Chapter 9
On Similarity-Based Surrogate Models for Expensive Single- and Multi-objective Evolutionary Optimization

L.G. Fonseca, H.J.C. Barbosa, and A.C.C. Lemonge

Abstract. In this chapter we propose a surrogate-assisted framework for expensive single- and multi-objective evolutionary optimization, under a fixed budget of computationally intensive evaluations. The framework uses similarity-based surrogate models and an individual-based model management with pre-selection. Instead of existing frameworks where the surrogates are used to improve the performance of evolutionary operators or as local search tools, here we use them to allow for an augmented number of generations to evolve solutions. The introduction of the surrogates into the evolutionary cycle is controlled by a single parameter, which is related with the number of generations performed by the evolutionary algorithm. Numerical experiments are conducted in order to assess the applicability and the performance in constrained and unconstrained, single- and multi-objective optimization problems. The results show that the present framework arises as an attractive alternative to improve the final solutions with a fixed budget of expensive evaluations.

9.1 Introduction

Several problems of interest in science and engineering are or can be advantageously formulated as optimization problems. However, modern problems have lead to the development of increasingly complex and computationally expensive simulation models. When the optimization algorithm involves the repeated use of expensive simulations to evaluate the candidate solutions, the computational cost of such

L.G. Fonseca
Natl Lab for Scientific Computing – LNCC, Petropolis RJ Brazil
e-mail: goliatt@lncc.br

H.J.C. Barbosa
Natl Lab for Scientific Computing – LNCC, Petropolis RJ Brazil
e-mail: hcbm@lncc.br

A.C.C. Lemonge
Department of Applied and Computational Mechanics,
Federal University of Juiz de Fora – UFJF, Juiz de Fora MG Brazil
e-mail: afonso.lemonge@ufjf.edu.br

Y. Tenne and C.-K. Goh (Eds.): Computational Intel. in Expensive Opti. Prob., ALO 2, pp. 219–248.
springerlink.com

applications can be excessively high. A trade-off between the number of calls to the expensive simulation model and the quality of the final solutions must often be established. As result, an improvement of the optimization process is necessary.

A possible solution to this problem is the use of a surrogate model, or metamodel. In this case, when evaluating candidate solutions in the optimization cycle, the computationally intensive simulation model is substituted by the surrogate model, which should be a relatively inexpensive approximation of the original model [24].

Genetic Algorithms (GAs) [21], inspired by Darwin's theory of evolution by natural selection, are powerful and versatile tools in difficult search and optimization problems. They do not require differentiability or continuity of the objective function, are less sensitive to the initialization procedures, and less prone to entrapment in local optima. However, they usually require a large number of evaluations in order to reach a satisfactory solution, and when expensive simulations are involved, that can become a serious drawback to their application.

The idea of reducing the computation time or improving the solutions performing less computationally expensive function evaluations, appeared early in the evolutionary computation literature [22]. It should be mentioned also that there are additional reasons for using surrogate models in evolutionary algorithms: (a) to reduce complexity [37], (b) to smooth the fitness landscape [61], (c) when there is no explicit fitness available, and (d) in noisy environments [25].

Several surrogate models, of varying cost and accuracy, can be found in the literature, such as polynomial models [36], artificial neural networks [17], Kriging or Gaussian processes [15], radial basis functions [30, 31], and support vector machines [27]. Of course such techniques can also be combined and used as an ensemble [32, 41].

Research in surrogate-assisted frameworks for solving problems with computationally expensive objective functions has been receiving increasing attention in the last few years [7, 14, 16, 18, 26, 43, 64].

In the evolutionary optimization context, the surrogate model is constructed from previously obtained solutions and used to evaluate new candidate solutions, avoiding expensive simulations. An interesting strategy, when a given budget of expensive evaluations is assumed, is to combine both exact and surrogate evaluations along the evolutionary process in order to allow for an extension in the number of generations, which can have a positive impact in the final result.

This chapter is focused on the use of a similarity-based surrogate model (SBSM) to assist evolutionary algorithms in solving single- and multi-objective optimization problems with a limited computational budget. Examples of similarity-based surrogate models are fitness inheritance [56], fitness imitation [24], and the nearest neighbor approximation model [4, 54].

In the surrogate-assisted optimization presented here, the individuals in the parent population (evaluated by the original function) are sequentially stored in a database, and then they are used to construct a surrogate model, based on similarity, which is used along the optimization procedure to perform extra (surrogate) evaluations, resulting in a larger number of generations.

This chapter is organized as follows. The optimization problem is described in Section 9.2. Section 9.3 presents the similarity-based surrogate models and the surrogate assisted evolutionary optimization algorithm, for single- and multi-objective optimization. The numerical experiments conducted are presented and discussed in Section 9.4, and finally the concluding remarks are given in Section 9.5.

9.2 The Optimization Problem

The optimization problems considered here can be written as

$$\begin{array}{rl} \text{minimize} & f_1(x), f_2(x), \ldots, f_{n_{obj}}(x) \\ \text{with} & x = (x_1, \ldots, x_n) \in \mathscr{S} \\ \text{subject to} & g_j(x) \leq 0, \quad j = 1, \ldots, n_i \\ & x_i^L \leq x_i \leq x_i^U \end{array} \tag{9.1}$$

where $f_i(x)$ is the ith objective function to be minimized, n_{obj} is the number of objectives, n is the number of design variables, $\mathscr{S}$ is the search space bounded by $x^L \leq x \leq x^U$, and n_i is the number of inequality constraints. The feasible region is defined by $\mathscr{S}$ and the n_i inequality constraints $g_j(x)$.

We have multi-objective (MO) optimization when $n_{obj} \geq 2$. Single-objective (SO) optimization ($n_{obj} = 1$) is a special case of the formulation above. Also, in the absence of constraints ($n_i = 0$) we have the single- and multi-objective unconstrained optimization problems.

In MO optimization a set of solutions representing the tradeoff among the different objectives rather than an unique optimal solution is sought. This set of solutions is also known as the Pareto optimal set and these solutions are also termed noninferior, admissible, or efficient solutions [20]. The corresponding objective vectors of these solutions are termed nondominated and each objective component of any nondominated solution in the Pareto optimal set can only be improved by degrading at least one of its other objective components [58]. The concept of Pareto dominance and Pareto optimality will form the basis of solution quality. Pareto dominance is defined by

$$\begin{aligned} x_1 \prec_P x_2 \; (x_1 \text{ Pareto-dominates } x_2) :\Leftrightarrow & \\ & \forall i \in \{1, \ldots, n_{obj}\} : f_i(x_1) \leq f_i(x_2) \wedge \\ & \exists j \in \{1, \ldots, n_{obj}\} : f_j(x_1) < f_j(x_2). \end{aligned} \tag{9.2}$$

The Pareto optimal front (PFT) is the set of nondominated solutions such that $PFT = \{f_i(x^*) | \nexists f_j(x) \prec_P f_i(x^*),\ j \in \{1, \ldots, n_{obj}\}\}$.

9.3 Surrogate-Assisted Evolutionary Optimization

Surrogate modeling, or metamodeling, can be viewed as the process of capturing the essential features of a complex computer simulation (original evaluation function) in a simpler, analytical model by providing an approximation of the input/output relation of the original model. The surrogate model should be simple, general, and keep the number of control parameters as small as possible [5]. Examples of such surrogates are the similarity-based surrogate models.

In this section we describe the similarity-based models, and the surrogate-assisted evolutionary algorithms for single- and multi-objective optimization.

9.3.1 Similarity-Based Surrogate Models (SBSMs)

Similarity-based surrogate models (SBSMs) can be classified as *lazy* learners [2] (and also memory-based learners) since that, in contrast to *eager* learning algorithms such as Neural Networks, Polynomial Response Surfaces, and Support Vector Machines, which generate a model and then discard the inputs, SBSMs simply store their inputs and defer processing until a prediction of the fitness value of a new individual is requested. Then they reply by combining their stored data (previous samples) using a similarity measure, and discard the constructed answer as well as any intermediate results.

Among the SBSMs one can find the Fitness Inheritance procedure, Fitness Imitation, and the nearest neighbors method. In the following subsections we present those approaches, and describe with details the nearest neighbor method, used here as a surrogate model.

9.3.1.1 Fitness Inheritance

The fitness inheritance procedure was first proposed by Smith et al [56], and since then has been applied in several problems [6, 12, 13, 49, 52, 63] and algorithms [38, 45]. In fitness inheritance, all the individuals in the initial population have their fitness value obtained via fitness function. Thereafter, the fitness of a fraction of the individuals in the subsequent populations is inherited from their parents. The remaining individuals are evaluated using the original fitness function (referred to as simulation model).

The inheritance procedure is described as follows. Given an individual x^h generated by evolutionary operators (crossover and mutation), from the parents x^{p_1} and x^{p_2}. The surrogate evaluation is given by:

$$\widehat{f}(x^h) = \begin{cases} f(x^{p_i}) & \text{if } d(x^h, x^{p_i}) = 0,\ i = 1 \text{ or } 2 \\ \frac{s(x^h,x^{p_1})f(x^{p_1})+s(x^h,x^{p_2})f(x^{p_2})}{s(x^h,x^{p_1})+s(x^h,x^{p_2})} & \text{otherwise} \end{cases} \tag{9.3}$$

where $s(x^h, x^{p_i})$ is the similarity between x^h and x^{p_i}. The assumption is that an offspring is similar to its parents and thus its fitness is assigned as the weighted average of the parents fitness.

In the inheritance procedure an entire simulation is replaced by a procedure with negligible computational cost, which may lead to great computational savings that grow with the rate of application of the inheritance technique and the cost of the

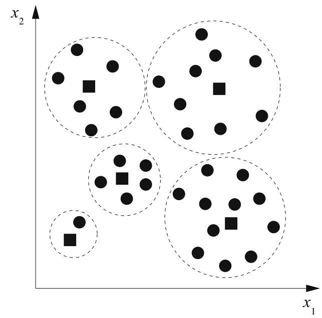


Fig. 9.1 Illustration of the Fitness Imitation procedure. The individuals inside the dotted circles belong to the same group. The representative individual, denoted by a black square, is evaluated by the exact function. The remaining individuals are evaluated by a surrogate model, their predicted fitness being calculated according to the distance to the representative individual

fitness function evaluation [8, 51]. In fact, the inheritance procedure may be orders of magnitude less expensive than the standard fitness evaluation. However, this approach introduces some noise in the search process and may adversely affect the final solution found [13].

9.3.1.2 Fitness Imitation

In Fitness Imitation [24], the individuals are clustered into several groups. Several clustering techniques can be used to perform this task [28]. Then, only the individual that represents its cluster is evaluated using the fitness function. The choice of the representative individual can be made either deterministically or randomly [35]. The fitness value of other individuals in the same cluster will be estimated from the representative individual based on a similarity measure. If a new individual to be evaluated does not belong to any cluster, it is evaluated by the original function. The term Fitness Imitation is used in contrast to Fitness Inheritance.

An illustration of the Fitness Imitation procedure is depicted in Figure 9.1. Examples of applications of this procedure can be found in [3, 28, 35].

9.3.1.3 Nearest Neighbors

The nearest neighbors surrogate model (k-NN) is a simple and transparent surrogate model where the approximations are built based on a set $\mathscr{D}$, which stores η individuals (samples).

The idea of using k-NN to assist an evolutionary algorithm was explored in [46, 47], where the aim was to reduce the number of exact function evaluations needed during the search. Here we use the surrogate to extend the generations, and to guide the search towards improved solutions.

Given an offspring x^h, the corresponding value $\widehat{f}(x^h) \approx f(x^h)$, to be assigned to x^h is

$$\widehat{f}(x^h) = \begin{cases} f(x^{\mathscr{I}_j}) & \text{if } x^h = x^{\mathscr{I}_j}, \text{ for some } j = 1,\ldots,\eta \\ \frac{\sum_{j=1}^{k} s(x^h, x^{\mathscr{I}_j})^u f(x^{\mathscr{I}_j})}{\sum_{j=1}^{k} s(x^h, x^{\mathscr{I}_j})^u} & \text{otherwise} \end{cases} \tag{9.4}$$

where $s(x^h, x^i)$ is a similarity measure between x^h and x^i, $\mathscr{I}_j$, $j = 1,\ldots,\eta$ is a list that stores the individuals in the set $\mathscr{D}$ most similar to x^h, k is the number of neighbors used to build the surrogate and u is set to 2.

The main advantages of the k-NN technique are that it is flexible, does not have severe restrictions, does not require any predefined functional form nor rely on any probability distribution, and the variables can be either continuous or discrete. Databases are also easy to maintain and updated when it is necessary to add or remove samples. Indeed, k-NN does not require a training procedure, and in each surrogate evaluation the database $\mathscr{D}$ must be ranked in order to determine the nearest neighbors.

The similarity measure used here is based on the Euclidean distance and it is given by

$$s(x^h, x^i) = 1 - \frac{d_E(x^h, x^i)}{d_E(x^U, x^L)}$$

where $d_E(x, y)$ is the Euclidean distance between x and y.

The nearest neighbors (and its variations) have been applied in two-dimensional interpolation [54], supervised learning [62], and recently in forest management planning [55].

9.3.2 Surrogate-Assisted Framework

Once a surrogate model has been chosen, there are many ways of introducing it into the evolutionary process. Some of them include: integrating GAs with surrogate approximations [40, 44] or landscape approximations [29], the use of surrogate-guided evolutionary operators [42], surrogate-assisted local search [33, 60], accelerating the optimization process using surrogates, pre-selection approaches [19, 39], multiple surrogates [1, 33, 50], and coevolution of fitness predictors [53].

In this chapter we introduce the surrogate models into the evolutionary cycle by means of a model management procedure [24] which, in each generation, uses in a cooperative way both surrogate and exact models, so that the evaluation of the population does not rely entirely on the surrogate model.

Maintaining a total of $N_{f,max}$ exact evaluations, surrogate model evaluations are introduced in the GA in increasing levels, by decreasing the parameter p_{sm}. The number of generations performed by the GA is given by $N_G = \frac{N_{f,max}}{p_{sm}\lambda}$. When $p_{sm} = 1$, all individuals are evaluated by the exact function, one has $N_G = N_{f,max}/\lambda$, and the standard GA is recovered. Indeed, as p_{sm} decreases, more surrogate evaluations are introduced into the evolutionary optimization process.

```
procedure Pre-selection (PS)
if p_sm ≠ 1 then
   repeat
      Evaluate individual using f̂
      N_f̂ = N_f̂ + 1
   until all  individuals in G_t evaluated
   Rank G_t according to the surrogate model
end if
repeat
   Evaluate individual using f
   N_f = N_f + 1
until all p_sm  best individuals in G_t evaluated
```

Fig. 9.2 Pre-selection (PS) management procedure. p_{sm} is the fraction of individuals evaluated by the original model, λ is the population size, f and $\widehat{f}$ are the original and surrogate functions, N_f the current number of exact evaluations and $N_{\widehat{f}}$ is the current number of surrogate evaluations

In the model management used here, only a fraction $0 < p_{sm} \leq 1$ of the population is evaluated by the time-consuming original model. We implement a pre-selection (PS) [19] strategy, where the surrogate model is used to decide which individuals will be evaluated by the original function. This procedure is described as follows: first, using evolutionary operators, λ individuals in the offspring population G_t are generated from λ parents in the parent population P_t. Then the offspring population G_t is entirely evaluated by the surrogate model and then ranked in decreasing order of quality. Based upon this rank, the $p_{sm}\lambda$ highest ranked individuals (according to the surrogate model $\widehat{f}$) are evaluated by the original model, and the remaining $\lambda - p_{sm}\lambda$ individuals in G_t maintain their objective function predicted by the surrogate model $\widehat{f}$. The PS model management procedure is shown in Figure 9.2.

In the PS model management it is not necessary that the surrogate model approximates the objective function closely. It is sufficient that the ranking of the individuals in the offspring population be similar to the ranking that would be obtained using the simulation model.

9.3.3 The Surrogate-Assisted Evolutionary Algorithm

The similarity-based surrogate-assisted GA for computationally expensive optimization problems, is shown in Figure 9.3. The developed algorithm will be referred to as SBSM-GA. The variant developed for single-objective optimization is named SBSM-SOGA while the multi-objective one is referred to as SBSM-MOGA. The differences between them are (i) the ranking procedures (line 5 and 12) and (ii) the parent population update procedure (line 13).

In the presented algorithm, we adopted the standard floating-point coding: each variable is encoded as a real number and concatenated to form a vector which is an individual in the population of candidate solutions. The following step is to randomly generate an initial population. Each individual has then one or more objective function values assigned to it and, in cases of constrained optimization, also a

1: **procedure** SBSM-GA
2: $t \leftarrow 0$; $N_f \leftarrow 0$; $N_{\widehat{f}} \leftarrow 0$; $\mathscr{D} \leftarrow \emptyset$
3: Generate an initial population P_t with λ individuals
4: Evaluate P_t using the simulation model f
5: Rank P_t in decreasing order of quality
6: Initialize the set $\mathscr{D} \leftarrow P_t$
7: **while** $N_f \leq N_{f,max}$ **do**
8: Select parents from P_t
9: Generate a population G_t from P_t
10: Apply the model management (Figure 9.2).
11: Update the set $\mathscr{D}$
12: Rank P_t in decreasing order of quality
13: Update parent population P_{t+1} from P_t and G_t
14: $t \leftarrow t+1$
15: **end while**

Fig. 9.3 Similarity-based surrogate-assisted GA (SBSM-GA). Pseudo-code for single- (SBSM-SOGA) or multi-objective (SBSM-MOGA) optimization. P_t is the parent population, G_t is the offspring population, λ is the population size, f and $\widehat{f}$ are the original and surrogate functions. $N_{f,max}$ is maximum number of exact evaluations, N_f is the current number of exact evaluations and $N_{\widehat{f}}$ is the number of surrogate evaluations

measure of constraint violation associated with it. The population is sorted in order to establish a "ranking". Individuals are then selected for reproduction in a way that better performing solutions have a higher probability of being selected. The genetic material contained in the chromosome of such "parent" individuals is then recombined and mutated, by means of crossover and mutation operators, giving rise to offspring which will form a new generation of individuals. Finally, the whole process is repeated until a termination criterion is attained.

Elitism is applied in the parent population update procedure (line 13): some individuals of the parent population are saved to the offspring population before the new parent population is created. In the single-objective version (SBSM-SOGA), the best ranked individual of the parent population P_t is copied to the offspring population G_t.

In single-objective constrained optimization ($n_i \neq 0$), we use a constraint handling technique presented in [10] to guide the search toward the (feasible) optimum. The individuals are ranked according to a pair-wise comparison procedure, where the following criteria are enforced:

1. when two feasible solutions are compared, the one with better objective function value is chosen,
2. when one feasible and one infeasible solutions are compared, the feasible solution is chosen, and
3. when two infeasible solutions are compared, the one with smaller constraint violation is chosen.

The constraint violation is given by $\sum_{j=1}^{n_i} \max(0, g_j(x))^2$.

The surrogate-assisted multi-objective GA (SBSM-MOGA) uses the operators from the Non-dominated Sorting Genetic Algorithm (NSGA-II) [11]. The multi-objective version of the algorithm differs from the single-objective version in the following aspects: (i) the ranking procedure, which uses the fast non-dominated sorting algorithm and the crowding comparison operator, and (ii) the elitism mechanism used in the parent population update procedure.

The ranking procedure that appears in lines 5 and 12 of the Figure 9.3 is replaced by the non-dominated sorting procedure [11], where the population is first partitioned by means of nondominated sorting and, then, a crowding comparison operator is employed by considering distances between individuals of the same rank. The update procedure shown in line 13 is performed over the union of the parent and offspring populations. The offspring population G_t is added to the parent population P_t and the combined population (of size 2λ) is ranked according to non-domination, then the highest ranked individuals are copied to the next generation.

The constraint handling technique for multi-objective optimization problems is based on the constraint-domination criteria [11], where feasible solutions have a better non-domination rank than any infeasible solution. A solution i is said to constraint-dominate a solution j, if any of the following conditions is true:

1. Solution i is feasible and solution j is not.
2. Solution i and solution j are both infeasible, but solution i has a smaller overall constraint violation.
3. Solution i dominates solution j

To improve the quality of the approximations in Eq. (9.4) the surrogate models are updated along the optimization process, by updating the set $\mathscr{D}$. In the SBSM-GA, all individuals exactly evaluated are recorded into the set $\mathscr{D}$, and when the maximum size η of the set is reached, the oldest individual is chosen to be replaced. As a result, one has a relatively small and updated sample set, as older individuals are discarded from $\mathscr{D}$ as the population evolves. The set size is equal to λ in the first generation (line 6 of the algorithm 9.3) and limited to η individuals along the evolutionary process.

In order to avoid convergence to false optima, and the need to re-evaluate the best solutions in each generation, after the ranking procedure (either for single- or multi-objective version), a sorting algorithm is applied in order to ensure that individuals evaluated by the original function are ranked highest in the population.

9.4 Computational Experiments

The algorithmic parameters for both SBSM-SOGA and SBSM-MOGA are summarized in Table 9.1.

We remark that, as described in Table 9.1, we have set a lower bound $p_{sm} = 1/\lambda$. For single-objective problems, we must have $p_{sm} \geq 1/\lambda = 1/40 = 0.025$, and $p_{sm} \geq 1/\lambda = 1/50 = 0.02$ for multi-objective problems. In the computational experiments of this section, we have set $p_{sm} \geq 0.05$.

Table 9.1 Algorithmic parameters setting for SBSM-GA (single- and multi-objective optimization)

Algorithmic Parameters	
Population size (λ)	Single-obj. optimization problems: $\lambda = 40$ Multi-obj. optimization problems: $\lambda = 50$
Representation	Floating-point coding: vectors of real numbers.
Operators	*Single-obj. opt. problems*: Uniform mutation, [34], Heuristic, One- and Two-point crossover, [23], Rank-based selection [57] and Elitism (best individual copied to the next generation) *Multi-obj. opt. problems*: Uniform mutation, Heuristic, One- and Two-point crossover, Rank-based selection (fast-non-dominated sorting and crowding distance [11]), and Elitism (parent and offspring population mixed and sorted in order to create the next generation)
Stop criterium	Maximum number of *exact* evaluations, given by $N_{f,max}$.
Crossover Probability (p_c)	$p_{c,heu} = 0.54$ (Heuristic), $p_{c,1p} = 0.18$ (One-point) and $p_{c,2p} = 0.18$ (Two-Point)
Mutation Rate (p_m)	$p_m = 0.02$
Database size (η)	$\eta = \{\lambda,\ 2\lambda,\ 5\lambda,\ 15\lambda\}$ or DPR=$(\eta/\lambda) = \{1,\ 2,\ 5,\ 15\}$
Database update	Replace the oldest individual. Only individuals evaluated by the original function can replace individuals in the database $\mathscr{D}$.
Surrogate Model	Nearest Neighbors (k-NN)
Number of Neighbors (k)	$k \in \{1,2,5,10,15\}$
Model Management	Individual-based Pre-Selection (PS) [19]. At each generation, the offspring population G_t is entirely evaluated by the surrogate model and ranked in decreasing order of quality. The $p_{sm}\lambda$ highest ranked individuals (according to the surrogate model $\widehat{f}$) are evaluated by the original model, and the remaining $\lambda - p_{sm}\lambda$ individuals in G_t maintain their objective function predicted by the surrogate model $\widehat{f}$.
Fraction p_{sm}	$p_{sm} \in [0.05, 1.00]$. The parameter p_{sm} defines the fraction of individuals evaluated by the original model: $p_{sm} = 1$ means the standard GA (no surrogates) with $N_G = N_{f,max}/\lambda$ generations. As we must ensure at least one individual evaluated by the original model in each generation, we have $p_{sm} \geq 1/\lambda$.
Performance Measures	The performance of the SBSM-GA is to be compared to the Standard GA ($p_{sm} = 1$). *Single-obj. opt. problems*: The value of the objective function. For constrained problems, also the number of runs leading to feasible final solutions. *Multi-obj. opt. problems*: Generational Distance [59], Maximum Spread [33] and Spacing [20].
Number of runs	50

DPR: Database size Population size Ratio, with DPR $= \frac{\text{Database size}}{\text{Population size}} = \frac{\eta}{\lambda}$

As the surrogate evaluations are introduced into the Standard GA, errors due the surrogate model evaluations are also introduced, which may adversely affect the quality of the final solutions. On the other hand, the extra surrogate evaluations allow for a longer period to search for improved solutions. There is a trade-off beetwen the noise introduced by the surrogate models and the beneficial impact in increasing the number of generations. We recall that, given a budget of $N_{f,max}$ exact evaluations, as the parameter p_{sm} decreases, the number of generations increases according to $N_G = N_{f,max}/p_{sm}\lambda$.

It is assumed that for complex real-world applications the cost of a surrogate model evaluation is negligible when compared to that of a simulation, hence total computational time will be only slightly increased due to the extra surrogate evaluations.

9.4.1 Single-Objective Optimization

In this section we show the results obtained for unconstrained and constrained problems, using the GA assisted with the k-NN surrogate model.

The single-objective minimization problems are shown in Table 9.2, and the constrained optimization problems considered are shown in the Table 9.3. For the constrained problems, the bounds for each parameter, in each function, are defined in the Table 9.4. More details of this set of constrained problems can be found in [48].

In problems with a large number of constraints (n_i), similarity-based surrogate models are computationally interesting, since they do not require a training procedure, leading to a simple and inexpensive way to estimate the constraints of the individuals in the population.

9.4.1.1 Effects of the Number of Neighbors

In this first experiment, we study the impact of increasing the number of neighbors, given a fixed database size (fixed DPR), in order to choose an appropriate neighborhood size. Under a fixed DPR=$\eta/\lambda = 2$. the experiments were conducted

Table 9.2 Single-objective minimization problems. The maximum number of simulations is $N_{f,max}$, the lower and upper bounds are respectively x^U and x^L, n is the number of design variables, and f^* is the optimal objective function value

#	Objective function	$N_{f,max}$	n	$[x^L, x^U]$	f^*
F01	$\sum_{i=1}^{n} x_i^2$	1000	10	$[-5.12, 5.12]$	0
F02	$\sum_{i=1}^{n} (\lfloor x_i + 0.5 \rfloor)^2$	1000	10	$[-100, 100]$	0
F03	$\sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod_{i=1}^{n} \frac{\cos x_i}{\sqrt{i}} + 1$	1600	10	$[-600, 600]$	0
F04	$-20e^{-0.2\sqrt{\frac{\sum_{i=1}^{n} x_i^2}{n}}} - e^{\frac{\sum_{i=1}^{n} \cos(2\pi x_i)}{n}} + 20 + e$	1000	10	$[-32.768, 32.768]$	0
F05	$\sum_{i=1}^{n-1} 100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2$	2000	10	$[-5.12, 5.12]$	0
F06	$\sum_{i=1}^{n} i x_i^4 + U(0,1)$	1000	10	$[-4.28, 4.28]$	0
F07	$\sum_{i=1}^{n} (x_i^2 - 10\cos(2\pi x_i) + 10)$	2000	10	$[-10, 10]$	0
F08	$\sum_{i=1}^{n} -x_i \sin(\sqrt{\lvert x_i \rvert}) - 418.982887272433n$	1000	10	$[-500, 500]$	0

Table 9.3 Constrained minimization problems – The number of design variables is n. The constraints read $g_j = g_j(x) \leq 0,\ j = 1, \ldots, n_i$

#	Objective function	Constraints	n
G01	$5\sum_{i=1}^{4} x_i - 5\sum_{i=1}^{4} x_i^2 - \sum_{i=5}^{13} x_i$	$g_1 = 2x_1 + 2x_2 + x_{10} + x_{11} - 10$ $g_2 = 2x_1 + 2x_3 + x_{10} + x_{12} - 10$ $g_3 = 2x_3 + 2x_2 + x_{12} + x_{11} - 10$ $g_4 = -8x_1 + x_{10}$ $g_5 = -8x_2 + x_{11}$ $g_6 = -8x_3 + x_{12}$ $g_7 = -2x_4 - x_5 + x_{10}$ $g_8 = -2x_6 - x_7 + x_{11}$ $g_9 = -2x_8 - x_9 + x_{12}$	13
G02	$\frac{\sum_{i=1}^{n} \cos^4(x_i) - 2\prod_{i=1}^{n} \cos^2(x_i)}{\sqrt{\sum_{i=1}^{n} i x_i^2}}$	$g_1 = 0.75 - \prod_{i=1}^{n} x_i$ $g_2 = \sum_{i=1}^{n} x_i - 7.5n$	20
G04	$5.3578547x_3^2 + 0.8356891x_1$ $+x_2 37.293239x_1 - 40792.141$	$g_1 =\ 85.334407 + 0.0056858x_2x_5+$ $0.0006262x1x_4 - 0.0022053x_3x_5 - 92$ $g_2 =\ -85.334407 - 0.0056858x_2x_5-$ $0.0006262x1x_4 + 0.0022053x_3x_5 0$ $g_3 =\ 80.51249 + 0.0071317x_2x_5+$ $0.0029955x_1x_2 + 0.0021813x_3^2 - 110$ $g_4 =\ -80.51249 - 0.0071317x_2x_5-$ $0.0029955x_1x_2 - 0.0021813x_3^2 90$ $g_5 =\ 9.300961 + 0.0047026x_3x_5+$ $0.0012547x_1x_3 + 0.0019085x_3x_4 - 25$ $g_6 =\ -9.300961 - 0.0047026x_3x_5-$ $0.0012547x_1x_3 - 0.0019085x_3x_4 20$	5
G06	$(x_1 - 10)^3 + (x_2 - 20)^3$	$g_1 = -(x_1 - 5)^2 - (x_2 - 5)^2 + 100$ $g_2 = (x_1 - 6)^2 - (x_2 - 5)^2 + 82.81$	2
G07	$x_1^2 + x_2^2 + x_1x_2 - 14x_1 - 16x_2+$ $(x_3 - 10)^2 + 4(x_4 - 5)^2+$ $(x_5 - 3)^2 + 2(x_6 - 1)^2+$ $5x_7^2 + 7(x_8 - 11)^2+$ $2(x_9 - 10)^2 + (x_{10} - 7)^2 + 45$	$g_1 = -105 + 4x_1 + 5x_2 + 3x_7 + 9x_8$ $g_2 = 10x_1 - 8x_2 - 17x_7 + 2x_8$ $g_3 = -8x_1 + 2x_2 + 5x_9 - 2x_{10} - 12$ $g_4 = 3(x_1 - 2)^2 + 4(x_2 - 3)^2 + 2x_3^2 - 7x_4 - 120$ $g_5 = 5x_1^2 + 8x_2 + (x_3 - 6)^2 - 2x_4 - 40$ $g_6 = x_1^2 + 2(x_2 - 2)^2 - 2x_1x_2 + 14x_5 - 6x_6$ $g_7 = 0.5(x_1 - 8)^2 + 2(x_2 - 4)^2 + 3x_5^2 - x_6 - 30$ $g_8 = -3x_1 + 6x_2 + 12(x_9 - 8)^2 - 7x_{10}$	10
G08	$\frac{\sin(2\pi x_1)\sin(2\pi x_2)}{x_1^3(x_1+x_2)}$	$g_1 = x_1^2 - x_2 + 1$ $g_2 = 1 - x_1 + (x_2 - 4)^2$	2
G09	$(x_1 - 10)^2 + 5(x_2 - 12)^2+$ $xqb_3^4 + 3(x_4 - 11)^2+$ $10x_6^5 + x_6^2 + x_7^4-$ $4x_6x_7 - 10x_6 - 8x_7$	$g_1 = -127 + 2x_1^2 + 3x_2^4 + x_3 + 4x_4^2 + 5x_5$ $g_2 = -282 + 7x_1 + 3x_2 + 10x_3^2 + x_4 - x_5$ $g_3 = -196 + 23x_1 + x_2^2 + 6x_6^2 - 8x_7$ $g_4 = 4x_1^2 + x_2^2 - 3x_1x_2 + 2x_3^2 + 5x_6 - 11x_7$	7
G10	$x_1 + x_2 + x_3$	$g_1 = -1 + 0.0025(x_4 + x_6)$ $g_2 = -1 + 0.0025(x_5 + x_7 - x_4)$ $g_3 = -1 + 0.01(x_8 - x_5)$ $g_4 = -x_1x_6 + 833.33252x_4 + 100x_1 - 83333.333$ $g_5 = -x_2x_7 + 1250x_5 + x_2x_4 - 1250x_4$ $g_6 = -x_3x_8 + 1250000 + x_3x_5 - 2500x_5$	8

Table 9.4 Bound constraints for single-objective constrained optimization problems. The maximum number of simulations is $N_{f,max}$ and f^* is the optimal objective function value

Function	Bound constraints	$N_{f,max}$	f^*
G_{01}	$0 \le x_i \le 1$ $(i = 1,\ldots,9)$, $0 \le x_i \le 100$ $(i = 10,11,12)$, $0 \le x_{13} \le 1$	600	-15
G_{02}	$0 \le x_i \le 10$ $(i = 1,\ldots,n)$ $n = 20$	1200	-0.80355
G_{04}	$78 \le x_1 \le 102$, $33 \le x_2 \le 45$, $27 \le x_i \le 45 (i = 3,4,5)$	6000	-30665.539
G_{06}	$13 \le x_1 \le 100$, $0 \le x_2 \le 100$	2400	-6961.81388
G_{07}	$-10 \le x_i \le 10 (i = 1,\ldots,10)$	1000	24.3062091
G_{08}	$0 \le x_1, x_2 \le 10$	8000	-0.095825
G_{09}	$-10 \le x_i \le 10 (i = 1,\ldots,7)$	800	680.6300573
G_{10}	$100 \le x_1 \le 10000$, $1000 \le x_i \le 10000 (i = 2,3)$, $10 \le x_i \le 1000 (i = 4,...,8)$	3000	7049.3307

for $k = \{1,2,5,15\}$ neighbors and the averaged fitness in 50 runs was used as performance measure.

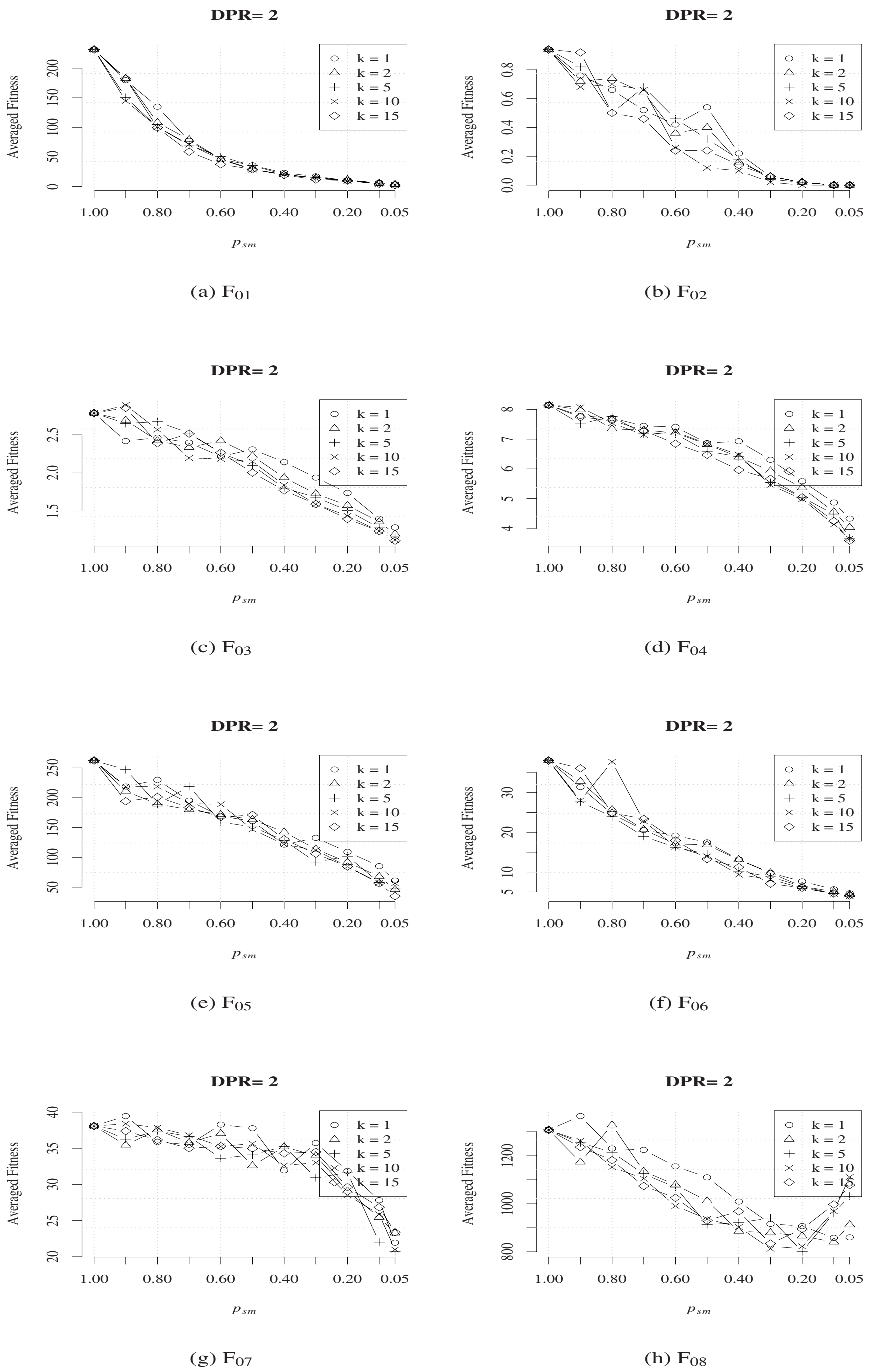
The neighborhood size affects the surrogate model in a way that small neighborhood leads to estimates very close to the data in the database $\mathscr{D}$, while a larger neighborhood tends to smooth the surrogate output, resulting in estimates close to the mean of the data in $\mathscr{D}$ [4].

The results for the SBSM-SOGA applied to the single objective optimization problems in Tables 9.2 and 9.3 for different values of p_{sm}, and using 1, 2, 5, 10, and 15 neighbors are shown in Figures 9.4 and 9.5. In Figure 9.5, for each test-problem, the average of the objective function in 50 runs is displayed. The average was calculated considering only the feasible runs, i.e. those producing a final solution which does not violate the constraints in Eq. (9.1).

For the all unconstrained functions, except for F_{08}, as p_{sm} decreases, increasingly better results are obtained. For those functions, it is possible to use very small values of p_{sm}. In this set of experiments we set $p_{sm} = 0.05$, although we may use $p_{sm} > 1/40 = 0.025$, as described in Table 9.1. The results obtained for function F_{08}, show that improvements with respect to the Standard GA are obtained for p_{sm} values below a certain threshold value, and the maximum improvement (compared to the Standard GA) were obtained when $p_{sm} > 0.20$.

The same trend with respect to the number of neighbors and the parameter p_{sm} is observed for all unconstrained functions. We observe that the extra evaluations performed by the surrogate are beneficial to the evolutionary search, and improved results are obtained when the number of generations increases.

From the results obtained for function G_{08}, we can see that reducing p_{sm}, no longer improves the final results, which means that the noise introduced by the



(a) F_{01} (b) F_{02} (c) F_{03} (d) F_{04} (e) F_{05} (f) F_{06} (g) F_{07} (h) F_{08}

Fig. 9.4 Averaged Fitness for different values of p_{sm}, with DPR=2, using 1, 2, 5, 10, and 15 neighbors in the surrogate model shown in Eq. (9.4)

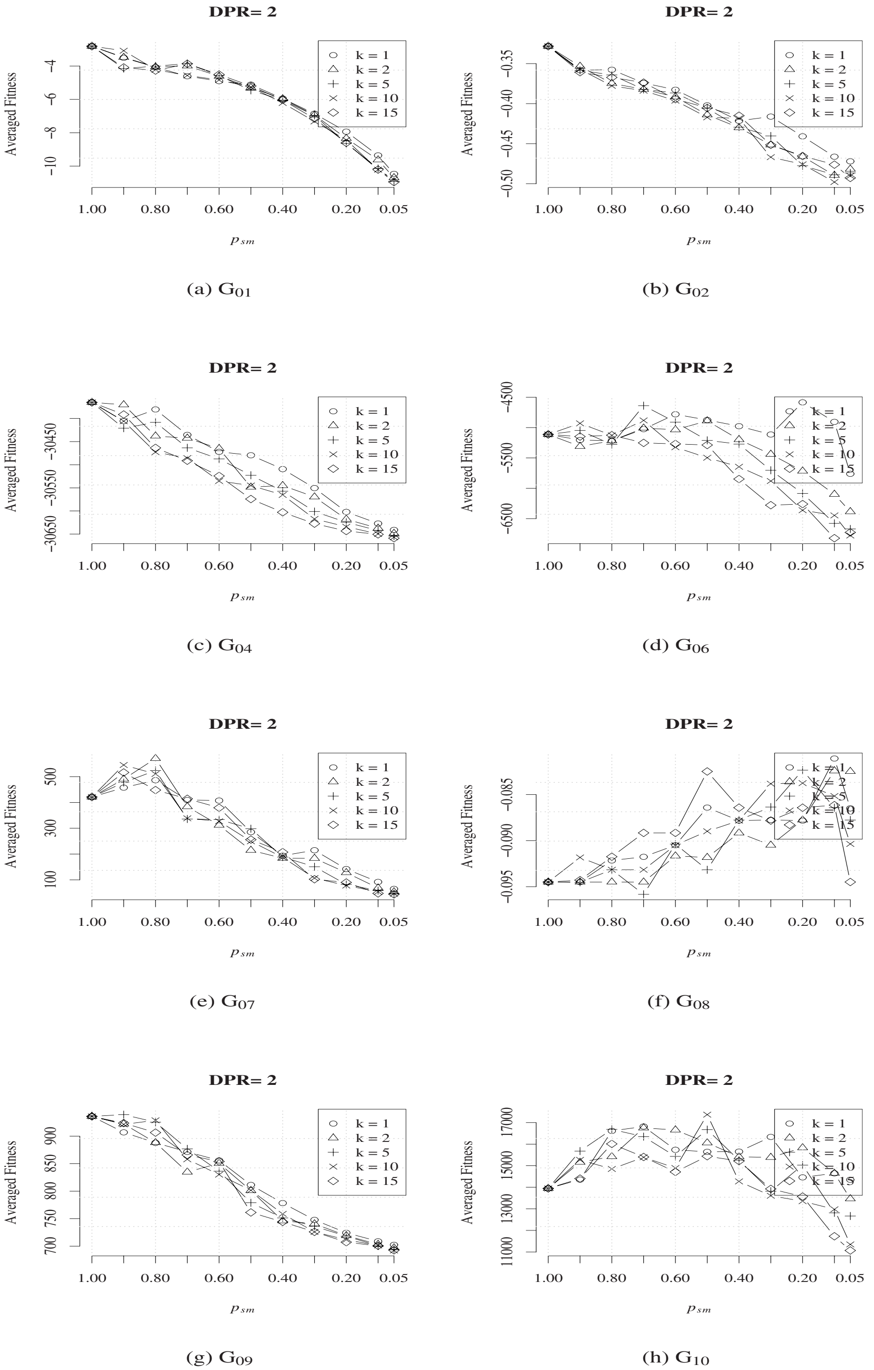


Fig. 9.5 Averaged Fitness for different values of p_{sm}, with DPR=2, using 1, 2, 5, 10, and 15 neighbors in the surrogate model shown in Eq. (9.4)

surrogate model affects the search in a negative way. Function G_{08}, corresponds to a complex landscape which could not be well approximated by the surrogate model. Although faster and simple, the k-NN surrogate model has limited capabilities to approximate complex mapping in $\Re^n$, which, as an inner-product space, allows for other calculus-based approximation. However, when the search occurs in a metric space, k-NN may be one of the few available alternatives.

As observed in function G_{08}, the constraints make the problem harder for the SBSM-SOGA, since more approximations are involved (objective functions and constraints) and the use of surrogates may lead the evolutionary process to poorer regions of the search space.

The results displayed in Figure 9.5, except for function G_{06}, and for G_{08} (where no improvements were obtained), show that the number of neighbors does not significantly affect the performance of the SBSM-SOGA for the set of functions considered here.

Table 9.5 shows the number of feasible runs for the SBSM-GA. The results were obtained using $k = 2$ neighbors and DPR=2 to build the surrogate in Eq. (9.4). We observe that the introduction of the surrogate does not affect the number of feasible runs, except in test-problem G_{06}, where a slightly decrease occurs. In G_{01} and G_{10}, the SBSM-GA increased the number of feasible runs.

Table 9.5 Constrained optimization problems – Number of runs that produce a final feasible solution with respect to the parameter p_{sm}. The results were obtained using 2 neighbors and DPR=2 to build the surrogate in Eq. (9.1)

p_{sm}	G_{01}	G_{02}	G_{04}	G_{06}	G_{07}	G_{08}	G_{09}	G_{10}
1	12	50	50	50	46	50	50	20
0.9	13	50	50	49	42	50	50	15
0.8	25	50	50	49	45	50	50	20
0.7	29	50	50	47	49	50	50	20
0.6	43	50	50	48	50	50	50	15
0.5	48	50	50	49	49	50	50	27
0.4	50	50	50	47	50	50	50	28
0.3	50	50	50	47	50	50	50	33
0.2	50	50	50	48	50	50	50	39
0.1	50	50	50	46	50	50	50	41
0.05	50	50	50	47	50	50	50	40

In frameworks that use surrogates as a local search tools or to enhance operators, the improvements are directly related to the surrogate models. In this set of experiments, the contribution of the surrogates to the evolutionary search is indirect: the surrogates allow for an extended number of generations (although with inexact evaluations), which provided the GA a longer period to evolve solutions.

9.4.1.2 Effects of the Database Size

In this section, a study of the impact of the database size on the evolutionary process is performed. Based on the experiments presented in the previous section, we set the neighborhood size to $k = 2$ and we perform experiments for DPR=$\{1,2,5,15\}$, corresponding to $\eta = \{1\lambda,\ 2\lambda,\ 5\lambda,\ 10\lambda,\ 15\lambda\}$ where $\eta = |\mathscr{D}|$.

Figures 9.6 and 9.7 display the results obtained by the SBSM-SOGA. We observe that for G_{06} larger values of η improve the results for smaller p_{sm}. The remaining test-problems are not affected by the database size. Except for G_{06}, we observe the same trend for all unconstrained and constrained functions, independent of the database size.

One can verify that the negative impact of the surrogate model persists for test problem G_{08}: the average results become worse as p_{sm} decreases, independently of the size of $\mathscr{D}$.

The results suggest that, for single-objective problems, a smaller database $\mathscr{D}$, combined with smaller values of p_{sm} are enough to improve the final solutions found by the SBSM-SOGA (when compared to the Standard GA, where $p_{sm} = 1$). However as the ruggedness/complexity of the optimization problem increases, and when constraints are involved (requiring more surrogate approximations), the performance may be not satisfactory, leading in some cases to deteriorated final solutions.

The results presented in sections 9.4.1.1 and 9.4.1.2 suggest to use a small value of the parameter p_{sm}. For SO problems, where one has no previous knowledge, we suggest as an initial trial $p_{sm} = 0.20$. Indeed, the results are indifferent to the database size, and we sugest a database size $\eta = 2\lambda$ (DPR=2).

9.4.2 Multi-objective Optimization

In this section we present and discuss the performance of the SBSM-MOGA when applied to constrained and unconstrained multi-objective problems.

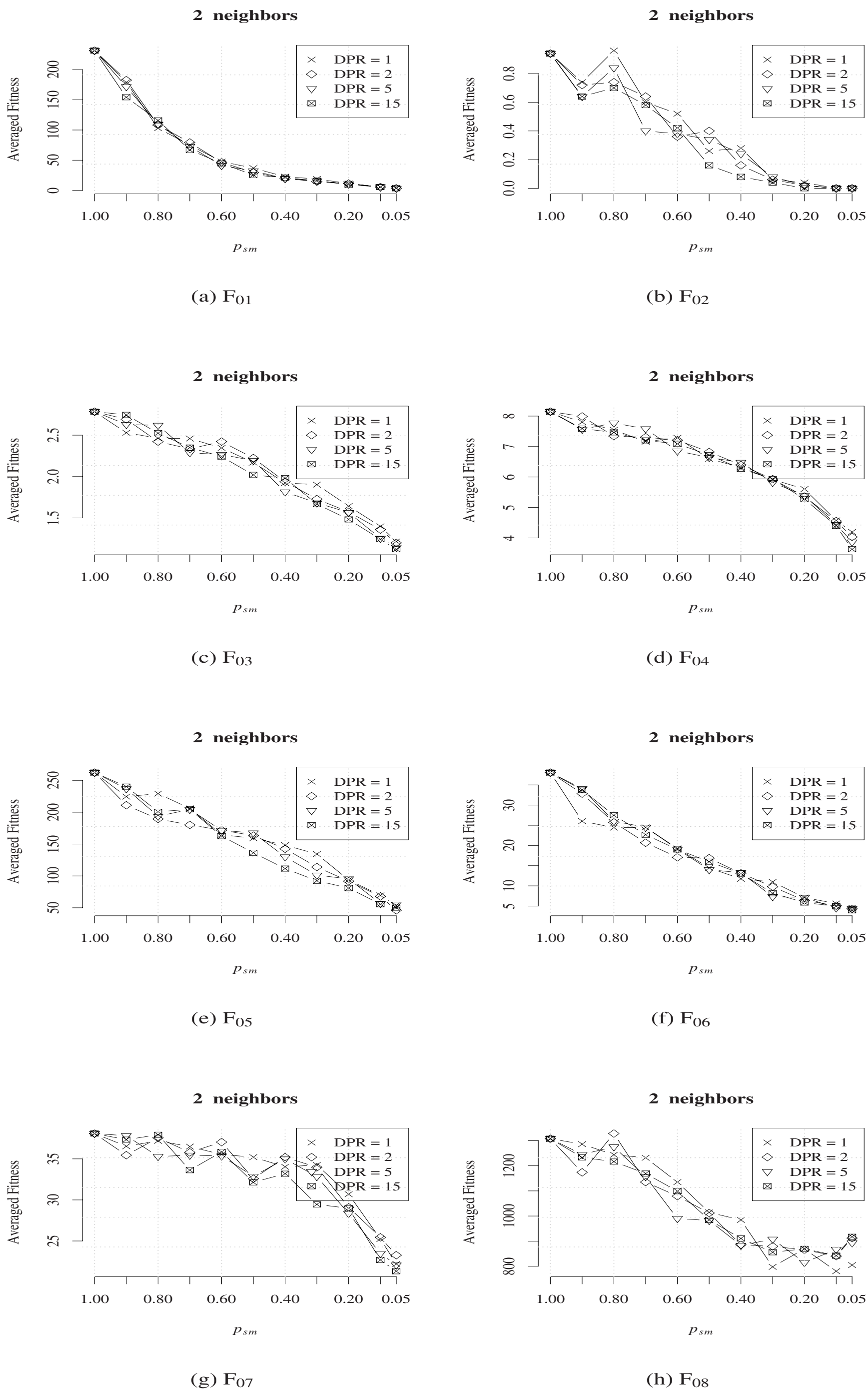
A total of 14 MO problems (8 unconstrained and 6 constrained) were collected [9] to study the impact of the surrogates into the SBSM-MOGA. Tables 9.6 and 9.7 show respectively the multi-objective unconstrained and constrained optimization problems, the bounds for each parameter, the constraints (for the constrained ones), the number of variables, and the maximum number of evaluations. Details can be found in [9].

In order to investigate the impact of the surrogate models in multi-objective optimization, we use as performance metrics the Generational Distance indicator (*GD*) [59], the Maximum Spread [33] and Spacing [20].

The *GD* indicator measures the gap between the evolved Pareto front (PFE) and the true Pareto front (PFT), given by

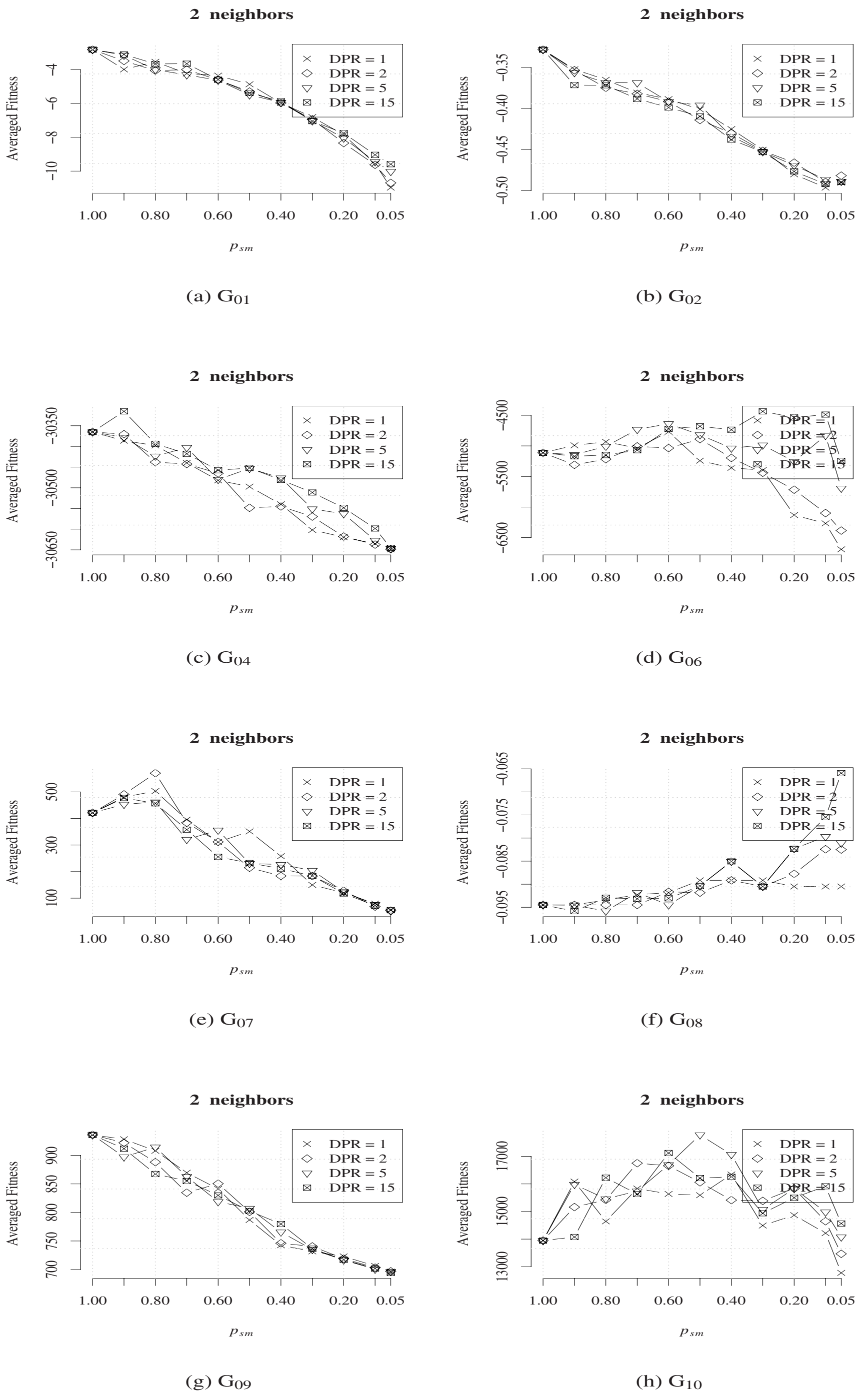
$$GD = \sqrt{\frac{1}{N_{PF}} \sum_{j=1}^{N_{PF}} d_j^2} \tag{9.5}$$

where N_{PF} is the number of individuals in PFT, d_j is the Euclidean distance (in the objective space) beetwen an individual j in PFE and its nearest individual in PFT. The generational distance in Eq. (9.5) measures the convergence to the true Pareto front, and lower values of *GD* are better.



(a) F_{01} (b) F_{02} (c) F_{03} (d) F_{04} (e) F_{05} (f) F_{06} (g) F_{07} (h) F_{08}

Fig. 9.6 Surrogate-assisted single-objective evolutionary unconstrained optimization



(a) G_{01} (b) G_{02} (c) G_{04} (d) G_{06} (e) G_{07} (f) G_{08} (g) G_{09} (h) G_{10}

Fig. 9.7 Surrogate-assisted single-objective evolutionary constrained optimization

Table 9.6 Unconstrained multi-objective optimization problems. The maximum number of simulations is $N_{f,max}$, the lower and upper bounds are respectively x^U and x^L, and n is the number of design variables

#	Objective Functions	n	$[x^L, x^U]$	$N_{f,max}$
MF_{01}	$f_1(x) = x_1$ $f_2(x) = 1 - \sqrt{f_1(x)/g(x)}$ $g(x) = 1 + 9(\sum_{i=2}^{n} x_i)/(n-1)$	30	$[0,1]$	1000
MF_{02}	$f_1(x) = x_1$ $f_2(x) = g(x)\left[1 - (f_1(x)/g(x))^2\right]$ $g(x) = 1 + 9(\sum_{i=2}^{n} x_i)/(n-1)$	30	$[0,1]$	1000
MF_{03}	$f_1(x) = x_1$ $f_2(x) = g(x)\left[1 - \sqrt{f_1(x)/g(x)} - (f_1(x)/g(x))\sin 10\pi f_1\right]$ $g(x) = 1 + 9(\sum_{i=2}^{n} x_i)/(n-1)$	30	$[0,1]$	1000
MF_{04}	$f_1(x) = x_1$ $f_2(x) = 1 - \sqrt{f_1(x)/g(x)}$ $g(x) = 1 + 10(n-1) + 9(\sum_{i=2}^{n}(x_i^2 - 10\cos 4\pi x_i)$	10	$x_1 \in [0,1]$, $x_i \in [-5,5]$ $i = 2,\ldots,10$	1000
MF_{05}	$f_1(x) = 0.5x_1x_2(1+g(x))$ $f_2(x) = 0.5(1-x_2)(1+g(x))$ $f_3(x) = 0.5(1-x_1)(1+g(x))$ $g(x) = 1000 + 100\sum_{i=3}^{n}[(x_i-0.5)^2 - \cos 20\pi(x_i-0.5)]$	12	$[0,1]$	2000
MF_{06}	$f_1(x) = 1 - \exp(-4x_1)\sin^6(6\pi x_1)$ $f_2(x) = g(x)[1 - f_1(x)/g(x)^2]$ $g(x) = 1 + 9[\sum_{i=2}^{n} x_i/(n-1)]^{0.25}$	10	$[0,1]$	1000
MF_{07}	$f_1(x) = \cos\frac{\pi}{2}x_1 \cos\frac{\pi}{2}x_2(1+g(x))$ $f_2(x) = \cos\frac{\pi}{2}x_1 \sin\frac{\pi}{2}x_2(1+g(x))$ $f_3(x) = \sin\frac{\pi}{2}x_1(1+g(x))$ $g(x) = \sum_{i=3}^{n}(x_i-0.5)^2$	12	$[0,1]$	1000
MF_{08}	$f_1(x) = \cos\frac{\pi}{2}x_1 \cos\frac{\pi}{2}x_2(1+g(x))$ $f_2(x) = \cos\frac{\pi}{2}x_1 \sin\frac{\pi}{2}x_2(1+g(x))$ $f_3(x) = \sin\frac{\pi}{2}x_1(1+g(x))$ $g(x) = 1000 + 100\sum_{i=3}^{n}[(x_i-0.5)^2 - \cos 20\pi(x_i-0.5)]$	12	$[0,1]$	1400

The Maximum Spread (MS) is used to measure how well the true Pareto front PFT is covered by the evolved Pareto front PFE. A larger value of MS reflects that a larger area of the PFT is covered by PFE. The MS is given as

$$MS = \sqrt{\frac{1}{n_{obj}} \sum_{i=1}^{n_{obj}} \left[\frac{\min(f_i^{max}, F_i^{max}) - \max(f_i^{min}, F_i^{min})}{F_i^{max} - F_i^{min}} \right]^2} \tag{9.6}$$

where f_i^{max} and f_i^{min} are the maximum and minimum of the i^{th} objective in the evolved Pareto front, respectively, and F_i^{max} and F_i^{min} the maximum and minimum of the i^{th} objective in the true Pareto front, respectively.

Table 9.7 Constrained multi-objective optimization problems. The maximum number of simulations is $N_{f,max}$, the lower and upper bounds are respectively x^U and x^L, and n is the number of design variables. The constraints are $g_j = g_j(x) \le 0,\ j = 1,\dots,n_i$

#	Objective functions	Constraints	n	Domain	$N_{f,max}$
MG_{01}	$f_1 = -2x_1 + x_2$ $f_2 = +2x_1 + x_2$	$g_1 = -x_1 + x_2 - 1$ $g_2 = +x_1 + x_2 - 7$	2	$0 \le x_1 \le 5$ $0 \le x_2 \le 3$	1000
MG_{02}	$f_1 = (x_1-2)^2 + (x_2-1)^2 - 2$ $f_2 = 9x_1 + (x_2-1)^2$	$g_1 = x_1^2 + x_2^2 - 225$ $g_2 = x_1 - 3x_2 + 10$	2	$[-20, 20]$	800
MG_{03}	$f_1 = x_1$ $f_2 = x_2$	$g_1 = 1 - x_1^2 - x_2^2 +$ $0.1\cos\left(16\arctan\frac{x_1}{x_2}\right)$ $g_2 = (x_1-0.5)^2 +$ $(x_2-0.5)^2 - 0.5$	2	$[0, \pi]$	4000
MG_{04}	$f_1 = -25(x_1-2)^2 + (x_2-2)^2 +$ $(x_3-1)^2 + (x_4-4)^2 +$ $(x_5-1)^2$ $f_2 = \sum_{i=1}^{n} x_i^2$	$g_1 = x_1 + x_2 - 2$ $g_2 = 6 - x_1 - x_2$ $g_3 = 2 - x_2 + x_1$ $g_4 = 2 - x_1 + 3 * x_2$ $g_5 = 4 - (x_3-3)^2 - x_4$ $g_6 = (x_5-3)^2 + x_6 - 4$	6	$0 \le x_1 \le 10$ $0 \le x_2 \le 10$ $1 \le x_3 \le 5$ $0 \le x_4 \le 6$ $1 \le x_5 \le 5$ $0 \le x_6 \le 10$	800
MG_{05}	$f_1 = -x_1$ $f_2 = -x_2$ $f_2 = -x_3$	$g_1 = -1 + \sum_{i=1}^{n} x_i^2$	3	$[0, 1]$	1200
MG_{06}	$f_1 = \frac{1}{10}\sum_{i=1}^{10} x_i$ $f_2 = \frac{1}{10}\sum_{i=11}^{20} x_i$ $f_3 = \frac{1}{10}\sum_{i=11}^{30} x_i$	$g_1 = 1 - f_3 - 4f_1$ $g_2 = 1 - f_3 - 4f_2$ $g_3 = 1 - 2f_3 - f_1 - f_2$	30	$[0, 1]$	2000

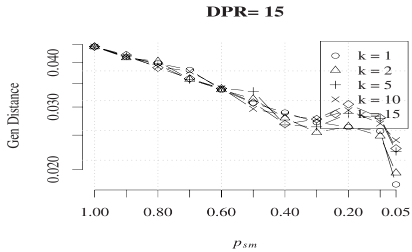
The metric of Spacing (S) shows how the nondominated solutions are distributed along the evolved Pareto front and is given as

$$S = \frac{1}{\hat{d}}\sqrt{\frac{1}{N_{PF}}\sum_{j=1}^{N_{PF}}(\hat{d} - d_j)^2}, \qquad \hat{d} = \frac{1}{N_{PF}}\sum_{k=1}^{N_{PF}} d_k \tag{9.7}$$

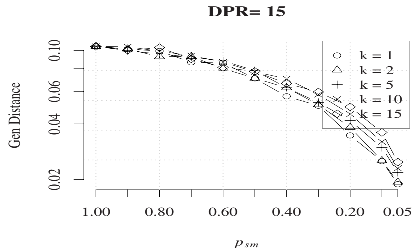
where N_{PF} is the number of individuals in PFT and d_i is the Euclidean distance (in the objective space) beetwen an individual i in the evolved Pareto front PFE and its nearest individual in the true Pareto front PFT.

9.4.2.1 Effects of the Number of Neighbors

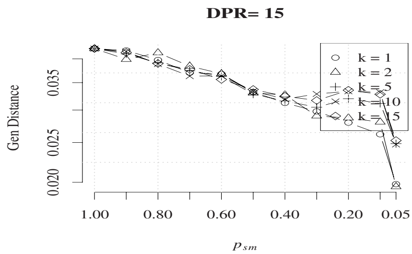
In this section we analyze the impact of the number of neighbors to the evolutionary search, given a fixed database size. According to the database replacement policy, the oldest individual is always chosen to be replaced. However, by removing solutions according to age, we may inevitably remove some important information. In order to alleviate this effect, we enlarge the training size for MO problems, and set the database size to $\eta = 15\lambda$, which corresponds to DPR=15. This value of



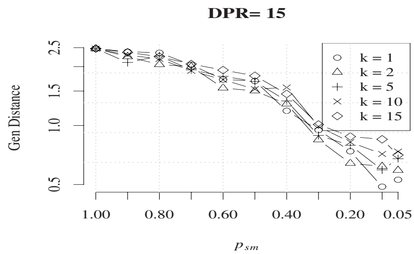
(a) MF_{01}



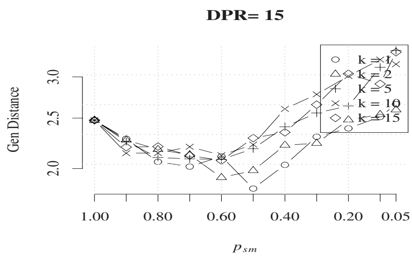
(b) MF_{02}



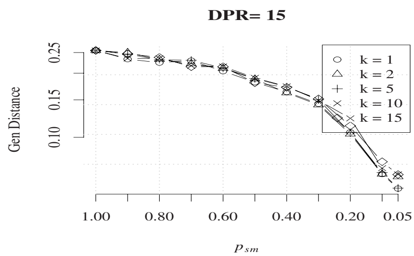
(c) MF_{03}



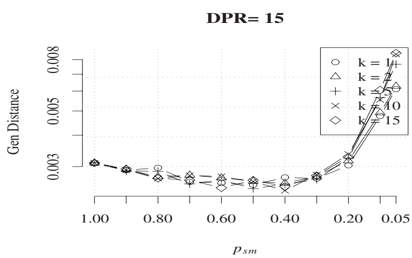
(d) MF_{04}



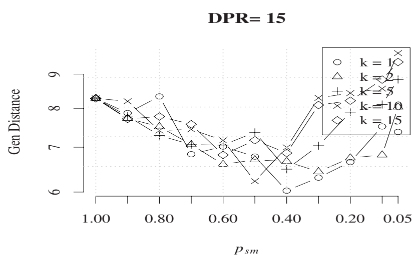
(e) MF_{05}



(f) MF_{06}

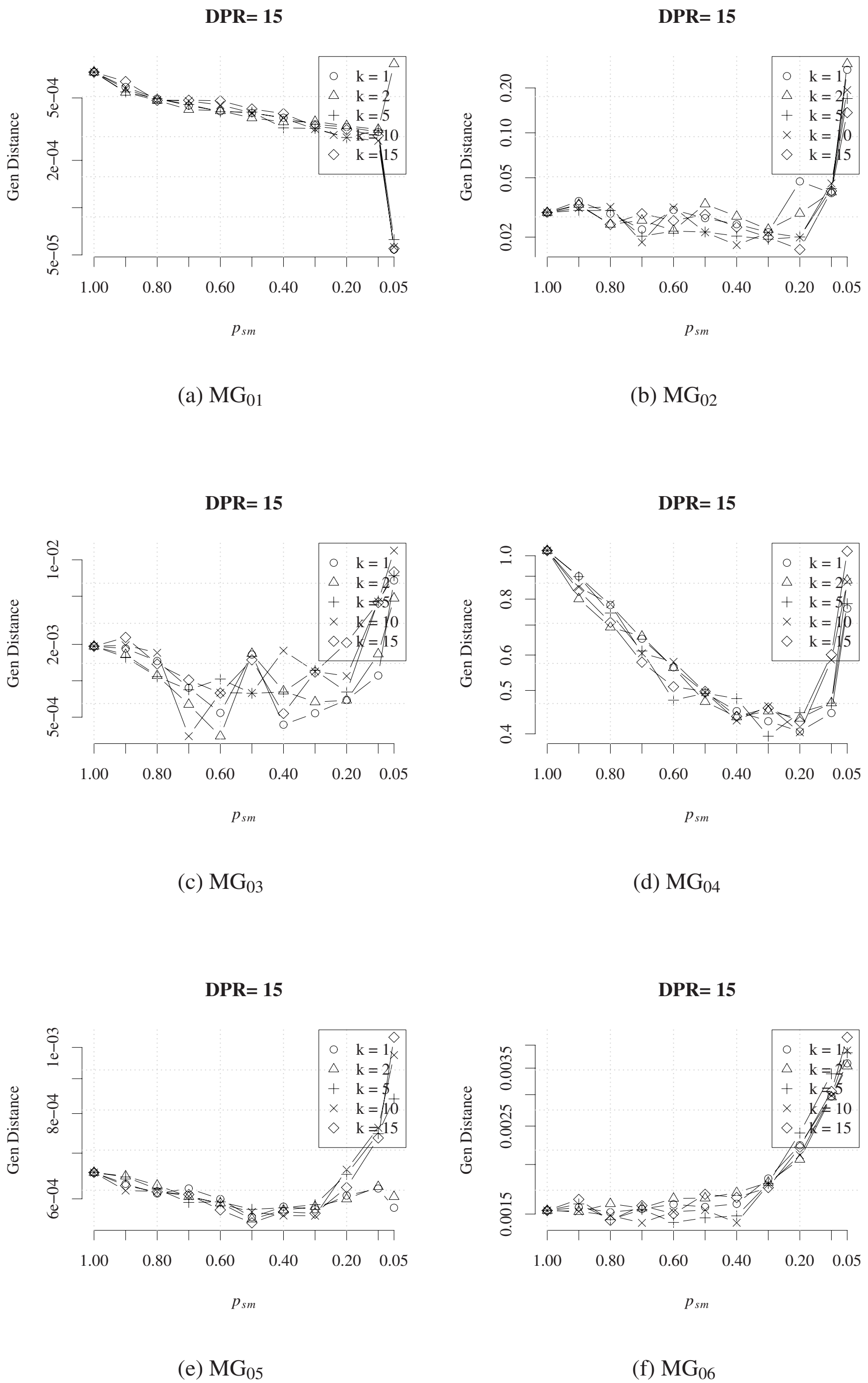


(g) MF_{07}



(h) MF_{08}

Fig. 9.8 Generational Distance (*GD*) indicator: surrogate-assisted multi-objective optimization using DPR=15 and $k = \{1,\ 2,\ 5,\ 10,\ 15\}$ neighbors

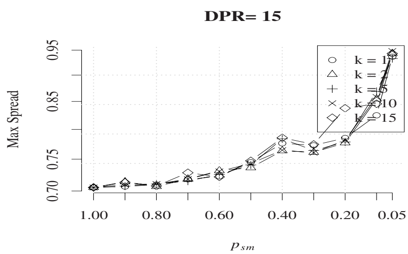


(a) MG_{01} (b) MG_{02}

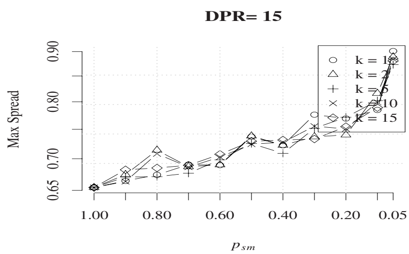
(c) MG_{03} (d) MG_{04}

(e) MG_{05} (f) MG_{06}

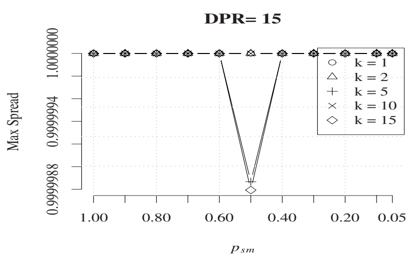
Fig. 9.9 Generational Distance (*GD*) indicator: surrogate-assisted multi-objective optimization using DPR=15 and $k = \{1,\ 2,\ 5,\ 10,\ 15\}$ neighbors



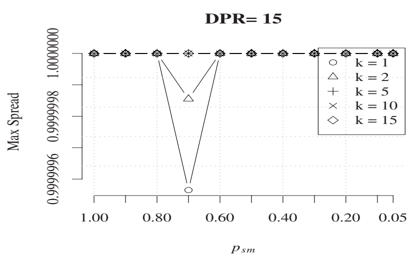
(a) MF_{01}



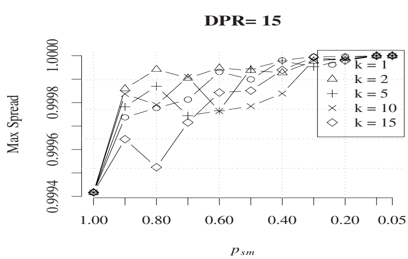
(b) MF_{03}



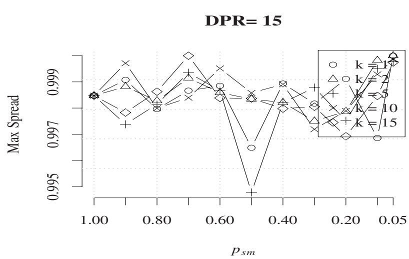
(c) MF_{05}



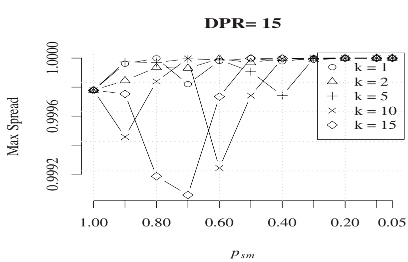
(d) MF_{08}



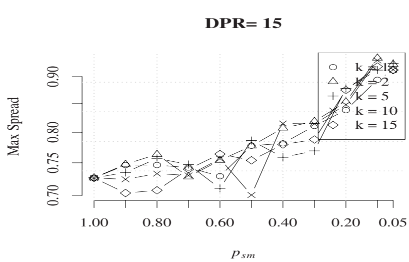
(e) MG_{01}



(f) MG_{02}

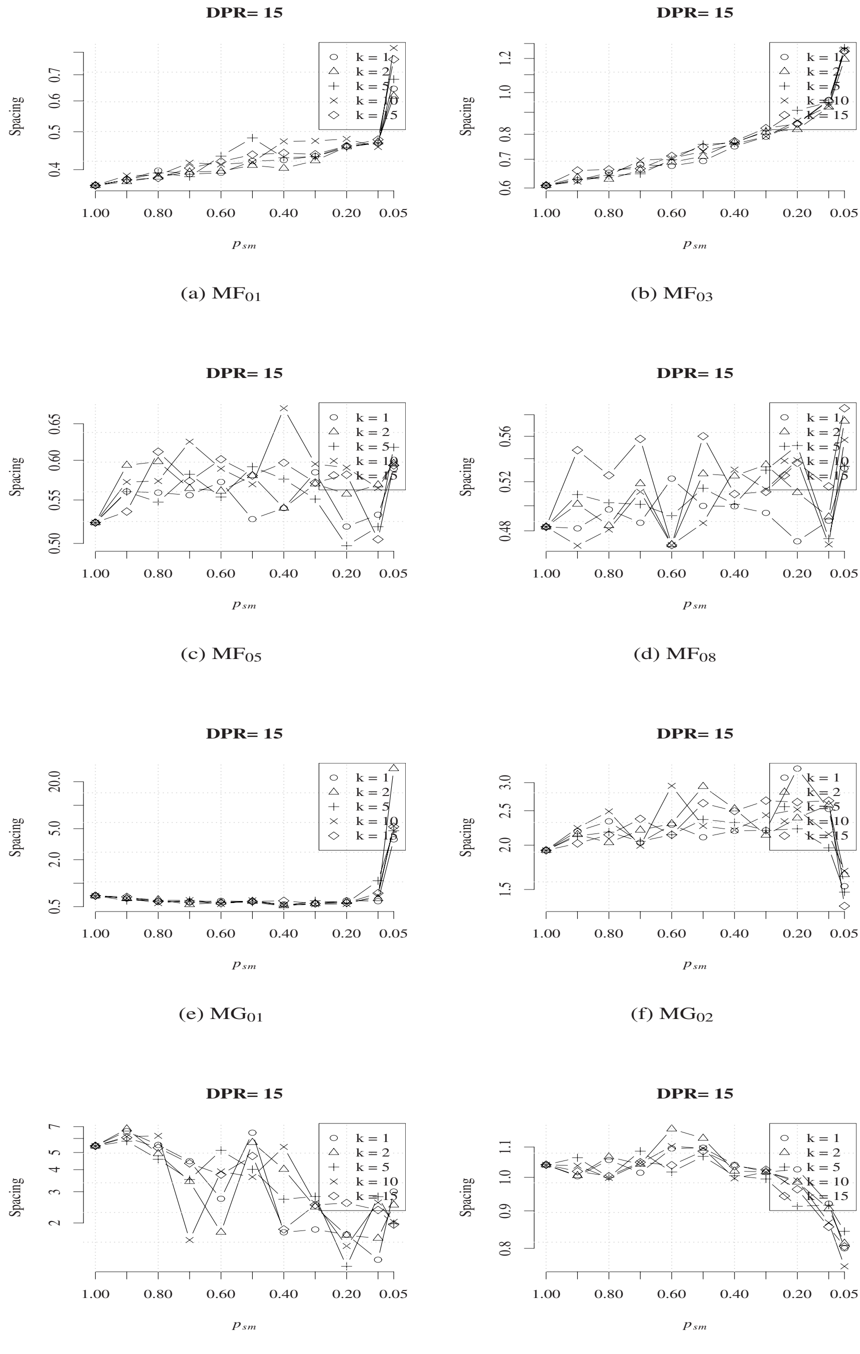


(g) MG_{03}



(h) MG_{06}

Fig. 9.10 Maximum Spread (*MS*): surrogate-assisted multi-objective optimization using DPR=15 and $k = \{1,\ 2,\ 5,\ 10,\ 15\}$ neighbors



(a) MF_{01} (b) MF_{03}

(c) MF_{05} (d) MF_{08}

(e) MG_{01} (f) MG_{02}

(g) MG_{03} (h) MG_{06}

Fig. 9.11 Spacing (S): surrogate-assisted multi-objective optimization using DPR=15 and $k = \{1,\ 2,\ 5,\ 10,\ 15\}$ neighbors

DPR results in storing information of at least 15 past generations, considering only individuals evaluated by the simulation model are stored in $\mathcal{D}$.

Figures 9.8 and 9.9 show the Generational Distance (*GD*) values, calculated for the (final) population at the end of the evolutionary process.

We observe that the SBSM-MOGA produced better results (compared to the Standard GA) depending on the values of the parameter p_{sm}. Except for the test-problems MF_{05}, MF_{07}, and MF_{08}, we observe that lower values of p_{sm} allow for final solutions closer to the true Pareto front. Also, the performance does not vary significantly as we change the number of neighbors used in the surrogate model. For function MG_{06} we observe that the surrogate model is not able to consistently help the GA in searching for improved solutions.

Figures 9.10 and 9.11 show the values of the Maximum Spread (*MS*) and Spacing (*S*) metrics, respectively, for a group of functions from those shown in Tables 9.6 and 9.7. From the results presented in Figure 9.10, we observe that, independently of the number of neighbors, smaller values of p_{sm} improve the performance of the SBSM-MOGA in the *MS* metric for MF_{01}, MF_{03}, MG_{01} and MG_{06}, and for MF_{05}, MF_{08}, MG_{02}, MG_{03} the *MS* is slightly affected when decreasing the parameter p_{sm}. Considering the Spacing metric, decreasing p_{sm} consistently improves the solutions in test-problems MF_{01}, MF_{03}, and MG_{01}.

9.5 Concluding Remarks

In this chapter we have proposed the introduction of a similarity-based surrogate model into a real-coded GA to assist the optimization of single- and multi-objective, constrained and unconstrained optimization problems, under a fixed computational budget.

We used the nearest neighbor approximation as a surrogate model, which is integrated into the evolutionary cycle by means of an individual-based evolution control where the surrogate is used to select individuals to be evaluated by the exact function according to a single parameter p_{sm}.

Instead of existing frameworks where the surrogates are used to improve the performance of evolutionary operators or as local search tools, here we use them to allow for an augmented number of generations to evolve solutions.

The tests performed so far support the following general conclusions:

Single-objective optimization: The augmented number of generations leads to improved solutions, when compared to the standard GA with the same number of expensive evaluations. Also, the number of neighbors does not affect in a significant way the final results, and a uniform trend is observed for unconstrained and constrained problems, as the parameter p_{sm} decreases. Also, the final results are not affected by the database size, which stores individuals previously evaluated by the simulation model.

Multi-objective optimization: For the set of multi-objective unconstrained optimization problems considered, small values of the parameter p_{sm} help to achieve a better convergence to the true Pareto front, according to the performance metrics, and the results are not significantly affected by the number of neighbors used.

In the nearest neighbor approximation model no training procedure is required and the prediction involves finding the nearest neighbors in an archive of previously evaluated individuals. Under a fixed number of expensive simulations, the cost of the surrogate-assisted procedure is only slightly increased due to the negligible computational cost of the extra surrogate evaluations as the cost of the expensive simulation increases.

The framework presented here seems to be a simple and effective way to tackle single- and multi-objective unconstrained or constrained expensive optimization problems. Additionally, the proposed framework can be easily extended to other population-based metaheuristics, such as Differential Evolution, Ant Colony Optimization and Particle Swarm Optimization.

References

1. Acar, E., Rais-Rohani, M.: Ensemble of metamodels with optimized weight factors. Struct. Multidisc. Optim. 37(3), 279–294 (2009)
2. Aha, D.W.: Editorial. Artif. Intell. Rev. 11(1-5), 1–6 (1997); special issue on lazy learning
3. Akbarzadeh-T, M.R., Davarynejad, M., Pariz, N.: Adaptive fuzzy fitness granulation for evolutionary optimization. International Journal of Approximate Reasoning 49(3), 523 (2008)
4. Altman, N.S.: An introduction to kernel and nearest-neighbor nonparametric regression. The American Statistician 46(3), 175–185 (1992)
5. Blanning, R.W.: The source and uses of sensivity information. Interfaces 4(4), 32–38 (1974)
6. Bui, L.T., Abbass, H.A., Essam, D.: Fitness inheritance for noisy evolutionary multi-objective optimization. In: GECCO 2005: Proceedings of the 2005 conference on Genetic and evolutionary computation, pp. 779–785. ACM, New York (2005)
7. Bull, L.: On model-based evolutionary computation. Soft Computing 3(2), 76–82 (1999)
8. Chen, J.H., Goldberg, D.E., Ho, S.Y., Sastry, K.: Fitness inheritance in multi-objective optimization. In: GECCO 2002: Proceedings of the Genetic and Evolutionary Computation Conference, pp. 319–326. Morgan Kaufmann Publishers Inc., San Francisco (2002)
9. Coello, C.A.C., Lamont, G.B., Veldhuizen, D.A.V.: Evolutionary Algorithms for Solving Multi-Objective Problems. Kluwer Academic Publishers, Norwell (2002)
10. Deb, K.: An Efficient Constraint Handling Method for Genetic Algorithms. Computer Methods in Applied Mechanics and Engineering 186(2/4), 311–338 (2000)
11. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation 6(2), 182–197 (2002)
12. Ducheyne, E., De Baets, B., de Wulf, R.: Is fitness inheritance useful for real-world applications? In: Fonseca, C.M., Fleming, P.J., Zitzler, E., Deb, K., Thiele, L. (eds.) EMO 2003. LNCS, vol. 2632, pp. 31–42. Springer, Heidelberg (2003)
13. Ducheyne, E., Baets, B.D., Wulf, R.D.: Fitness inheritance in multiple objective evolutionary algorithms: A test bench and real-world evaluation. Applied Soft Computing 8(1), 337–349 (2007)

14. El-Beltagy, M., Nair, P., Keane, A.: Metamodeling techniques for evolutionary optimization of computationally expensive problems: promises and limitations. In: Proceedings of Genetic and Evolutionary Conference, pp. 196–203. Morgan Kaufmann, Orlando (1999)
15. Emmerich, M., Giannakoglou, K., Naujoks, B.: Single- and multiobjective evolutionary optimization assisted by gaussian random field metamodels. Evolutionary Computation 10(4), 421–439 (2006)
16. Emmerich, M.T.M.: Single- and multi-objective evolutionary design optimization assisted by gaussian random field metamodels. PhD thesis, Technische Universitaet Dortmund (2005)
17. Ferrari, S., Stengel, R.F.: Smooth function approximation using neural networks. IEEE Transactions on Neural Networks 16(1), 24–38 (2005)
18. Forrester, A.I., Keane, A.J.: Recent advances in surrogate-based optimization. Progress in Aerospace Sciences 45, 50–79 (2009)
19. Giannakoglou, K.C.: Design of optimal aerodynamic shapes using stochastic optimization methods and computational intelligence. Progress in Aerospace Sciences 38(1), 43–76 (2002)
20. Goh, C.K., Tan, K.C.: A competitive-cooperative coevolutionary paradigm for dynamic multiobjective optimization. IEEE Transactions on Evolutionary Computation 13(1), 103–127 (2009)
21. Goldberg, D.: Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley Publishing Co., Reading (1989)
22. Grefenstette, J., Fitzpatrick, J.: Genetic search with approximate fitness evaluations. In: Proceedings of the International Conference on Genetic Algorithms and Their Applications, pp. 112–120 (1985)
23. Herrera, F., Lozano, M., Verdegay, J.L.: Tackling real-coded genetic algorithms: Operators and tools for behavioural analysis. Artificial Intelligence Review 12(4), 265–319 (1998)
24. Jin, Y.: A comprehensive survey of fitness approximation in evolutionary computation. Soft Computing Journal 9(1), 3–12 (2005)
25. Jin, Y., Branke, J.: Evolutionary optimization in uncertain environments-a survey. IEEE Transactions on Evolutionary Computation 9(3), 303–317 (2005)
26. Jin, Y., Olhofer, M., Sendhoff, B.: A framework for evolutionary optimization with approximate fitness functions. IEEE Transactions on Evolutionary Computation 6(5), 481–494 (2002)
27. Kecman, V.: Learning and soft computing: support vector machines, neural networks, and fuzzy logic models. Complex adaptive systems. MIT Press, Cambridge (2001)
28. Kim, H.S., Cho, S.B.: An efficient genetic algorithm with less fitness evaluation by clustering. In: Proceedings of the 2001 Congress on Evolutionary Computation, vol. 2, pp. 887–894 (2001)
29. Knowles, J.: Parego: A hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems. IEEE Transactions on Evolutionary Computation 10(1), 50–66 (2006)
30. Kybic, J., Blu, T., Unser, M.: Generalized sampling; a variational approach – Part I: Theory. IEEE Transactions on Signal Processing 50(8), 1965–1976 (2002)
31. Kybic, J., Blu, T., Unser, M.: Generalized sampling; a variational approach – Part II: Applications. IEEE Transactions on Signal Processing 50(8), 1977–1985 (2002)

32. Lim, D., Ong, Y., Jin, Y., Sendhoff, B.: A study on metamodeling techniques, ensembles, and multi-surrogates in evolutionary computation. In: Proceedings of the 9th annual conference on Genetic and evolutionary computation, pp. 1288–1295. ACM Press, New York (2007)
33. Lim, D., Jin, Y., Ong, Y.S., Sendhoff, B.: Generalizing surrogate-assisted evolutionary computation. IEEE Transactions on Evolutionary Computation (2008) (in press)
34. Michalewicz, Z.: Genetic Algorithms + Data Structures = Evolution Programs, 3rd edn. Springer, Heidelberg (1996)
35. Mota, F., Gomide, F.: Fuzzy clustering in fitness estimation models for genetic algorithms and applications. In: IEEE International Conference on Fuzzy Systems, pp. 1388–1395 (2006) ISBN: 0-7803-9488-7
36. Myers, R.H., Montgomery, D.C.: Response Surface Methodology – Process and Product Optimization Using Designed Experiments. Wiley Series in Probability and Statistics. John Wiley & Sons Inc., New York (2002)
37. Ong, Y., Nair, P., Keane, A.: Evolutionary optimization of computationally expensive problems via surrogate modeling. AIAA Journal 41(4), 687–696 (2003)
38. Pilato, C., Tumeo, A., Palermo, G., Ferrandi, F., Lanzi, P.L., Sciuto, D.: Improving evolutionary exploration to area-time optimization of FPGA designs. Journal of Systems Architecture 54(11), 1046 (2008)
39. Praveen, C., Duvigneau, R.: Low cost PSO using metamodels and inexact pre-evaluation: Application to aerodynamic shape design. Computer Methods in Applied Mechanics and Engineering 198(9-12), 1087–1096 (2009)
40. Queipo, N., Arévalo, C., Pintos, S.: The integration of design of experiments, surrogate modeling, and optimization for thermoscience research. Engineering with Computers 20, 309–315 (2005)
41. Queipo, N.V., Haftka, R.T., Shyy, W., Goela, T., Vaidyanathana, R., Tucker, P.K.: Surrogate-based analysis and optimization. Progress in Aerospace Sciences 41(1), 1–28 (2005)
42. Rasheed, K., Vattam, S., Ni, X.: Comparison of methods for using reduced models to speed up design optimization. In: Proceedings of Genetic and Evolutionary Computation Conference, pp. 1180–1187. Morgan Kaufmann, New York (2002)
43. Rasheed, K., Ni, X., Vattam, S.: Comparison of methods for developing dynamic reduced models for design optimization. Soft Computing Journal 9, 29–37 (2005)
44. Regis, R.G., Shoemaker, C.A.: Local function approximation in evolutionary algorithms for the optimization of costly functions. IEEE Trans. Evolutionary Computation 8(5), 490–505 (2004)
45. Reyes-Sierra, M., Coello, C.A.C.: A study of fitness inheritance and approximation techniques for multi-objective particle swarm optimization. In: The 2005 IEEE Congress on Evolutionary Computation, vol. 1, pp. 65–72 (2005)
46. Runarsson, T.: Approximate evolution strategy using stochastic ranking. In: Yen, G.G., Wang, L., Bonissone, P., Lucas, S.M. (eds.) IEEE World Congress on Computational Intelligence, Vancouver, Canada (2006)
47. Runarsson, T.P.: Constrained Evolutionary Optimization by Approximate Ranking and Surrogate Models. In: Yao, X., Burke, E.K., Lozano, J.A., Smith, J., Merelo-Guervós, J.J., Bullinaria, J.A., Rowe, J.E., Tiňo, P., Kabán, A., Schwefel, H.-P. (eds.) PPSN 2004. LNCS, vol. 3242, pp. 401–410. Springer, Heidelberg (2004)
48. Runarsson, T.P., Yao, X.: Stochastic Ranking for Constrained Evolutionary Optimization. IEEE Transactions on Evolutionary Computation 4(3), 284–294 (2000)

49. Salami, M., Hendtlass, T.: A fast evaluation strategy for evolutionary algorithms. Applied Soft Computing 2, 156–173 (2003)
50. Sanchez, E., Pintos, S., Queipo, N.: Toward an optimal ensemble of kernel-based approximations with engineering applications. Structural and Multidisciplinary Optimization, 1–15 (2007)
51. Sastry, K., Goldberg, D.E., Pelikan, M.: Don't evaluate, inherit. Tech. Rep. IlliGAL Report No. 2001013, Illinois Genetic Algorithms Laboratory (IlliGAL), Department of General Engineering, University of Illinois at Urbana-Champaign (2001)
52. Sastry, K., Pelikan, M., Goldberg, D.E.: Efficiency enhancement of genetic algorithms via building-block-wise fitness estimation. In: Congress on Evolutionary Computation, CEC 2004, pp. 720–727 (2004)
53. Schmidt, M., Lipson, H.: Coevolution of fitness predictors. IEEE Transactions on Evolutionary Computation 12(6), 736–749 (2008)
54. Shepard, D.: A two-dimensional interpolation function for irregularly-spaced data. In: Proceedings of the 1968 23rd ACM National Conference, pp. 517–524. ACM Press, New York (1968)
55. Sironen, S., Kangas, A., Maltamo, M., Kalliovirta, J.: Localization of growth estimates using non-parametric imputation methods. Forest Ecology and Management 256, 674–684 (2008)
56. Smith, R.E., Dike, B.A., Stegmann, S.A.: Fitness inheritance in genetic algorithms. In: SAC 1995: Proceedings of the 1995 ACM symposium on Applied computing, pp. 345–350. ACM Press, New York (1995)
57. Sokolov, A., Whitley, D., Barreto, A.M.S.: A note on the variance of rank-based selection strategies for genetic algorithms and genetic programming. Genetic Programming and Evolvable Machines 8(3), 221–237 (2007)
58. Srinivas, N., Deb, K.: Multiobjective optimization using nondominated sorting in genetic algorithms. Evolutionary Computation 2(3), 221–248 (1994)
59. Van Veldhuizen, D.A., Lamont, G.B.: Evolutionary computation and convergence to a pareto front. In: Koza, J.R. (ed.) Late Breaking Papers at the Genetic Programming 1998 Conference, Stanford University Bookstore, University of Wisconsin, Madison, Wisconsin, USA, Stanford, CA, USA (1998)
60. Wanner, E.F., Guimaraes, F.G., Takahashi, R.H.C., Lowther, D.A., Ramirez, J.A.: Multiobjective memetic algorithms with quadratic approximation-based local search for expensive optimization in electromagnetics. IEEE Transactions on Magnetics 44(6), 1126–1129 (2008)
61. Yang, D., Flockton, S.J.: Evolutionary algorithms with a coarse-to-fine function smoothing. In: IEEE International Conference on Evolutionary Computation, vol. 2, pp. 657–662 (1995)
62. Zhang, J., Yim, Y.S., Yang, J.: Intelligent selection of instances for prediction functions in lazy learning algorithms. Artif. Intell. Rev. 11(1-5), 175–191 (1997)
63. Zheng, X., Julstrom, B.A., Cheng, W.: Design of vector quantization codebooks using a genetic algorithm. In: Proceedings of 1997 IEEE International Conference on Evolutionary Computation, Piacataway, NJ, pp. 525–530 (1997)
64. Zhou, Z., Ong, Y.S., Nair, P.B.: Hierarchical surrogate-assisted evolutionary optimization framework. In: Congress on Evolutionary Computation, pp. 1586–1593. IEEE, Los Alamitos (2004)