Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2014, Article ID 184540, 10 pages
http://dx.doi.org/10.1155/2014/184540

*Research Article*

# Geometric Generalisation of Surrogate Model-Based Optimisation to Combinatorial and Program Spaces

**Yong-Hyuk Kim,[1] Alberto Moraglio,[2] Ahmed Kattan,[3] and Yourim Yoon[4]**

[1] *Department of Computer Science and Engineering, Kwangwoon University, Nowon-gu, Seoul 139-701, Republic of Korea*
[2] *Department of Computer Science, Streatham Campus, University of Exeter, Exeter EX4 4QF, UK*
[3] *Computer Science Department, Umm Al-Qura University, Makkah 21955, Saudi Arabia*
[4] *Department of Computer Engineering, Gachon University, Seongnam-si, Gyeonggi-do 461-701, Republic of Korea*

Correspondence should be addressed to Yourim Yoon; yryoon@gachon.ac.kr

Received 14 March 2014; Accepted 31 March 2014; Published 29 April 2014

Academic Editor: Ker-Wei Yu

Copyright © 2014 Yong-Hyuk Kim et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Surrogate models (SMs) can profitably be employed, often in conjunction with evolutionary algorithms, in optimisation in which it is expensive to test candidate solutions. The spatial intuition behind SMs makes them naturally suited to continuous problems, and the only combinatorial problems that have been previously addressed are those with solutions that can be encoded as integer vectors. We show how radial basis functions can provide a generalised SM for combinatorial problems which have a geometric solution representation, through the conversion of that representation to a different metric space. This approach allows an SM to be cast in a natural way for the problem at hand, without ad hoc adaptation to a specific representation. We test this adaptation process on problems involving binary strings, permutations, and tree-based genetic programs.

## 1. Introduction

Some optimisation problems have objective functions which are prohibitively expensive to evaluate [1, 2]. Functions may be mathematically ill behaved (e.g., discontinuous, nonlinear, or nonconvex) or even a black box with largely unknown characteristics. Many engineering design problems have functions of this type [3, 4] and require experiments, lengthy simulations or both, to evaluate the extent to which the design objectives are met by a function of parameters controlling the design. In the jargon of evolutionary computation, these controlling parameters are the genotype that encodes the design (i.e., the phenotype) which has to be expressed by means of an expensive simulation (i.e., a fitness evaluation).

Optimisation methods based on surrogate models (SMs), also known as response surface models, can tackle this problem of expensive objective functions [5–7]. A survey of surrogate model-based optimisation (SMBO) methods can be found elsewhere [8]. An SM is an easily evaluated mathematical model that approximates an expensive objective function as precisely as possible. Inside knowledge of the objective function is not necessary to construct an SM, which is solely built from discrete evaluations of the expensive objective function. We refer to a pair of a candidate solution and its known objective function value as a data-point. Many simple problems have solutions which are real numbers, and perhaps the simplest example of an SM is piecewise-linear interpolation, which creates a function from data-points by linking them with straight-line segments. More useful SMs for solutions on the real line are polynomial interpolants, which have a continuous differential. These and other methods of building SMs naturally extend to spatial interpolation and regression.

The usual SMBO procedure [8] is given in Algorithm 1. An initial SM is constructed from a few solutions of the expensive objective function. Further evaluations are applied to candidate solutions which the SM predicts to be promising. Subsequently, the processes of searching the SM to obtain an optimum set of solutions, evaluation of the solutions using the expensive objective function, and update of the SM with the new data-points are repeated. An evolutionary algorithm can be used in the SMBO procedure to infer the location of

(1) Sample uniformly at random a small set of candidate solutions and evaluate them using the expensive objective function (initial set of data-points)
(2) **while** a limit on the number of expensive function evaluations has not been reached **do**
(3) Construct a new surrogate model (SM) using all data-points available
(4) Determine the optimum value of the SM by search, for example, using an evolutionary algorithm (this is feasible as the model is cheap to evaluate)
(5) Evaluate the solution which optimises the SM using the expensive objective function (making an additional data-point available)
(6) **end while**
(7) Return the best solution found

Algorithm 1: Surrogate model-based optimisation (SMBO).

a promising set of solutions using the SM, rather than having to evaluate the expensive objective function. This is feasible because the computational cost of a complete run of the evolutionary algorithm on the SM is negligible (in the order of few seconds) with regard to the cost of evaluating a solution using the expensive objective function of the problem (in the order of minutes, hours, or even days depending on the problem).

Virtually all SMs are implicitly or explicitly spatial models, and the prediction process involves exploiting some assumed spatial relations (e.g., a smooth curve of surface) between the values of the objective function at a query point and those at the known data-points. This makes SMBOs naturally suited to continuous optimisation problems. However they are not obviously applicable to combinatorial optimisation problems, except those with solutions which are naturally represented as vectors of integers, when a discretized SM may be used. When each solution is a vector, an integer, or a real number, techniques for building SMs from data-points can be borrowed from statistics (e.g., multivariate regression [9]) or from machine learning (e.g., supervised learning by neural networks or support vector machines [10–12]).

There is increasing interest in optimisation problems with solutions with complicated representations which also have expensive objective functions. For example, permutations and related representations are natural representations of solutions to many scheduling problems. But a candidate schedule may have to be tested by simulating an entire production process, making the SMBO approach very attractive. However, although a permutation can be regarded as a special type of vector, permutations cannot be treated in the same way, because the information they encode is in the order of the elements, not their values. This makes the standard SMBO approach unsuitable.

Variable-length sequences occur in many bioinformatics problems [13], and an SMBO can be used to select biological sequences for detailed study or simulation at an atomic level: an example is the search for proteins with desired properties.

Genetic programming (GP) [14] normally operated on a tree representation of a problem, and a number of its well-known applications have expensive objective functions. For example, genetic programs can be used to encode a robot's behavioral controller, which may need to be tested repeatedly in a virtual or real environment to assess how good it is at controlling the robot in performing a task such as wall-following or obstacle avoidance [15].

Let us summarize current situation of SM with regard to solution representations. Evolutionary algorithms and other search algorithms have been widely used to optimise SMs for continuous spaces [16]. More recent work [17] has considered vector solutions. Other studies [18] have approached applications with expensive objective functions which are inherently combinatorial problems with structured solutions (e.g., graphs) by encoding solutions in vector form to allow the use of standard SMs. Evolutionary algorithms have also been used to train, rather than search, the SM using the known data-points [19]; in the approach, GP performs symbolic regression to obtain the vector-input function which best fits the data-points.

Apart from the recent initial work of the present authors [20, 21], SMs do not seem to have been defined *directly* on more complicated representations than vectors. In order to use SMs on search problems with structured representations, the state of the art is to shoe-horn the original representation into a vector form in a preprocessing phase, known as feature extraction in the machine learning literature [22]. There are a number of drawbacks to this approach. For a start, feature extraction is a very delicate task. Only a carefully chosen vector of features will be a good representation of the information relevant to a learning task. Secondly, the unnatural encoding of a solution in vector form introduces extra nonlinearity into an already expensive objective function, making it harder to learn and consequently requiring additional expensive function evaluations to approximate it well enough to locate the optimum solution. In addition, the extraction of features from structured representations such as GP trees is itself unnatural and hence ineffective. For example, a symbolic regression formula or a Boolean formula would appear to have no obvious mapping to a fixed-length vector.

The underlying difficulty is that of making a problem fit the format of current SMs. Surely is it better to modify the SM to accommodate the problem? Or is there some way to modify satisfactory SMs to accept more complicated solution representations?

We recently [20, 21] answered these questions by generalizing a well-known class of SMs—radial basis function networks [23]—using a geometric framework [24–27] which had previously been used to generalize search algorithms, such

as particle swarm optimisation and differential evolution, from continuous spaces to combinatorial spaces. The generalization method is conceptually simple. Firstly, an algorithm which operated in a continuous space is rewritten in terms of Euclidean distances between points. Many spatial algorithms can be rewritten in this way. Then Euclidean distance is replaced with a generic distance metric, which yields a formally well-defined algorithm. This algorithm can be adapted to any solution representation by specifying an appropriate distance metric for that representation.

An algorithm generalised using this geometric methodology can readily be adapted to complicated representations because many types of structured object admit natural relations of distance or similarity. In particular *edit distances* are well suited to structured objects. The edit distance between two configurations is the minimum number of unit edit operations required to transform one of them into the other. For example, hamming distance is an edit distance between binary strings based on the unit edit of a bit flip. For permutations, another metric is swap distance, which is the minimum number of binary exchanges of elements required to transform one permutation into the other. For variable-length sequences, Levenshtein distance measures the minimum number of insertions, deletions, or changes of characters required to transform one sequence into the other. There are also edit distances defined on trees and graphs, based on modifications of edges and nodes.

In the remainder of this paper, we first review how radial basis function networks [23] can be generalised to a range of solution representations using this geometric methodology. We will show how the resulting generalised models can be linked to a target representation using an appropriate distance metric and then used within an SMBO to optimise problems on the target representation. We will illustrate the derivation of SMBOs for three target representations: binary strings, permutations, and GP trees. All our test problems are assumed to have costly objective functions. We use hamming distance as the metric for binary strings and test the resulting SMBO on the well-known NK-landscapes [28] problem. We use hamming distance and swap distance with permutations and test the SMBO on the quadratic assignment problem [29]. We use a form of tree edit distance with GP trees and address standard GP benchmarks of symbolic regression and parity. We should be clear that we are not aiming to show that a generalised SMBO can replace expensive objective functions with structured representations in solving practical problems, but to demonstrate that generalised SMBOs can be in principle applied to such problems, and that it provides meaningful results when applied to classic example problems in simple discrete spaces, which is itself a large conceptual leap.

## 2. Radial Basis Function Networks

The machine learning literature [22] contains a number of approaches to problems of finding a function in a certain class that best interpolates a set of the data-points which are naturally cast in terms of Euclidean distances, which could readily be generalised to other metric spaces, by replacing Euclidean distance with some metric. These methods include nearest-neighbor regression, inverse distance-weighted interpolation, radial basis function network interpolation, and Gaussian process regression (also known as kriging). The first two methods are relatively simple but they cannot be used as SMs because the global optimum of the functions created from the data-points coincides with a data-point used in the construction of these functions and these methods never provide better solutions than any of the data-points. Gaussian process regression [30] is a very powerful method with a solid theoretical foundation, which can not only extrapolate a global optimum but also give it an interval of confidence. Radial basis function network interpolation is similar to Gaussian process regression but conceptually simpler. We focus on radial basis function networks (RBFNs) and leave the generalization of Gaussian process regression for future work.

*2.1. Classic RBFNs.* A radial basis function (RBF) is a real-valued function $\phi : \mathbb{R}^n \to \mathbb{R}$ whose value depends only on the distance from some point $\mathbf{c}$, called its *center*, so that $\phi(\mathbf{x}) = \phi(\|\mathbf{x} - \mathbf{c}\|)$. The point $\mathbf{c}$ is an argument of the function. The norm is usually Euclidean, so $\|\mathbf{x}-\mathbf{c}\|$ is Euclidean distance between $\mathbf{c}$ and $\mathbf{x}$, but other norms are possible and have been used. Commonly used RBFs include Gaussian functions, multiquadrics, poly-harmonic splines, and thin-plate splines. The most frequently used are Gaussian functions of the form:

$$\phi(\mathbf{x}) = \exp\left(-\beta\|\mathbf{x} - \mathbf{c}\|^2\right), \tag{1}$$

where $\beta > 0$ is the width parameter.

RBFs are typically used to build function approximations of the form:

$$y(\mathbf{x}) = w_0 + \sum_{i=1}^{N} w_i \, \phi\left(\left\|\mathbf{x} - \mathbf{c}_i\right\|\right). \tag{2}$$

The approximating function $y(\mathbf{x})$ is thus the sum of $N$ RBFs, each associated with its own center $\mathbf{c}_i$, width $\beta_i$, and weighted by a coefficient $w_i$ and there is a bias term $w_0$. Figure 1 shows an example of a function obtained in this way. Any continuous function can in principle be approximated with arbitrary accuracy by such a sum, if enough RBFs are used.

In an RBFN, there are three types of parameters that need to be determined to optimise the fit between $y(\mathbf{x})$ and the data: the weights $w_i$, the centers $\mathbf{c}_i$, and the width parameters $\beta_i$. The most common way to find these parameters has two phases. Firstly, unsupervised learning (i.e., clustering) is used to determine the position of the centers and the widths of the RBFs. Then, the weights $w_i$ that optimise the fit are obtained by least-squares minimisation.

A simplified procedure for fitting an RBFN, which skips the unsupervised learning phase, is widely used. The centers $\mathbf{c}_i$ are first chosen to coincide with the known points $\mathbf{x}_i$. Then the widths $\beta_i$ are determined by a heuristic based on the distance of each center $\mathbf{c}_i$ to the nearest neighbors (local model) or all widths are set to the same value, which is chosen in relation to the maximum distance between any two centers

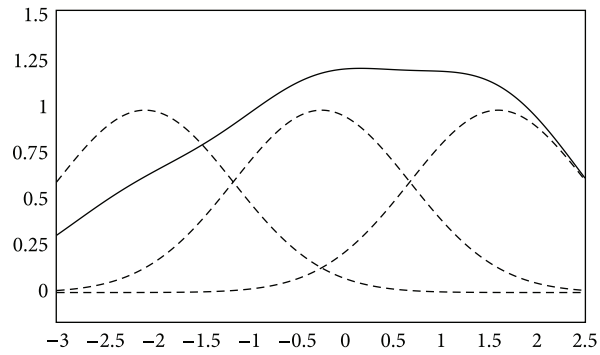



Figure 1: Example of a function (solid line) obtained as a weighted sum of three Gaussian functions (dashed lines) on the real line: the weighting factors $w_1$, $w_2$, and $w_3$ are 0.5, 1, and 1, respectively.

(global model). The bias $w_0$ can either be set to the mean of the function values $b_i$ at the known data-points (i.e., training set), or to 0. The weights $w_i$ are then determined by solving the system of $N$ simultaneous linear equations in $w_i$ which express the requirement that the function interpolates the data-points:

$$y(\mathbf{x}_i) = b_i, \quad i = 1 \cdots N. \tag{3}$$

Setting $g_{ij} = \phi(\|\mathbf{x}_j - \mathbf{x}_i\|)$, the system can be written in matrix form as $G\mathbf{w} = \mathbf{b}$. The matrix $G$ is nonsingular if the points $\mathbf{x}_i$ are distinct and the family of functions $\phi$ is positive definite (which is the case for Gaussian functions), and thus the weights $\mathbf{w}$ can be obtained by simple linear algebra:

$$\mathbf{w} = G^{-1}\mathbf{b}. \tag{4}$$

*2.2. Generalization of RBFNs to Arbitrary Representations.* To generalize RBFNs, we need to generalize (i) the class of functions used to approximate the unknown function, (ii) the training procedure which finds the function within that class that best fits the data-points, and (iii) the model query procedure that predicts the value of the unknown function at a query point.

Following the geometric methodology of our generalization, we first need to rewrite each of the above three elements as a function of Euclidean distance alone and then substitute a distance metric which is chosen to suit the target representation. Finally we rewrite the algorithm in terms of that distance to obtain an instance of that algorithm specific to the target representation.

Let $\mathbf{M}$ be a metric space associated with a distance function $d$. An RBF $\phi : \mathbb{R}^n \to \mathbb{R}$ whose value depends only on the distance from some point $\mathbf{c} \in \mathbb{R}^n$, so that $\phi(\mathbf{x}) = \phi(\|\mathbf{x}-\mathbf{c}\|)$, can be generalised to a function $\phi : \mathbf{M} \to \mathbb{R}$ whose value depends only on the distance from some point $\mathbf{c} \in M$ *in the metric space*, so that $\phi(\mathbf{x}) = \phi(d(\mathbf{x}, \mathbf{c}))$. For example, generalised Gaussian functions can be obtained by replacing Euclidean distance with the generic metric $d$ in the original definition, so that $\phi(\mathbf{x}) = \exp(-\beta d(\mathbf{x}, \mathbf{c})^2)$.

A set of configurations and an associated edit distance comprise a metric space, as all edit distances meet the metric axioms [27, 31, 32]. Consequently, a generalised RBF is well-defined on any set of configurations, making it a *representation-independent function*. For example, the set of binary strings $\mathbf{H}$ and hamming distance $d_H$ form a metric space. If hamming distance $d_H$ is used as the metric $d$, then generalised Gaussian functions become well-defined functions $\phi : \mathbf{H} \to \mathbb{R}$, which map binary strings to real numbers. Note that both $\mathbf{c}$ and $\mathbf{x}$ are binary strings. Alternatively, if the swap distance on permutations replaces the metric $d$, then these generalised Gaussian functions become well-defined functions mapping permutations to real numbers.

The SM $y(\mathbf{x})$, which is a linear combination of RBFs, can be generalised to a linear combination of generalised RBFs: $y(\mathbf{x}) = w_0 + \sum_{i=1}^{N} w_i\ \phi(d(\mathbf{x}, \mathbf{c}_i))$. Like its components, the generalised SM is representation independent and it can be applied to any solution representation by replacing the metric $d$ with a metric appropriate to the target representation. An SM is generalized in this way of parameterizing many functions on general metric spaces economically in terms of $\mathbf{c}_i$, $w_i$, and $\beta_i$. This property is independent of the underlying representation. When the underlying metric space is finite as it is in combinatorial optimisation, any function can be approximated with arbitrary accuracy by a sufficiently large number of RBFs. In the limit, every point in space would be associated with an RBF, parameterised to fit the function value exactly.

The SM is fitted to the known data-points without reference to their underlying representation but solely in terms of the distances between data-points and the objective function values $b_i$. Therefore the fitting process is representation independent, like the model. In particular, a simplified model-fitting procedure can obtain the centers, widths, and weights by least-squares minimisation of the system $G\mathbf{w} = \mathbf{b}$. However, when the distance function $d$ is not embeddable in Euclidean space, the RBFs are no longer necessarily positive definite, and neither is the matrix $G$, and hence the inverse matrix $G^{-1}$ needed to determine the weights $w_i$, may not exist. This difficulty can be overcome by using the pseudoinverse of $G$, which always exists, is unique, and corresponds to $G^{-1}$ when that exists. It can also be shown that the weights $w_i$ determined by solving the system $G\mathbf{w} = \mathbf{b}$ using the pseudoinverse are the same as those obtained by least-squares minimisation. This way of generalizing RBFNs to structured representations is related to kernel methods in machine learning. However, in those methods, the types of distances to be used between objects can be difficult to design, because they must be implicitly embedded in a vector space (i.e., positive-definite kernels), which is not necessary for our approach.

## 3. Experiments on Binary Strings

Binary strings are of course a special type of vector. However, they can illustrate the application of generalised SMBOs to combinatorial spaces because their property of being vectors is not utilised. We experimented with the well-known NK-Landscape problem [28], which provides a tunable set of rugged, epistatic landscapes over a space of binary strings,

and we consider it to have a costly objective function. We evaluated the SMBO algorithm with landscapes of size $n = 10, 15, 20, 25$, each for $k = 2, 3, 4, 5$.

We used a standard SMBO algorithm (Algorithm 1). The SM is an RBFN model fitted to the data-points using the simplified learning procedure presented in the previous section. The centers $\mathbf{c}_i$ of the RBFs are the data-points. The widths $\beta_i$ of the RBFs are all set to $1/2D^2$, where $D$ is the maximum distance between any two centers. Thus each RBF extends over all the centers, allowing the known function value at each center to contribute to the prediction of the function value at any point in the landscape near the given center. The value of the bias term $w_0$ is set to the average of the function values $b_i$ at all the known data-points. Thus the SM returns this value at any point outside the influence of all centers. The coefficients $w_i$ are determined by least-squares minimisation, as described in the previous section.

We set other parameters as a function of the problem size $n$. Our aim is to find the best solution to this problem with $2^n$ candidate solutions in quadratic time; that is, we set the number of allowable expensive function evaluations to $n^2$. Initially, 2 data-points are sampled, and $n^2 - 2$ sample points are suggested by the SM. To search the SM, we use a standard generational evolutionary algorithm with tournament selection with a tournament size of 2, uniform crossover at a rate of 0.5, and bitwise mutation at a rate of $1/n$. The population size and the number of generations are both set to $10n$. If the predicted value of the best solution found by the SM is better than the best value at any of the known data-points, then the model could extrapolate from the data, and that solution is evaluated using the expensive objective function. Otherwise, a point is chosen at random and evaluated with the expensive objective function in an attempt to gather more data about undersampled regions.

We compared SMBO with random search (RS), a standard (1 + 1) evolutionary algorithm ((1 + 1) EA), and a generational evolutionary algorithm (EA), all using the expensive objective function directly. We expect evolutionary algorithms to outperform random search, but we include the latter as it can do well with small samples. We allowed all the algorithms $n^2$ evaluations of the expensive objective function.

The (1 + 1) EA has a population of a single individual and uses bitwise mutation with a bit-flip probability of $1/n$. EA has a population of $n$ individuals, runs for $n$ generations, and uses tournament selection with tournament size 2, bitwise mutation with a bit-flip probability of $1/n$, and uniform crossover at a rate of 0.5. For each of the 16 combinations of $n$ and $k$, we generated a single fitness landscape and ran all for algorithms 50 times each. We also estimated the global optimum using an evolutionary algorithm with 1,000 individuals and 1,000 generations.

Table 1 shows that, for each combination of $n$ and $k$, SMBO consistently found the best solution and the best average solution. Furthermore, in 12 out of 16 cases, SMBO was able to find the estimated real optimum. As the problem size $n$ increases, the differential in favor of SMBO increases. As expected, as the ruggedness $k$ of the problem increases, search algorithms get less close to the estimated real optimum. As for the other algorithms in the comparison, the population-based EA generally did better than (1 + 1) EA and RS, especially on larger problems. Perhaps surprisingly, RS often did better than (1 + 1) EA. It seems that (1 + 1) EA can easily get trapped at local optima, especially when the sample and problem sizes are large.

## 4. Experiments on Permutations

This section greatly extends our previous work [21]. Experiments were carried out on six standard quadratic assignment problems (QAPs), kra30a, kra32, lipa30a, nug30, ste36a, and tho30 (where the number in the name indicates the problem size), and on two instances of a unimodal problem on permutations of size 30, in which the fitness of the permutation, to be minimised, is given by its distance to some fixed permutation. This unimodal problem can be seen as a generalization of the OneMax problem for binary strings [33], in which the fitness of a solution is the number of 1$s$ in the string. This is in turn equivalent to a problem in which the fitness of a solution is given by hamming distance from the solution to the string with all bits set to 1. From the symmetry of hamming space, this problem is again equivalent to any problem in which a string with all bits set to one is to be replaced with some target string. The two instances of the unimodal problem are obtained by using two different distance functions on permutations, hamming distance (unih30), and swap distance [27] (unis30). We address this unimodal problem to test the SMBO on a fitness landscape with an explicit and visible topography. We will consider the problems in the test-bed as having costly objective functions and leave as future work testing the SMBO on real-world problems with expensive objective functions. Furthermore, using a larger test-bed and testing the scalability of SMBO with respect to instance size would be desirable. However, we found that it would take an excessive amount of time, as the SM is searched every time it is used to suggest a solution to test in the expensive objective function. We will also consider a larger test-bed and a scalability analysis in future work.

The algorithm that uses hamming distance is called $\text{SMBO}_H$ and the algorithm using Swap distance is called $\text{SMBO}_S$. Clearly, the choice of a distance well suited to the problem at hand is crucial to obtain an SM able to make meaningful predictions and guide appropriately the search of the SMBO. In this paper, we limit ourselves to experiment with these two distances. In future work, we will investigate other distances and other problems in the attempt to find out a rule to select *a priori* a good distance for a given type of problem.

As in the previous section, that is, binary strings, we used a standard SMBO algorithm (Algorithm 1) with an RBFN model which is fitted to the available data-points using the simplified learning procedure. For $\text{SMBO}_H$, all the RBFs have the same widths $\beta = 1/2D^2$, where $D$ is the maximum distance across all centers. However, this setting did not work well for $\text{SMBO}_S$, and we found that $\beta = 1/(D/5)$ produced better results. The value of $\beta$ greatly affects the accuracy of the predictions of the SM. So, it needs to be tuned but might in

Table 1: Results for SMBO, an evolutionary algorithm (EA), a (1 + 1) evolutionary algorithm ((1 + 1) EA), and random search (RS) on the NK-landscape benchmark for all combinations of $k = 2, 3, 4, 5$ and $n = 10, 15, 20, 25$.

| $k$ (optimum) | SMBO | | EA | | (1 + 1) EA | | RS | |
|---|---|---|---|---|---|---|---|---|
| | Best | Average | Best | Average | Best | Average | Best | Average |
| | | | | $n = 10$ | | | | |
| 2 (0.704) | **0.704** | 0.702 | **0.704** | 0.686 | *0.698* | 0.675 | **0.704** | 0.649 |
| 3 (0.794) | **0.794** | 0.775 | **0.794** | 0.724 | *0.745* | 0.705 | **0.794** | 0.724 |
| 4 (0.787) | **0.787** | 0.755 | **0.787** | 0.725 | **0.787** | 0.714 | **0.787** | 0.727 |
| 5 (0.810) | **0.810** | 0.762 | *0.727* | 0.706 | **0.810** | 0.729 | **0.810** | 0.718 |
| | | | | $n = 15$ | | | | |
| 2 (0.743) | **0.743** | 0.742 | *0.714* | 0.693 | 0.681 | 0.628 | *0.714* | 0.674 |
| 3 (0.738) | **0.738** | 0.718 | 0.706 | 0.678 | 0.706 | 0.622 | *0.717* | 0.677 |
| 4 (0.747) | **0.747** | 0.721 | *0.711* | 0.685 | 0.705 | 0.646 | 0.710 | 0.680 |
| 5 (0.760) | **0.758** | 0.737 | 0.749 | 0.711 | 0.728 | 0.672 | *0.757* | 0.700 |
| | | | | $n = 20$ | | | | |
| 2 (0.729) | **0.729** | 0.726 | *0.718* | 0.689 | 0.668 | 0.613 | 0.711 | 0.673 |
| 3 (0.777) | **0.777** | 0.767 | *0.761* | 0.718 | 0.639 | 0.606 | **0.777** | 0.706 |
| 4 (0.775) | **0.775** | 0.747 | *0.731* | 0.708 | 0.676 | 0.640 | 0.707 | 0.684 |
| 5 (0.766) | **0.761** | 0.744 | *0.745* | 0.710 | 0.709 | 0.637 | 0.721 | 0.684 |
| | | | | $n = 25$ | | | | |
| 2 (0.753) | **0.753** | 0.747 | *0.727* | 0.698 | 0.679 | 0.590 | 0.701 | 0.673 |
| 3 (0.798) | **0.798** | 0.781 | 0.742 | 0.727 | 0.666 | 0.607 | *0.749* | 0.698 |
| 4 (0.775) | **0.762** | 0.743 | *0.750* | 0.714 | 0.639 | 0.595 | 0.695 | 0.679 |
| 5 (0.774) | **0.756** | 0.736 | *0.751* | 0.713 | 0.705 | 0.622 | 0.722 | 0.676 |

*The best (maximum) and average values of the best solution found by each algorithm are given for 50 runs.
†Bold numbers are the highest maxima and italic numbers are the second highest maxima.

future be "learnt" from the sampled data-point. The value of the bias term $w_0$ is set to the average $b_i s$ of the known data-points, and coefficients $w_i$ of the RBFs are determined by least-squares minimisation, as in the previous section.

The other settings of the SMBO are as follows. For all problems, the allowance $n$ of expensive function evaluations is set to 100. Initially, 10 data-points are sampled, and the number of sample points suggested by the SM is $n - 10 = 90$. To search the SM, we used a memetic algorithm on permutations with truncation selection, cycle crossover, swap mutation at a mutation rate of 0.01, and local search based on the 2-opt neighborhood. The population size and the number of generations were both set to 20, and 10 new offspring were generated in each generation, allowing an adequate solution to be obtained from the SM. The solution with the best predicted objective value is evaluated with the expensive objective function, provided it has not been sampled before. If it has, the second-best is sampled provided that it has not been sampled before. Otherwise the third best is sampled and so on.

We compared the SMBO algorithms with random search (RS) and a standard genetic algorithm (GA), both using the expensive objective function directly. We allowed all the algorithms the same number of evaluations of expensive objective function.

The GA had a population of 10 individuals ran for 18 generations, with 5 new individuals in each generation. It uses truncation selection, swap mutation with a probability of 0.01, and cycle crossover. We did 50 runs for each algorithm and problem.

The results, presented in Table 2, consistently rank $\text{SMBO}_H$ as the most effective algorithm, followed by the GA and $\text{SMBO}_S$, and then random search (RS). Clearly, the SM based on hamming distance is effective and better than swap distance by a surprising margin, considering that hamming and swap distance are closely related. $\text{SMBO}_H$ even outperforms $\text{SMBO}_S$ on the unimodal landscape under Swap distance (unis30), which we expected to favor $\text{SMBO}_S$.

We performed further analyses to try to understand the mechanism of the SMBO algorithms more fully. Firstly, to make sure that the distance metrics used by the SMBOs are suitable, we did a static analysis of the predictive power of the SMs in isolation from the SMBOs. This analysis is presented in Table 3. Looking at the number of significantly positive correlations and the average correlation, presented in Table 3, it is evident that hamming metric always gives better predictions than swap. Neither metric is deceptive, as there are no negative correlations. Fitness-distance correlations [34] of both distance metrics, given in Table 4, show that swap distance has higher correlations for the QAP, even though this is not reflected in its performance. This suggests that static prediction power gives a better indication whether a distance metric is appropriate for an SMBO.

Other aspects of the SM may affect the performance of an SMBO. For instance, we would obviously prefer to locate the real optimum of the SM before sampling with the expensive

Table 2: Results for random search (RS), a genetic algorithm (GA), $SMBO_H$, and $SMBO_S$ on QAP instances (kra30a, kra32, lipa30a, nug30, ste36a, and tho30) and unimodal instances (unih30 and unis30) of permutation problems.

| Instance | RS | | | GA | | | $SMBO_H$ | | | $SMBO_S$ | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Best | Average | SD | Best | Average | SD | Best | Average | SD | Best | Average | SD |
| kra30a | 118730 | 122777.00 | 2034.86 | 115840 | 123445.80 | 2642.21 | **110850** | **119649.60** | 3389.70 | 117270 | 122027.00 | 2205.65 |
| kra32 | 24156 | 25008.04 | 434.75 | 23440 | 24625.03 | 586.60 | **22590** | **24094.32** | 616.71 | 23848 | 24833.92 | 452.22 |
| lipa30a | 13664 | 13710.08 | 16.38 | 13646 | 13704.42 | 22.53 | **13633** | 13700.52 | 21.89 | 13638 | **13696.32** | 22.03 |
| nug30 | 7350 | 7618.00 | 84.60 | 7296 | 7558.36 | 126.65 | **7276** | **7500.24** | 97.36 | 7328 | 7563.88 | 94.38 |
| ste36a | 16736 | 18335.56 | 602.04 | 16516 | 18287.96 | 841.59 | **15364** | **17311.00** | 922.20 | 15654 | 17840.68 | 889.74 |
| tho30 | 190256 | 196662.82 | 3211.20 | **180274** | 194389.91 | 4414.22 | 180860 | **193415.44** | 4629.03 | 186172 | 195231.92 | 3534.21 |
| unih30 | 24 | 25.80 | 0.78 | 22 | 25.04 | 1.26 | **17** | **20.84** | 1.55 | 21 | 25.18 | 0.95 |
| unis30 | 19 | 21.66 | 0.87 | 17 | 20.91 | 1.29 | **15** | **18.48** | 1.83 | 19 | 21.04 | 0.94 |

*The best, average, and standard deviation of the best fitness found by each algorithm are reported for 50 runs.

Table 3: Correlation between predicted and real fitness on a test set of randomly sampled solutions after the SMs have been trained on 50 randomly sampled data-points. The columns contain counts of significantly positive (larger than 0.15) and significantly negative correlations (less than −0.15), together with average correlation coefficients.

| Instance | Hamming model | | Swap model | |
|---|---|---|---|---|
| | Pos. (Ave.) | Neg. | Pos. (Ave) | Neg. |
| kra30a | 50 (0.26) | 0 | 31 (0.20) | 0 |
| kra32 | 46 (0.22) | 0 | 34 (0.20) | 0 |
| lipa30a | 0 (N/A) | 0 | 0 (N/A) | 0 |
| nug30 | 32 (0.18) | 0 | 21 (0.19) | 0 |
| ste36a | 50 (0.23) | 0 | 27 (0.18) | 0 |
| tho30 | 12 (0.18) | 0 | 7 (0.16) | 0 |
| unih30 | 50 (0.77) | 0 | 50 (0.43) | 0 |
| unis30 | 47 (0.24) | 0 | 39 (0.19) | 0 |

*Each test was repeated 50 times.

Table 4: Fitness-distance correlation for permutation problems using hamming and swap distance.

| Instance | Hamming | Swap |
|---|---|---|
| kra30a | −0.02 | 0.31 |
| kra32 | −0.11 | 0.11 |
| lipa30a | 0.32 | 0.41 |
| nug30 | 0.38 | 0.44 |
| ste36a | 0.07 | 0.12 |
| tho30 | 0.46 | 0.52 |
| unih30 | 1.00 | 1.00 |
| unis30 | 1.00 | 1.00 |

Table 5: Fitness-distance correlation for SMs based on hamming and swap distance, after training them with 100 randomly sampled data-points.

| Instance | Hamming SM | Swap SM |
|---|---|---|
| kra30a | 0.85 | 0.57 |
| kra32 | 0.88 | 0.80 |
| lipa30a | 0.93 | 0.39 |
| nug30 | 0.88 | 0.71 |
| ste36a | 0.81 | 0.59 |
| tho30 | 0.82 | 0.83 |
| unih30 | 0.86 | 0.93 |
| unis30 | 0.87 | 0.79 |

Table 6: Number of solutions of permutation problems obtained by optimizing the SM by the SMBO algorithm (90 sequential optimisations) with predicted fitness which is better than or equal to the fitness of the best previous solution.

| Instance | $SMBO_H$ | | $SMBO_S$ | |
|---|---|---|---|---|
| | Better | Equal | Better | Equal |
| kra30a | 22.78 | 0 | 0.00 | 0 |
| kra32 | 22.28 | 0 | 0.00 | 0 |
| lipa30a | 33.80 | 0 | 0.04 | 0 |
| nug30 | 26.70 | 0 | 0.02 | 0 |
| ste36a | 29.66 | 0 | 0.04 | 0 |
| tho30 | 28.16 | 0 | 0.04 | 0 |
| unih30 | 4.28 | 0 | 0.28 | 0 |
| unis30 | 19.64 | 0 | 0.04 | 0 |

*The results are averaged over 50 runs.

objective function and the GA which we actually use to search the SM provides no guarantee of finding of the optimum or even a good solution. How good are the solutions that it finds? Table 5 shows fitness-distance correlations for the SMs, after training with 100 randomly sampled data-points. All these values are extremely high, suggesting that the GA usually locates very good solutions.

Another attribute of the SM that may affect the performance of an SMBO is the effect of the distance metric and the parameter $\beta$ on the topography of the model. These choices affect the extrapolative property of the model, which allows an optimum value to be found which is higher than that of any data-points. Table 6 shows that $SMBO_H$ can extrapolate much more often than $SMBO_S$. This may well provide any reason why $SMBO_H$ outperforms $SMBO_S$. However, the precise merit of hamming distance in this regard remains a subject for future work.

## 5. Experiments on Genetic Programming

Experiments were carried out on standard GP problems, symbolic regression and parity problems, and a unimodal problem, in which the fitness of a tree (to be minimised) is given by its distance to a given tree. This last problem can again be seen as a generalization of the OneMax problem for binary strings [33].

We have used structural hamming distance [35] as the metric of distance between two GP trees: this is a parameterless variant of the well-known structural distance for GP trees [36].

As in previous sections, we used a standard SMBO with an RBFN model fitted using the simplified learning procedure. The RBFs have the same widths $\beta = 1/2D^2$, where $D$ is the maximum distance across all centers. The value of the bias term $w_0$ is set to the average function value of the known data-points. The coefficients $w_i$ of the RBFs in the linear model are determined by least-squares minimisation.

We set other parameters as a function of the maximum depth md of the trees in the initial population, which is likely to determine the proportion of the search space that will actually be visited. The maximum number of nodes in a binary tree with a maximum depth md is $2^{\text{md}} - 1$. The number of expensive function evaluations allowed was $n = 2^{\text{md}}$. Thus our aim was to get each algorithm to produce the best solution in a time linearly proportional to the maximum size of the trees in the initial population. We set the initial sample size to 2 data-points and the number of points suggested by the SM to $n - 2$. To search the SM, we use a standard GP with tournament selection using a tournament size of 2, subtree crossover at a rate of 0.8, subtree mutation at a rate of 0.17, and reproduction operator at a rate of 0.03. The population size and the number of generations were both set to $n$, which we expected to provide GP with enough trials to locate a good solution of the SM. If the predicted value of the best solution found by the SM is better than the best value at any of the known data-points, then the model could extrapolate from the data, and that solution is evaluated using the expensive objective function. Otherwise, a point is chosen at random and evaluated with the expensive objective function in an attempt to gather more data about undersampled regions.

We compare the SMBO algorithm with random search (RS) and a standard GP, both using the expensive objective function directly. We allowed all the algorithms $n$ evaluations of the expensive objective function. The GP used has a population of approximately $\sqrt{n}$ individuals and it runs for approximately $\sqrt{n}$ generations. For fairness, the exact values of these two parameters are assigned in a way that their product is exactly $n$. It uses tournament selection with a tournament of size 2, subtree mutation with a probability of 0.17, subtree crossover at a rate of 0.8, and reproduction operator at a rate of 0.03. For each problem, we varied the maximum depth md between 3 and 7 and did 50 runs.

The results given in Table 7 make it immediately apparent that all algorithms get better results as md is increased, as we would expect. On the unimodal problem, looking at the average results, SMBO is consistently the best, followed by RS and finally by GP. The unimodal problem has the best fitness distance correlation with structural Hamming distance, suggesting that this metric is well suited for applying SMBO to this problem. This suggests that a good distance metric for SMBO in general should have good fitness-distance correlation for the problem at hand.

Surprisingly, RS does better than GP, which appears not to have had enough fitness evaluations available to get the evolution process properly started, especially when the sample and problem sizes were large. On the parity problem, SMBO wins again but with a smaller margin. Again, GP is worse than RS; however, if it is allowed a larger budget of expensive evaluations (i.e., md = 7), its performance matches RS. But more evaluations improve the performance of SMBO even more. On the symbolic regression problem, RS performs the best and GP the worst, although more evaluations allow SMBO and GP to outperform RS. This suggests that structural hamming distance is not particularly suitable for applying the SMBO to this problem.

There are many possible distances for parse trees we could use as basis for the SMBO. In future work, we should select distances suitable for the problem at hand, that is, that give rise to smoother/more unimodal landscape. In recent, Moraglio et al. [37] introduced a distance for GP, the semantic distance, that turns any GP problems into a unimodal problem. So for future work it could be interesting to use this distance as a base for SMBO.

## 6. Conclusions and Future Work

New applications are opened up by extending surrogate model-based optimisation (SMBO) to more complicated representations which cannot be naturally mapped to vectors of features. We have put forward a conceptually simple, formal, general, and systematic approach to adapting SMBO using radial basis function (RBF) networks to *any* target representation. Any algorithm that can be written in terms of Euclidean distances between candidate solutions can be generalised by replacing Euclidean distance function with a generic metric appropriate to the target representation (e.g., edit distance). RBF networks can be naturally generalised to encompass any representations because both the approximating model and the learning of the model parameter can be cast completely in a representation-independent way and rely only on distance relations between training instances and query instances.

We have validated experimentally the framework on three representations. First, we have considered the binary strings representation endowed with the hamming distance and tested the SMBO on the NK-landscapes, obtaining consistently that with the same budget of expensive function evaluations, the SMBO performs the best in comparison with other search algorithms. The second representation we have considered is the permutation representation endowed with hamming distance and with swap distance and tested the SMBO on the quadratic assignment problem and on unimodal problems, obtaining consistently that with the same budget of expensive function evaluations, the SMBO with hamming distance performs the best in comparison with other search algorithms. Surprisingly, the SMBO based on swap distance does not work as well as the SMBO based

Table 7: Results for unimodal, parity, and symbolic regression problems obtained by SMBO, random search (RS), and genetic programming (GP).

(a) ⟨Unimodal⟩

| md | SMBO | | | RS | | | GP | | |
|---|---|---|---|---|---|---|---|---|---|
| | Best | Average | SD | Best | Average | SD | Best | Average | SD |
| 3 | **0.11** | 0.47 | 0.19 | **0.11** | **0.43** | 0.16 | 0.22 | 0.50 | 0.18 |
| 4 | **0.07** | **0.14** | 0.05 | 0.13 | 0.37 | 0.13 | 0.11 | 0.49 | 0.25 |
| 5 | **0.04** | **0.07** | 0.03 | 0.08 | 0.24 | 0.08 | 0.05 | 0.48 | 0.21 |
| 6 | **0.01** | **0.04** | 0.04 | 0.08 | 0.14 | 0.08 | 0.14 | 0.46 | 0.21 |
| 7 | **0.01** | **0.02** | 0.04 | 0.04 | 0.18 | 0.04 | 0.06 | 0.32 | 0.20 |

(b) ⟨4-Odd Parity⟩

| md | SMBO | | | RS | | | GP | | |
|---|---|---|---|---|---|---|---|---|---|
| | Best | Average | SD | Best | Average | SD | Best | Average | SD |
| 3 | 37.50 | **45.00** | 6.45 | 37.50 | **45.00** | 6.45 | 37.50 | 48.75 | 3.95 |
| 4 | 37.50 | **40.00** | 5.27 | 37.50 | 41.25 | 6.04 | 37.50 | 42.50 | 6.45 |
| 5 | 37.50 | **37.50** | 0.00 | 37.50 | **37.50** | 0.00 | 37.50 | 47.50 | 5.27 |
| 6 | 37.50 | **37.50** | 0.00 | 37.50 | **37.50** | 0.00 | 37.50 | 41.25 | 6.04 |
| 7 | **25.00** | **33.75** | 6.04 | 37.50 | 37.50 | 0.00 | 37.50 | 37.50 | 0.00 |

(c) ⟨Symbolic Regression⟩

| md | SMBO | | | RS | | | GP | | |
|---|---|---|---|---|---|---|---|---|---|
| | Best | Average | SD | Best | Average | SD | Best | Average | SD |
| 3 | 3.44 | **4.88** | 0.82 | 3.44 | **4.88** | 0.78 | **2.64** | 5.17 | 1.35 |
| 4 | 4.46 | 6.35 | 1.17 | **4.27** | **5.78** | 1.41 | 4.46 | 6.39 | 1.58 |
| 5 | 3.84 | 5.58 | 1.21 | **3.51** | **5.18** | 1.21 | 4.05 | 5.39 | 1.27 |
| 6 | **2.95** | 3.74 | 0.73 | 2.99 | **3.52** | 3.48 | 3.48 | 4.39 | 0.57 |
| 7 | **3.45** | **4.50** | 0.77 | 3.81 | 4.96 | 0.54 | 3.71 | 4.62 | 0.67 |

*The best (minimum) and average fitness values for the best solution found by each algorithm, for md = 3, 4, 5, 6, 7, over 50 runs.

on hamming distance. We have presented an analysis in the attempt to elucidate the causes of the different performance. Further investigation is required to pinpoint the structural difference between Hamming distance and Swap distance that gives rise to the performance difference. Lastly, as an experimental validation of the framework on a nontrivial discrete space and structured representation, we have considered the genetic programming (GP) trees endowed with the structural hamming distance and tested the SMBO on a test-bed of standard GP problems, obtaining that with the same budget of expensive function evaluations, the SMBO performs well in a comparison with other search algorithms. These results suggest that our approach has the potential to solve real-world combinatorial optimisation problems with complicated solution representations and nontrivial discrete search spaces.

Much work remains to be done. Firstly, we plan to look at further well-known permutation and GP problems and consider different distance metrics. For instance, the traveling salesman problem may be cast in terms of a distance based on the 2-opt move. Then we intend to consider problems with other complicated nonvectorial representations, such as variable-length sequences. Our eventual aim is to address some challenging real-world problems in a new way. We will also experiment with different types of RBF and more complex learning processes (i.e., learning the centers and the widths of the RBFs). Lastly, we will attempt the generalization of more sophisticated interpolation and regression methods, including Gaussian process regression, which is a state-of-the-art method in machine learning.

## Disclosure

A preliminary version of this paper appeared in *Proceedings of the Eleventh European Conference on Evolutionary Computation in Combinatorial Optimization*, pp. 142–154, 2011, and in *Proceedings of the Genetic and Evolutionary Computation Conference (Companion Material)*, pp. 133-134, ACM SIGEVO, 2011.

## Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

## References

[1] P. M. Pardalos and M. G. C. Resende, *Handbook of Applied Optimization*, Oxford University Press, 2002.

[2] P. M. Pardalos and E. Romeijn, Eds., *Handbook of Global Optimization. Volume 2*, vol. 62, Kluwer Academic Publishers, 2002.

[3] J.-H. Seo, Y.-H. Kim, H.-B. Ryou, S.-H. Cha, and M. Jo, "Optimal sensor deployment for wireless surveillance sensor networks by a hybrid steady-state genetic algorithm," *IEICE Transactions on Communications*, vol. E91-B, no. 11, pp. 3534–3543, 2008.

[4] S. S. Tong and B. A. Gregory, "Turbine preliminary design using artificial intelligence and numerical optimization techniques," *Journal of Turbomachinery*, vol. 114, no. 1, pp. 1–10, 1992.

[5] H.-M. Gutmann, "A radial basis function method for global optimization," *Journal of Global Optimization*, vol. 19, no. 3, pp. 201–227, 2001.

[6] D. R. Jones, M. Schonlau, and W. J. Welch, "Efficient global optimization of expensive black-box functions," *Journal of Global Optimization*, vol. 13, no. 4, pp. 455–492, 1998.

[7] R. G. Regis and C. A. Shoemaker, "Constrained global optimization of expensive black box functions using radial basis functions," *Journal of Global Optimization*, vol. 31, no. 1, pp. 153–171, 2005.

[8] D. R. Jones, "A taxonomy of global optimization methods based on response surfaces," *Journal of Global Optimization*, vol. 21, no. 4, pp. 345–383, 2001.

[9] N. A. C. Cressie, *Statistics for Spatial Data*, John Wiley & Sons, New York, NY, USA, 1993.

[10] T. Mitchell, *Machine Learning*, McGraw Hill, 1997.

[11] J.-H. Seo, Y.-H. Lee, and Y.-H. Kim, "Feature selection for very short-term heavy rainfall prediction using evolutionary computation," *Advances in Meteorology*, vol. 2014, Article ID 203545, 15 pages, 2014.

[12] G.-M. Yoon, J. Kim, Y.-H. Kim, and B.-R. Moon, "Performance improvement by genetic feature selection and adjusting ratings'mid-point value in the neural network-based recommendation models," *Advances in Information Sciences and Service Sciences*, vol. 4, no. 11, pp. 37–43, 2012.

[13] A. Moraglio, R. Poli, and R. Seehuus, "Geometric crossover for biological sequences," in *Proceedings of the European Conference on Genetic Programming*, pp. 121–132, 2006.

[14] R. Poli, W. B. Langdon, and N. F. McPhee, *A Field Guide to Genetic Programming*, Lulu Enterprises, 2008.

[15] K. Seo, S. Hyun, and E. D. Goodman, "Genetic programming-based automatic gait generation in joint space for a quadruped robot," *Advanced Robotics*, vol. 24, no. 15, pp. 2199–2214, 2010.

[16] Y. Jin, "A comprehensive survey of fitness approximation in evolutionary computation," *Soft Computing Journal*, vol. 9, no. 1, pp. 3–12, 2005.

[17] L. Bajer and M. Holena, "Surrogate model for continuous and discrete genetic optimization based on RBF networks," in *Proceedings of the International Conference on Intelligent Data Engineering and Automated Learning*, pp. 251–258, 2010.

[18] I. Voutchkov, A. J. Keane, A. Bhaskar, and T. M. Olsen, "Weld sequence optimization: the use of surrogate models for solving sequential combinatorial problems," *Computer Methods in Applied Mechanics and Engineering*, vol. 194, no. 30–33, pp. 3535–3551, 2005.

[19] T. L. Lew, A. B. Spencer, F. Scarpa, K. Worden, A. Rutherford, and F. Hemez, "Identification of response surface models using genetic programming," *Mechanical Systems and Signal Processing*, vol. 20, no. 8, pp. 1819–1831, 2006.

[20] A. Moraglio and A. Kattan, "Geometric generalisation of surrogate model based optimisation to combinatorial spaces," in *Proceedings of the 11th European Conference on Evolutionary Computation in Combinatorial Optimization*, pp. 142–154, 2011.

[21] A. Moraglio, Y.-H. Kim, and Y. Yoon, "Geometric surrogate-based optimisation for permutation-based problems," in *Proceedings of the 13th Annual Genetic and Evolutionary Computation Conference (GECCO '11)*, pp. 133–134, July 2011.

[22] C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer, New York, NY, USA, 2007.

[23] L. C. Jain, *Radial Basis Function Networks*, Springer, 2001.

[24] A. Moraglio, *Towards a Geometric Unification of Evolutionary Algorithms [Ph.D. thesis]*, University of Essex, 2007.

[25] A. Moraglio, Y.-H. Kim, Y. Yoon, and B.-R. Moon, "Geometric crossovers for multiway graph partitioning," *Evolutionary Computation*, vol. 15, no. 4, pp. 445–474, 2007.

[26] Y. Yoon and Y.-H. Kim, "Geometricity of genetic operators for real-coded representation," *Applied Mathematics and Computation*, vol. 219, no. 23, pp. 10915–10927, 2013.

[27] Y. Yoon, Y.-H. Kim, A. Moraglio, and B.-R. Moon, "Quotient geometric crossovers and redundant encodings," *Theoretical Computer Science*, vol. 425, pp. 4–16, 2012.

[28] S. Kauffman, *Origins of Order: Self-Organization and Selection in Evolution*, Oxford University Press, 1993.

[29] Y.-H. Kim and Y. Yoon, "A new Kernighan-Lin-type local search for the quadratic assignment problem," in *Proceedings of the International Conference on Scientific Computing*, pp. 185–189, 2009.

[30] C. E. Rasmussen, *Gaussian Processes for Machine Learning*, MIT Press, 2006.

[31] Y.-H. Kim and B.-R. Moon, "New topologies for genetic search space," in *Genetic and Evolutionary Computation Conference (GECCO '05)*, pp. 1393–1399, June 2005.

[32] Y. Yoon and Y.-H. Kim, "A Mathematical Design of Genetic Operators on $GL_n(Z_2)$," *Mathematical Problems in Engineering*, vol. 2014, Article ID 540936, 8 pages, 2014.

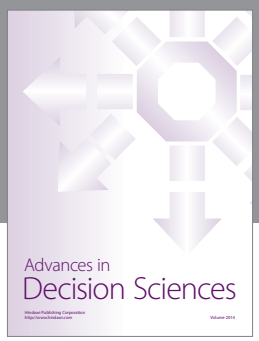
[33] J. D. Schaffer and L. J. Eshelman, "On crossover as an evolutionary viable strategy," in *Proceedings of the 4th International Conference on Genetic Algorithms*, pp. 61–68, 1991.

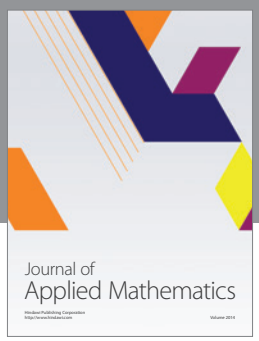
[34] T. Jones and S. Forrest, "Fitness distance correlation as a measure of problem difficulty for genetic algorithms," in *Proceedings of the 6th International Conference on Genetic Algorithms*, pp. 184–192, 1995.

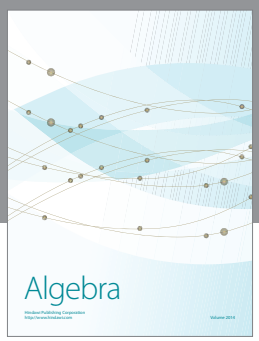
[35] A. Moraglio and R. Poli, "Geometric landscape of homologous crossover for syntactic trees," in *Proceedings of the IEEE Congress on Evolutionary Computation (IEEE CEC '05)*, pp. 427–434, September 2005.

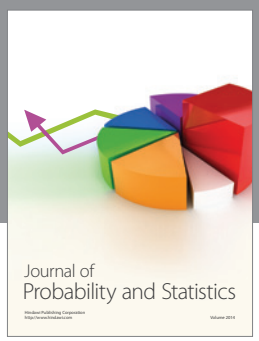
[36] A. Ekart and S. Z. Nemeth, "A metric for genetic programs and fitness sharing," in *Proceedings of the European Conference on Genetic Programming*, pp. 259–270, 2000.

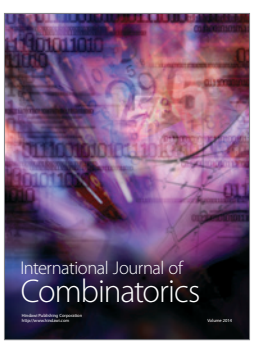
[37] A. Moraglio, K. Krawiec, and C. G. Johnson, "Geometric semantic genetic programming," in *Proceedings of the 12th International Conference on Parallel Problem Solving from Nature*, pp. 21–31, 2012.

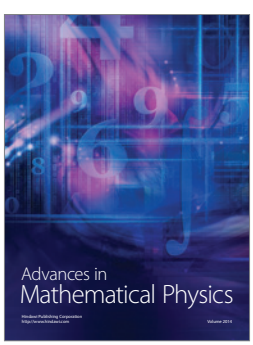


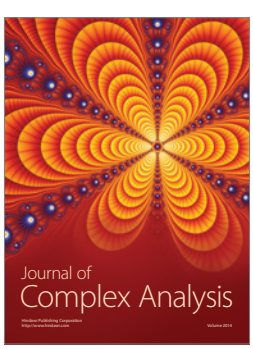


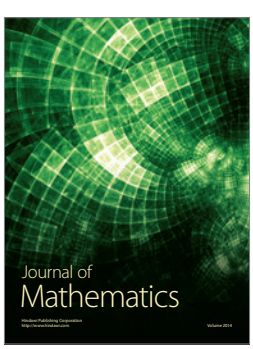


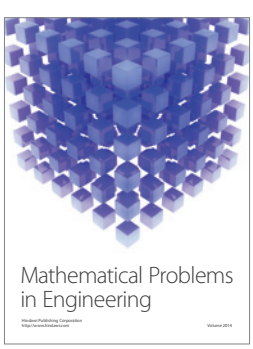


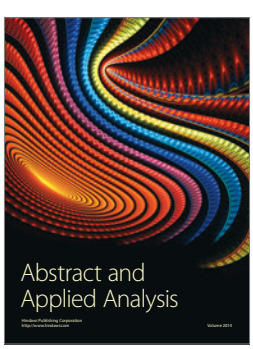


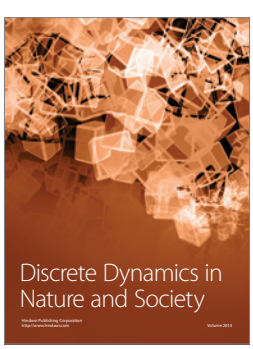


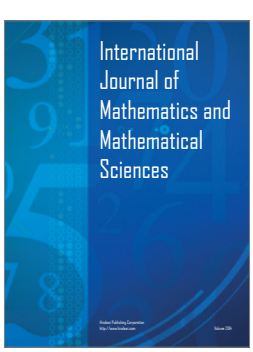












Hindawi

Submit your manuscripts at
http://www.hindawi.com