in $\mathrm{y}(\cdot)$. Finally, at each iteration of the Algorithm 1 the best overall solution, evaluated at $\mathrm{y}(\cdot)$, must be stored. It is important to highlight that step 7 of this SAEA/F can be performed by using any of the metamodels presented in Sect. 2.

4 Experimental design

This section presents the design protocol adopted to investigate the influence of each of the methods Ordinary Kriging (OK), First-Order Universal Kriging (UK1), Second-Order Universal Kriging (UK2), Blind Kriging (BK), and Radial Basis Function (RBF) in the SAEA/F performance.

To evaluate the influence of the metamodels cited above, five different configurations of the SAEA/F are considered, in which each one is defined in the SAEA/F regarding a specific metamodel. As the benchmark, it is considered the minimization of ten analytic functions described in Table 2. These functions have the value 0 (zero) as the global minimum. Function $\mathrm{y}_3(\cdot)$ is unimodal while the others are multimodal. Such functions are used in [5–7,9,10,38] and more details can be found in [39].

In this work, the optimizer employed on SAEA/F is a Differential Evolution algorithm with DE/best/1 and binomial recombination operators [40]. The crossover parameter is set as $C = 0.5$, and the scaling factor F is randomly chosen at each mutation operation with $F \in [0.4, 0.9]$. The Kriging metamodels are built through the ooDACE *toolbox* [33,41] and the RBF through the SURROGATES toolbox [42]. The SURROGATES toolbox uses the third party software RBF by [43]. It is worth mentioning that the computation presented in (16) is done by the ooDACE toolbox itself, as can be verified in their source files. Each configuration of the SAEA/F was executed five times for each test function from Table 2. The stop criterion was set as a

Table 2 Analytic functions used in the computational experiment

Function	Characteristics	Definition
Ackley	*Multimodal*	$y_1(\mathbf{x}) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + \exp(1)$ $x_i \in [-32.768, 32.768]$
Dixon–Price	*Multimodal (valley-shaped)*	$y_2(\mathbf{x}) = (x_1 - 1)^2 + \sum_{i=1}^{n} i(2x_i^2 - x_{i-1})^2$ $x_i \in [-10, 10]$
Ellipsoid	*Unimodal*	$y_3(\mathbf{x}) = \sum_{i=1}^{n} i \cdot x_i^2$ $x_i \in [-5.12, 5.12]$
Griewank	*Multimodal*	$y_4(\mathbf{x}) = 1 + \sum_{i=1}^{n}\frac{x_i^2}{4000} - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right)$ $x_i \in [-600, 600]$
Levy	*Multimodal*	$y_5(\mathbf{x}) = \sin^2(\pi w_i) + \sum_{i=1}^{n-1}(w_i - 1)^2[1 + 10\sin^2(\pi w_i + 1)] + (w_n - 1)^2[1 + \sin^2(2\pi w_n)]$ $x_i \in [-10, 10]$, $w_i = 1 + (x_i - 1)/4$, $i = i, \ldots, n$
Perm n, β	*Multimodal (bowl-shaped)*	$y_6(\mathbf{x}) = \sum_{i=1}^{n}\left(\sum_{j=1}^{n}(j + \beta)\left(x_j^i - \frac{1}{j^i}\right)\right)^2$ $x_i \in [-n, n]$, $\beta = 10$
Rastrigin	*Multimodal*	$y_7(\mathbf{x}) = 10n + \sum_{i=1}^{n}[x_i^2 - 10\cos(2\pi x_i)]$ $x_i \in [-5.12, 5.12]$
Rosenbrock	*Multimodal (narrow valley)*	$y_8(\mathbf{x}) = \sum_{i=1}^{n}[100(x_{i+1} - x_i^2)^2 + (1 - x_i^2)^2]$ $x_i \in [-2.048, 2.048]$
Styblinski–Tang	*Multimodal*	$y_9(\mathbf{x}) = \frac{1}{2}\sum_{i=1}^{n}(x_i^4 - 16x_i^2 + 5x_i)$ $x_i \in [-5, 5]$
Zakharov	*Multimodal (plate-shaped)*	$y_{10}(\mathbf{x}) = \sum_{i=1}^{n}x_i^2 + \left(\sum_{i=1}^{n}0.5ix_i\right)^2 + \left(\sum_{i=1}^{n}0.5ix_i\right)^4$ $x_i \in [-5, 10]$

Table 3 Values of the SAEA/F parameters used for each problem size (number of variables)

n	s	p	m	$neval$
2	100	20	70	1000
5	100	20	70	2000
10	100	20	70	3000
15	250	50	150	5000
20	250	50	240	5000

maximum number of evaluations on the original function ($neval$). Table 3 shows the settings for the SAEA/F parameters used in this work.

For each execution of a SAEA/F configuration the following information was collected at the end of each iteration: the value of the objective function of the best candidate solution and the metamodel building time. Consider y_I as the value of the objective function of the best solution of the initial population, and y_F as the value of the objective function of the best solution of the current population at a given iteration. An indicator for the algorithm convergence can be set as:

– $\Delta_{\%}$: percentage of improvement in the objective function of the best solution of the current iteration in relation to the best solution of the initial population:

$$\Delta_{\%} = \frac{(\mathrm{y}_I - \mathrm{y}_F)}{\mathrm{y}_I} \cdot 100 \tag{17}$$

in which $\Delta_{\%}$ is defined in the interval [0, 100]. Another indicator used is the absolute gap according to

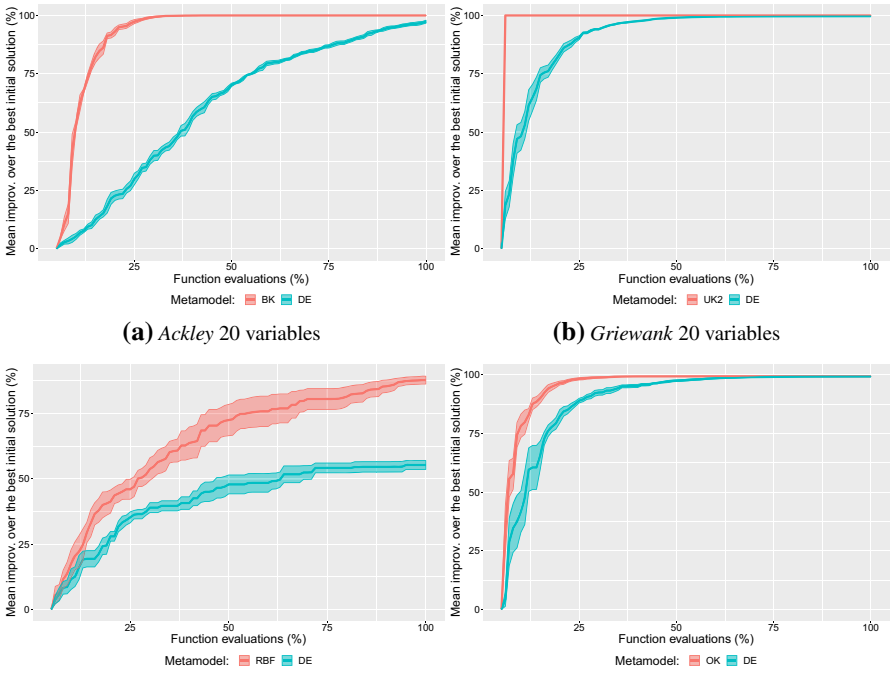
$$\Delta_{abs} = |\mathrm{y}_F - \mathrm{y}^{\star}| \tag{18}$$

where $\mathrm{y}^{\star} = 0$ is the global minimum value. As $\mathrm{y}^{\star} = 0$ for all benchmark functions, it has in (18) that $\Delta_{abs} = y_F$. This indicator is used to measure the quality of the final solution provided by the SAEA/F, which is also an estimate to the accuracy of the metamodel.

From the results, graphical analyses were performed to observe the behavior of the SAEA/F performance with different metamodels. Besides, a statistical test was performed in order to check statistically significant differences in SAEA/F performance, regarding the metamodel as interest factors. For this purpose, multiple comparisons are performed using the Tukey test [44] with a significance level $\alpha = 0.05$.

The workstation used to perform the experiments was a Intel(R) Xeon(R) Silver 4116 with 156 GiB of main memory running the Ubuntu 18.04 (64 bits). All algorithms were implemented in MATLAB 2017b, and the statistical analysis was performed using R 3.5.2.

The source code of the implemented algorithms and all scripts of analyses performed are available for download at https://github.com/monicavaladao/PhD.



(a) *Ackley* 20 variables

(b) *Griewank* 20 variables

(c) *Rastrigin* 20 variables

(d) *Rosenbrock* 20 variáveis

Fig. 1 Comparison of convergence curves of Blind Kriging (**a**), Universal Kriging 2 (**b**), RBF (**c**) and Ordinary Kriging (**d**) metamodels against the differential evolution

5 Analysis of the results

This section presents the results of the experiments described in the previous sections. Figure 1 shows, for some problems, convergence curves of DE against different SAEA/F configurations. This figure shows the mean and the standard error with respect to the percentage of function evaluations budget. It can be noticed, as reported in the literature [5–9,25,26], that incorporating metamodels on a EA speeds up convergence and allows the EA to achieve solutions of better quality using the same budget of function evaluations.

Table 4 presents mean values and the standard deviation (in parentheses) of the percentage improvement ($\Delta_{\%}$ (17)) of the final solution. The values in this table are stratified by problem size, i.e, for each dimension n; the mean values and the standard deviation are related to all the functions in Table 2.

It can be noticed, from Table 4, a similar behavior among the SAEA/F configurations. The SAEA/F configuration using the OK metamodel returns more promising values of $\Delta_{\%}$ for $n = 15, 20$. The RBF metamodel stands out over the metamodels UK1 and BK when $n = 15, 20$. For the smallest instances, i.e, $n = 2, 5$, UK2 presents better results than the others methods.

It is also worth noting the quality of the final solution (Δ_{abs} (18)) returned by each SAEA/F configuration. Table 5 presents mean values and standard deviation (in

Table 4 Mean and standard deviation (in parenthesis) of the percentage of improvement $\Delta_{\%}$ of each metamodel over all test functions, stratified by problem size

n	OK	UK1	UK2	BK	RBF
2	99.178 (3.408)	99.341 (2.745)	99.508 (3.251)	99.087 (3.692)	97.240 (9.516)
5	98.103 (5.065)	98.009 (5.522)	98.823 (3.578)	98.551 (4.523)	97.140 (5.540)
10	97.545 (4.826)	97.753 (4.342)	96.845 (8.074)	97.257 (5.393)	91.789 (13.937)
15	95.663 (7.257)	94.870 (10.004)	95.149 (11.951)	94.202 (11.189)	95.124 (8.097)
20	96.662 (6.463)	94.654 (11.879)	91.374 (16.673)	94.044 (12.040)	95.371 (6.738)

parentheses) of the objective function of the best solution found. The values in this table are stratified in the same way as in Table 4. Notice that the observations derived from the analysis of Table 4 should apply in Table 5, however, it does not happen for $n = 15$. Essentially, since all the metamodels have returned (regarding the stop criterion used) an equally poor performance on the y_6 function (when $n = 15, 20$), this function was not considered in Table 5 in order not to compromise the analysis of the mean and standard deviation values. The magnitude of the values that the y_6 function might assume is discrepant from the other functions when the number of variables increases. Also, it is worth mentioning that the quality of the final solution does not depend only on the accuracy of the metamodel. The ability of SAEA/F to find good solutions is also dependent on the evolutionary structure used. Although the values returned are not good when comparing against the global minimum value, we believe that these values are consistent with the simplicity of SAEA/F and with the goal of this work.

An exploratory analysis of the data shows that the SAEA/F configurations present a high discrepancy for the building time of the metamodels. Each problem size may lead to different magnitudes on the metamodel building time. Since the absolute value of the building time is not relevant for this study (as it depends on the computer configuration), it was decided to represent the normalized time, as illustrated in Fig. 2. This normalization is performed, for each metamodel met, according to

$$\bar{t}_{met} = \frac{t_{met}}{t_{max}} \tag{19}$$

where t_{met} is the spent time to build the metamodel met and t_{max} is the higher building time over all metamodels. An online supplementary material[1] provides, for each problem and dimension, absolute mean values and standard deviations of $\Delta_{\%}$ and building time, associated with each SAEA/F configuration. It also presents a figure with different charts, for each problem, in order to illustrate problem-dependent performance of the metamodels.

Figure 2 shows the mean values and the standard deviation of improvement percentages and metamodel building time (after normalization) for each one of the SAEA/F configurations along all functions and problem size. Notice that, when analyzing the mean improvement concerning the best initial solution ($\Delta_{\%}$) over all problems and

[1] https://github.com/monicavaladao/PhD/blob/master/supplementary-material.pdf.

Table 5 Mean and standard deviation (in parenthesis) of best objective value of each metamodel over all test functions, stratified by problem size

n	OK	UK1	UK2	BK	RBF
2	0.028 (0.150)	0.026 (0.149)	0.023 (0.148)	0.048 (0.207)	0.042 (0.128)
5	1.184 (3.191)	1.234 (2.948)	0.771 (2.357)	0.802 (2.546)	1.312 (3.112)
10	4.313 (8.247)	3.757 (6.709)	4.109 (9.515)	5.104 (9.302)	7.802 (14.769)
15	11.704 (16.854)	13.955 (21.257)	11.244 (21.514)	14.324 (21.457)	10.969 (19.506)
20	16.552 (23.024)	25.688 (42.085)	36.361 (51.714)	31.297 (49.007)	16.915 (24.167)

dimensions, the metamodels achieve a similar performance with $\Delta_{\%}$ ranging between 95% and 97.5%. However, the OK metamodel presents a slightly better improvement than the others, although it does not present statistically significant difference in relation to UK1, UK2 and BK (as shown in Fig. 3a hereinafter).

While the metamodels present similar performance in terms of mean improvement over the best initial solution, they require different runtime to build the metamodel. The x-axis in Fig. 2 shows the mean time required to build the metamodel. Notice that the runtime is normalized in relation to the BK, which is the metamodel that demands the longest runtime. It was already expected the BK to take longer than the other metamodels since it needs to build an OK metamodel as part of its process. Then, the values in the x-axis are the proportion of time required by the metamodels in relation to the time spent by BK. It can be noticed that metamodels OK, UK1 and UK2 do not present significant differences of runtime between them, but they demand around 10% of the time of BK. On the other hand, RBF is the fastest metamodel, demanding less than 1% of the time demanded by BK.

Notice that the comments derived from the analysis of Fig. 2 remain the same when analyzing the multiple comparisons illustrated in Fig. 3. Regarding the percentage of improvement $\Delta_{\%}$, Fig. 3a shows statistically significant differences between OK and RBF, and between UK1 and RBF. On the other hand, it is noted from Fig. 3b the absence of statistically significant differences in relation to building time among the metamodels UK1, UK2, and OK.

In view of these results, OK, UK1 and UK2 may represent the better trade-off between time and percent improvement from the metamodels studied. However, when the runtime is really critical, the RBF metamodel may be the choice, since it runs faster and its difference in percent improvement compared to OK, UK1 and UK2 is around 2%. It is noteworthy the results presented in Figs. 2 and 3 may override some problem-dependent performance of the metamodels. However, our main interest of study is the effect of the metamodels in the average performance of the SAEA. For the reader interested in problem-dependent behavior, the supplementary material presents different charts for each problem.

The analysis of the different SAEA/F configurations previously presented regards the data stratified by dimension, including all test problems. To illustrate the behavior of the different SAEA/F configurations, Fig. 4 presents boxplots of the percentage of improvement according to the number of function evaluations (%) for some problems. During the optimization process, the approach based on OK has suggested a non inferior convergence capability compared to UK1, UK2, and BK metamodels. Also, the RBF metamodel has showed an interesting behavior in some instances. For example, an improved convergence rate can be observed for the Zakharov, Rastrigin and Rosenbrock test functions.

Finally, it is important to highlight the potential of the RBF method concerning both its lowest computational cost and its convergence capability compared to the other methods. In the context of this work, the RBF performance might be associated with the choice of the basis function and with the way in which this method is used. Here, the RBF is employed as an approximation function for the region defined by the current population, and the basis function is set as a Gaussian with fixed parameter σ. Other works in the literature [9,26] employ the RBF as a local approximation function

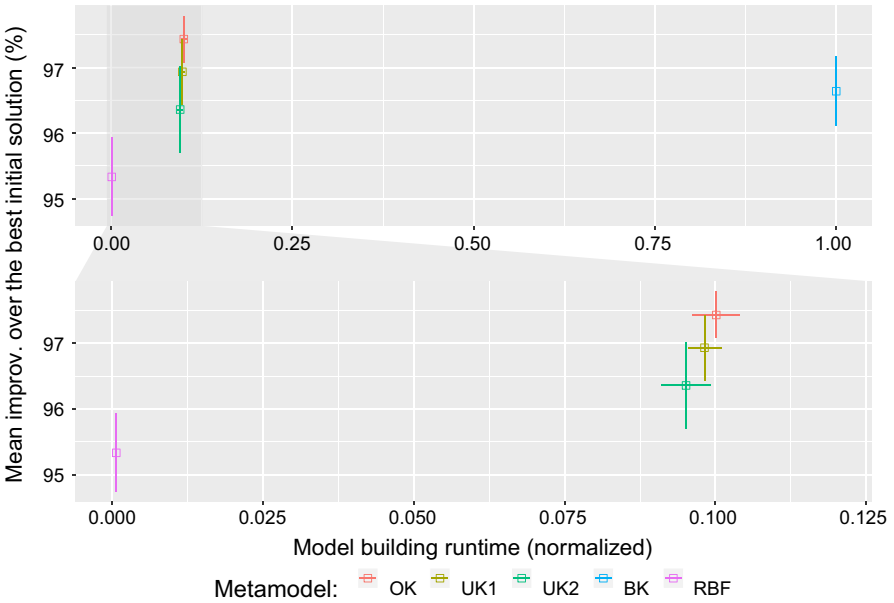
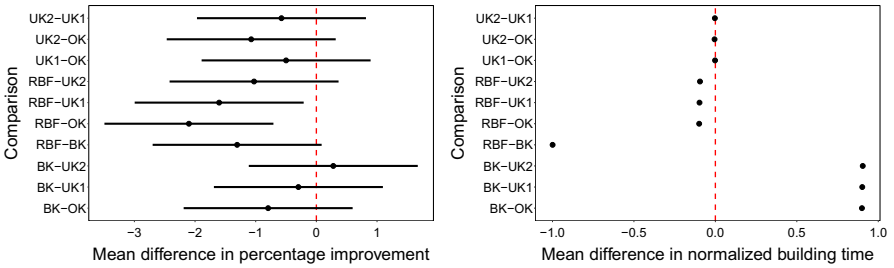


Fig. 2 Crossbar comparing the effect of the five different metamodels. Square markers are the mean values and the vertical lines are the standard error. The standard error for model building times are so small that square markers of the means cover their horizontal lines. The bottom panel is a zoom on the x-axis to highlight the performance of OK, UK1, UK2, and RBF metamodels

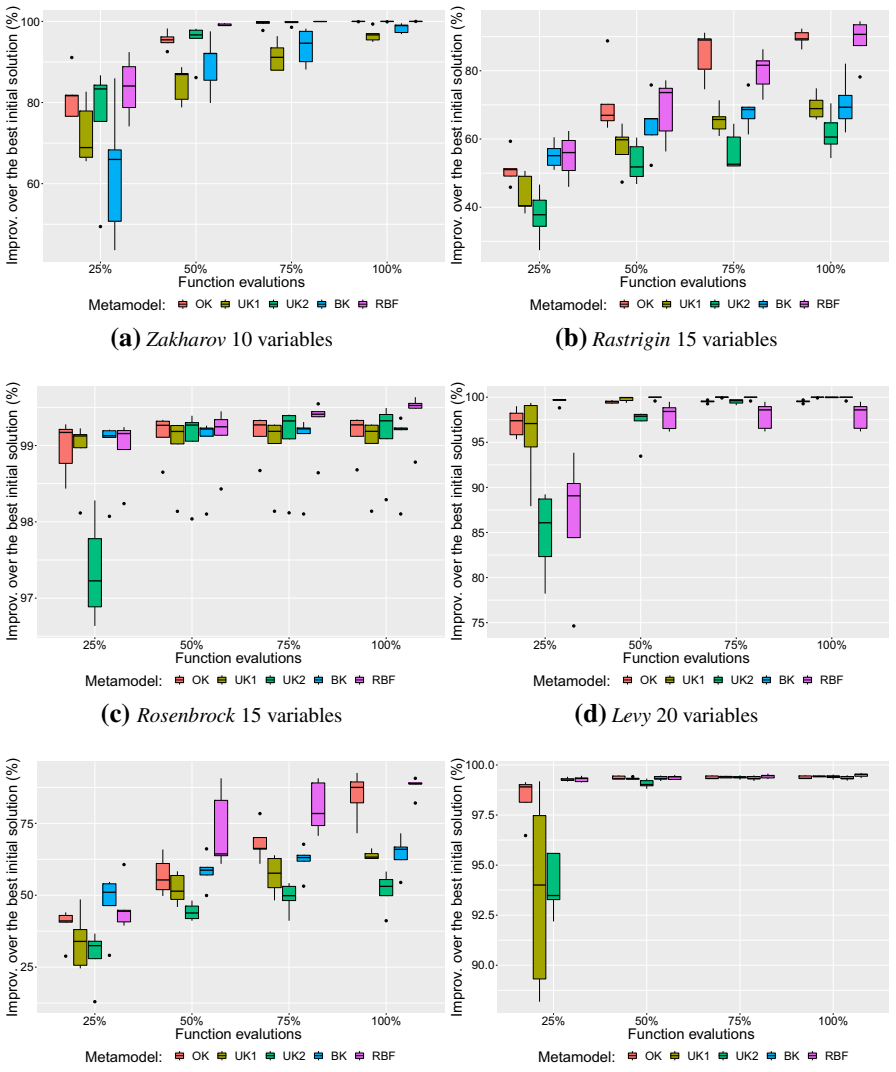


(a) Mean differences in percentage improvement

(b) Mean differences in normalized metamodel building times

Fig. 3 95%-confidence intervals obtained from the multi-comparison tests for solution improvement and metamodel building times

for each solution to be evaluated on the metamodel, which adds a computational cost but also increases its performance. Besides, in [7] the RBF is used as a global approximation together with a Gaussian basis function, with the parameter σ being iteratively updated. Thus, enhanced versions of the RBF are expected to improve its performance without considerably increase of its computational cost.



(a) *Zakharov* 10 variables

(b) *Rastrigin* 15 variables

(c) *Rosenbrock* 15 variables

(d) *Levy* 20 variables

(e) *Rastrigin* 20 variables

(f) *Rosenbrock* 20 variables

Fig. 4 Boxplots of the percentage improvement along the function evaluations for the SAEA/F with different metamodels

6 Conclusions

This work investigated the influence of different metamodels, namely OK, UK1, UK2, BK and RBF, on the performance of a SAEA/F. To that end, the quality of the final solution returned by the SAEA/F and the metamodel building time were considered as performance indicators. A set of analytical functions with different number of variables $n \in \{2, 5, 10, 15, 20\}$ was considered.

The use of the convergence indicator $\Delta_{\%}$, instead of the objective function, allowed to eliminate the magnitude differences in the values of each different problem. This indicator also allowed to represent, in the same graphic, the percentage of improvement along the time. To complement such analysis, multiple comparisons were performed to check statistically significant differences among the SAEA/F configurations. It is worth mentioning that, when analyzing the result of all test problems together may override some problem-dependent performance of the metamodels. However, our main interest was to investigate the effect of the metamodels in the average performance of the SAEA.

From the results obtained, it was observed that the OK metamodel presents a slightly better improvement than the others, although it does not present statistically significant difference in relation to UK1, UK2, and BK. For the metamodels OK, UK1 and UK2, a similar behavior was observed regarding the building times. In contrast, BK demanded a high computational cost. The RBF metamodel presented the lowest computational cost but the lowest performance with regards to solution quality. Among the compared methods, OK, UK1, and UK2 presented a promising trade-off between time and solution quality. However, it was observed that the RBF represents an interesting alternative to these metamodels since its difference in percent improvement compared to OK, UK1, and UK2 was around 2%. Such difference value may not have much practical significance for some real situations.

Acknowledgements This work was partially supported by PRPq/UFMG and by the Brazilian agencies CNPq, FAPEMIG and CAPES. The authors would also like to thank the comments and invaluable suggestions to improve the quality of this manuscript offered by our colleagues André L. Maravilha and Fillipe Goulart (ORCS Lab. / UFMG).

References

1. Coello, C.C., Lamont, G.B., van Veldhuizen, D.A.: Evolutionary Algorithms for Solving Multi-objective Problems. Genetic and Evolutionary Computation, 2nd edn. Springer, Berlin (2007)
2. Collette, Y., Siarry, P.: Multiobjective Optimization: Principles and Case Studies. Springer, Berlin (2004)
3. Deb, K.: Multi-objective Optimization using Evolutionary Algorithms, 1st edn. Wiley, Hoboken (2001)
4. Jin, Y.: A comprehensive survey of fitness approximation in evolutionary computation. Soft. Comput. **9**(1), 3–12 (2005)
5. Emmerich, M.T.M., Giannakoglou, K.C., Naujoks, B.: Single- and multiobjective evolutionary optimization assisted by Gaussian random field metamodels. IEEE Trans. Evol. Comput. **10**(4), 421–439 (2006)
6. Liu, B., Zhang, Q., Gielen, G.G.E.: A Gaussian process surrogate model assisted evolutionary algorithm for medium scale expensive optimization problems. IEEE Trans. Evol. Comput. **18**(2), 180–192 (2014)
7. Sun, C., Jin, Y., Cheng, R., Ding, J., Zeng, J.: Surrogate-assisted cooperative swarm optimization of high-dimensional expensive problems. IEEE Trans. Evol. Comput. **21**(4), 644–660 (2017)
8. Büche, D., Scharaudolph, N.N., Koumountsakos, P.: Accelerating evolutionary algorithms with Gaussian process fitness function models. IEEE Trans. Syst. Man Cybern. C (Appl. Rev) **35**(2), 183–194 (2005)
9. Zhou, Z., Ong, Y.S., Nair, P.B., Keane, A.J., Lum, K.Y.: Combining global and local surrogate models to accelerate evolutionary optimization. IEEE Trans. Syst. Man Cybern. C (Appl. Rev) **37**(1), 66–76 (2007)
10. Lim, D., Jin, Y., Ong, Y.S., Sendhoff, B.: Generalizing surrogate-assisted evolutionary computation. IEEE Trans. Evol. Comput. **14**, 329–355 (2010)

11. Hao, W., Shaoping, W., Tomovic, M.M.: Modified sequential kriging optimization for multidisciplinary complex product simulation. Chin. J. Aeronaut. **23**(5), 616–622 (2010)
12. Schonlau, M.: Computer experiments and global optimization. Ph.D. thesis, University of Waterloo (1997)
13. Forrester, A.I.J., Sóbester, A., Keane, A.J.: Engineering Design via Surrogate Modelling: A Practical Guide. Wiley, Hoboken (2008)
14. Jin, Y., Olhofer, M., Sendhoff, B.: On evolutioary optimization with approximate fitness function. In: Genetic and Evolutionary Computation Conference, pp. 786–793 (2000)
15. Zhao, L., Choi, K.K., Lee, I.: Metamodeling method using dynamic kriging for design optimization. AIAA J. **49**(9), 2034–2046 (2011)
16. Xia, B., Baatar, N., Ren, Z., Koh, C.S.: A numerically efficient muli-objective optimization algorithm: combination of dynamic Taylor Kriging and differential evolution. IEEE Trans. Magn. **51**(3), 1–4 (2015)
17. Xia, B., Ren, Z., seop Koh, C.: Comparative study on Kriging surrogate models for metaheuristic optimization of multidimensional eletromagnetic problems. IEEE Trans. Magn. **51**(3), 1–4 (2015)
18. Palar, P.S., Shimoyama, K.: On efficient global optimization via universal Kriging surrogate models. Struct. Multidiscip. Optim. **57**(6), 2377–2397 (2017)
19. Jones, D.R., Schonlau, M., Welch, W.J.: Efficient global optimization of expensive black-box functions. J. Global Optim. **13**(4), 455–492 (1998)
20. Giunta, A.A., Watson, L.T.: A comparison of approximation modeling techniques-polynomial versus interpolating models. In: 7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, p. 4758 (1998)
21. Jin, R., Chen, W., Simpson, T.W.: Comparative studies of metamodelling techniques under multiple modelling criteria. Struct. Multidiscip. Optim. **23**(1), 1–13 (2001)
22. Daberkow, D.D., Mavris, D.N.: New approaches to conceptual and preliminary aircraft design: a comparative assessment of a neural network formulation and a response surface methodology. In: 1998 World Aviation Conference, 985509, pp. 1–13. American Institute of Aeronautics and Astronautics (AIAA), Anaheim, CA (1998)
23. Trosset, M.W., Torczon, V.: Numerical optimization using computer experiments. Technical Report 97-38, Institute for Computer Applications in Science and Engineering (ICASE), NASA Langley Research Center, Hampton VA (1997)
24. Torczon, V., Trosset, M.W.: Using approximation to accelerate engineering design optimization. Technical Report 98-33, Institute for Computer Applications in Science and Engineering, NASA Langley Research Center (1998)
25. Ma, H., Fei, M., Simon, D., Mo, H.: Update-based evolution control: a new fitness approximation method for evolutionary algorithms. Eng. Optim. **47**(9), 1177–1190 (2015)
26. Regis, R.G., Shoemaker, C.A.: Local function approximation in evolutionary algorithms for the optimization of costly functions. IEEE Trans. Evol. Comput. **8**(5), 490–505 (2004)
27. Sacks, J., Welch, W.J., Mitchell, T.J., Wynn, H.P.: Design and analysis of computer experiments. Stat. Sci. **4**(4), 409–423 (1989)
28. Joseph, V.R., Hung, Y., Sudjianto, A.: Blind Kriging: a new method for developing metamodels. J. Mech. Des. **130**, 031102(1–7) (2008)
29. Mackay, D.J.C.: Introduction to Gaussian Process. Cambridge University (1998). http://www.inference.org.uk/mackay/gpB.pdf
30. Lophanev, S.N., Nielsen, H.B., Søndergaard, J.: DACE—a MATLAB kriging toolbox. Technical Report IMM-TR-2002-12, Technical University of Denmark (2002)
31. Roustant, O., Ginsbourger, D., Deville, Y.: DiceKriging, DiceOptim: two R packages for the analysis of computer experiments by Kriging-based metamodeling and optimization. J. Stat. Softw. **51**(1), 1–55 (2012)
32. Ginsbourger, D., Riche, R.L., Carraro, L.: Kriging is well-suited to parallelize optimization. In: Tenne, Y., Goh, C.K. (eds.) Computational Intelligence in Expensive Optimization Problems, Adaptation, Learning and Optimization, vol. 2, pp. 131–162. Springer, Berlin (2010). Chap. 6
33. Couckuyt, I., Forrester, A., Gorissen, D., Turck, F.D., Dhaene, T.: Blind Kriging: Implementation and performance analysis. Adv. Eng. Softw. **49**, 1–13 (2012)
34. Martin, J.D., Simpson, T.W.: Use of Kriging models to approximate deterministic computer models. AIAA J. **43**(4), 853–863 (2005)

35. Forrester, A.I.J., Keane, A.J.: Recent advances in surrogate-based optimization. Prog. Aerosp. Sci. **45**(1–3), 50–79 (2009)
36. Jin, R., Chen, W., Sudjianto, A.: On sequential sampling for global metamodeling in engineering design. In: International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, p. 10. American Society of Mechanical Engineers (ASME), Montreal, Canada (2002)
37. Rocha, H.: On the selection of the most adequate radial basis function. Appl. Math. Model. **33**(3), 1573–1583 (2009)
38. Laguna, M., Martí, R.: Experimental testing of advanced scatter search designs for global optimization of multimodal functions. J. Global Optim. **33**(2), 235–255 (2005)
39. Surjanovic, S., Bingham, D.: Virtual library of simulation experiments: test functions and datasets. http://www.sfu.ca/~ssurjano (2018). Retrieved 27 April 2018
40. Price, K.V., Storn, R.M.: Differential Evolution: A Practical Approach to Global Optimization. Springer, Berlin (2005)
41. Couckuyt, I., Dhaene, T., Demeester, P.: ooDACE toolbox a Matlab Kriging toolbox: getting started, 3rd June edn (2013). http://sumo.intec.ugent.be/ooDACE
42. Viana, F.A.C.: SURROGATES toolbox user's guide, version 2.1 edn (2010). http://sites.google.com/site/felipeacviana/surrogatestoolbox
43. Jēkabsons, G.: RBF: radial basis function interpolation for MATLAB/OCTAVE, version 1.1 edn (2009). http://www.cs.rtu.lv/jekabsons/regression.html
44. Dean, A., Voss, D.: Design and Analysis of Experiments. Springer Texts in Statistic. Springer, New York (1999)