Geometric Generalisation of Surrogate Model Based Optimisation to Combinatorial Spaces

Alberto Moraglio and Ahmed Kattan

School of Computing and Centre for Reasoning, University of Kent, Canterbury, UK
College of Computer and Information Systems, Um Al-Qura University, Saudi Arabia
a.moraglio@kent.ac.uk, akattan@uqu.edu.sa

Abstract. In continuous optimisation, Surrogate Models (SMs) are often indispensable components of optimisation algorithms aimed at tackling real-world problems whose candidate solutions are very expensive to evaluate. Because of the inherent spatial intuition behind these models, they are naturally suited to continuous problems but they do not seem applicable to combinatorial problems except for the special case when solutions are naturally encoded as integer vectors. In this paper, we show that SMs can be naturally generalised to encompass combinatorial spaces based in principle on *any arbitrarily complex underlying solution representation* by generalising their geometric interpretation from continuous to general metric spaces. As an initial illustrative example, we show how Radial Basis Function Networks (RBFNs) can be used successfully as surrogate models to optimise combinatorial problems defined on the Hamming space associated with binary strings.

1 Introduction

Some typologies of tasks when cast as optimisation problems give rise to objective functions which are prohibitively expensive to evaluate. Furthermore, oftentimes these problems are black-box problems, i.e., whose problem class is unknown, and they are possibly mathematically ill-behaved (e.g., discontinuous, non-linear, non-convex). For example, most engineering design problems are of this type (see e.g., [12]). They require experiments and/or simulations to evaluate to what extent the design objective has been met as a function of parameters controlling the design. In evolutionary computation parlance, the controlling parameters are the genotype that encodes the design solution (i.e., the phenotype) which needs to be expressed via an expensive simulation (i.e., the growth function) to be evaluated (i.e., fitness evaluation). The simulation can take many minutes, hours, or even days to complete.

Optimisation methods based on surrogate models, also known as response surface models, have been successfully employed to tackle expensive objective functions (EOFPs). For a survey on surrogate model based optimisation methods refer to [6]. A surrogate model is a mathematical model that approximates as precisely as possible the expensive objective function of the problem at hand, and that is computationally much cheaper to evaluate. The objective function is considered unknown. The surrogate model is built solely from available known

values of the expensive objective function evaluated on a set of solutions. We refer to the pair (solution, known objective function value) as data-point. When the search space is the real line (i.e., solutions are real numbers), perhaps the simplest example of surrogate model is linear interpolation which determines a function from the graph of the data-points available by linking each data-point to the closest with a straight line. A class of more natural-looking surrogate models for the real line is the class of polynomial curves of suitable degree, which can be used to interpolate all data-points and that are everywhere differentiable. These and other methods to build surrogate models on the real line naturally extend to higher dimensional spaces giving rise to various forms of spatial interpolation and spatial regression.

The traditional procedure of surrogate model based optimisation (SMBO) [6] is outlined in Algorithm 1. An initial surrogate model is constructed using the objective values of a small set of solutions evaluated using the expensive objective function. The remaining expensive objective function evaluations out of a limited budget are applied to candidate solutions which the surrogate model predicts to have promising performance. The process interleaves search of the surrogate model to obtain its optimum, evaluation of the optimum solution of the model using the expensive objective function, and update of the surrogate model with the new data-point. Note that the role of the evolutionary algorithm in the SMBO procedure is to infer the location of a promising solution of the problem using the surrogate model, and it is not directly applied to the original problem with the expensive objective function. This is feasible because the computational cost of a complete run of the evolutionary algorithm on the surrogate model 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).

Algorithm 1 Surrogate Model Based Optimisation

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** limit number of expensive function evaluations not reached **do**
3: Construct a new surrogate model using all data-points available
4: Determine the optimum of the surrogate model by search, e.g., using an evolutionary algorithm (this is feasible as the model is cheap to evaluate)
5: Evaluate the solution which optimises the surrogate model in the problem with the expensive objective function (additional data-point available)
6: **end while**
7: Return the best solution found (the best in the set of data-points)

Virtually all surrogate models are implicitly or explicitly spatial models as their predictions involve exploiting some assumed spatial relations (e.g., smoothness) between the values of the objective function at a query point whose value is unknown and has to be predicted, and the known data-points. This makes SMBOs naturally suited to continuous optimisation. However they do not seem

to be applicable to combinatorial optimisation problems except in those cases in which solutions are naturally represented as vectors of integers, in which case adequately discretized versions of the surrogate model may be used. Furthermore, when solutions are vectors, integer or real, a host of techniques to build functions from data-points can be borrowed from statistics (i.e., multi-variate regression [3]) and machine learning (i.e., supervised learning by e.g., neural networks and support vector machines [9]), which can be used to build surrogate models.

There is an increasing number of optimisation problems naturally associated with complex solution representations which have also very expensive objective functions. For example, permutations and related representations are natural representations for solutions of many types of scheduling problems. In real-world problems, candidate scheduling solutions may need to be tested in a complex setting by running a computationally expensive simulation of, for example, an entire production process, to be evaluated. Variable-length sequences are natural representations for biological sequences in bio-informatics problems. In this context, surrogate model based optimisation may be used to determine which particular biological sequences to study in detail by scientist or to simulate at an atomic level that, for example, are more likely to correspond to proteins with desired target properties/functions. Genetic Programming that normally uses a tree representation, has a number of application domains with expensive objective functions. For example, one of them is when genetic programs encode behavioral controllers of robots that may need to be tested in a virtual or real environment a number of times to assess how good the controller is at controlling the robot for certain target tasks (i.e., wall-following or obstacle avoidance).

The current situation of surrogate model with regard to solution representations is as follows. There is much existing literature on surrogate model based optimisation using evolutionary algorithms or other search algorithms to optimise the surrogate model for continuous spaces. See for example the survey [5]. Some recent work covers the case when the underlying solution representation is discrete vectors (e.g., [1]). There are works focusing on specific real-world applications with expensive objective functions which are inherently combinatorial problems with structured solutions (e.g., graphs) which are approached by encoding solutions in vectorial form to use standard surrogate models (e.g., [13][2]). There are also approaches in which evolutionary algorithms are not used to search the surrogate model but to train the surrogate model on the known data-points, see e.g. [8], in which Genetic Programming is used to do symbolic regression to determine the best fitting vector-input function to the data-points.

To the authors's best knowledge there are no works in literature on surrogate models defined *directly* on more complex representations than vectors. Therefore, for search problems naturally based on structured representations, surrogate models can be used only after shoe-horning the original representation to a vector form. This introduces extra non-linearity in the target expensive objective function, so making it harder to learn, and consequently requiring more expensive samples to approximate it well enough to locate its optimum. The aim of this paper is to introduce a framework that allows us to define systematically surrogate models directly on any underlying solution representation, allowing us to choose the most natural representation for the problem at hand.

A geometric framework of recent introduction [10] has been successfully used to generalise various search algorithms from continuous spaces to combinatorial spaces in a principled way. This paper shows that the geometric methodology extends naturally to the generalisation of machine learning algorithms. The method is conceptually simple. Firstly, (i) the original algorithm on the continuous space is rewritten only as a function of the Euclidean distance between points. Algorithms that are inherently spatial can be often rewritten in these terms. Then (ii) the generalisation is done by replacing the Euclidean distance with a generic distance (i.e., formally a metric), obtaining a general formally well-defined algorithm. Finally, (iii) the formal algorithm can be specified to any solution representation by specifying it to an edit distance directly defined on the target representation.

The generalised algorithms using the geometric methodology can be naturally specified to complex representations because many types of structured objects admit natural notion of distance/similarity between them. In particular *edit distances* are well suited to structured objects. The edit distance between two configurations is the minimum number of edit operations required to transform one of them into the other. Edit operations are unitary modifications that change one configuration into a similar configuration. For example, the Hamming distance is an edit distance between binary strings based of the bit-flip edit move. On permutations, the swap distance is the minimum number of swaps of two elements to sort one permutation into the other. On variable length sequences, the Levenshtein distance is the minimum number of insertions, deletions, or changes of characters in a sequence to transform it into the other. There are also various types of edit distances defined on trees and graphs based on moves editing edges and nodes.

In the reminder of the paper, we show how a supervised machine learning algorithm to learn functions from data, namely Radial Basis Function Networks (RBFNs) (see e.g., [4]), can be successfully generalised to encompass any solution representation using the geometric methodology. The generalised algorithm obtained is then formally instantiated to the binary strings representation endowed with the Hamming distance. As a preliminary experimental analysis, we use the resulting learning model within a SMBO to optimise a well-known testbed of problems on binary strings, the NK-landscapes [7], which we will consider having costly objective function [1]. The generalised RBFNs model derived in this paper is very general and can be used in principle with any solution representation. Furthermore, other learning algorithms naturally defined in spatial terms, e.g., spatial regression algorithms (i.e., Gaussian Process Regression [11]) can be generalised analogously.

[1] The aim of the present paper is *not* to show that the generalised SMBO can be competitive on real-world problems with expensive objective functions with structured representations. It is to show that the generalised SMBO can be in principle applied to such cases and that it provides meaningful results when applied to well-studied toy problems on a simple *discrete* space. This preliminary step is necessary because the transition from continuous to discrete spaces is a large conceptual leap, and there is no guarantee that such an approach would work on even simple discrete spaces.

2 Radial Basis Function Networks

In the machine learning literature, there are a number of approaches to "learning" a function belonging to a certain class of functions from data-points (i.e., finding a function in that class that interpolates and best fits the data-points according to some criteria), which can be naturally cast only in terms of (Euclidean) distances between points, hence readily generalised to metric spaces, by replacing the Euclidean distance with a general metric. These include Nearest-Neighbors Regression, Inverse Distance Weighting Interpolation, Radial Basis Function Network Interpolation, and Gaussian Process Regression (also known as Kriging). The first two methods are simpler but they are not adequate to be used as surrogate models because the global optimum of the learnt functions from the data-points coincide with a data-point used in the construction of the function. Consequently, these methods cannot be used to suggest a solution that improves over the known data-points (i.e., they cannot extrapolate from the data-points). Gaussian Process Regression is a very powerful method with a solid theoretical foundation, which not only can make a rational extrapolation about the location of the global optimum, but also gives an interval of confidence about the prediction made. Radial Basis Function Network Interpolation is conceptually simpler than Gaussian Process Regression and can extrapolate the global optimum from the known data-points. In this paper, we focus on RBFNs, and leave the generalisation of Gaussian Process Regression as future work.

2.1 Classic RBFNs

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

$$\phi(x) = \exp(-\beta\|\mathbf{x} - \mathbf{c}\|^2)$$

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

Radial basis functions are typically used to build function approximations of the form:

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

Therefore the approximating function $y(x)$ is represented as a sum of N radial basis functions, each associated with a different center c_i, a different width β_i, and weighted by an appropriate coefficient w_i, plus a bias term w_0. Any continuous function can in principle be approximated with arbitrary accuracy by a sum of this form, if a sufficiently large number N of radial basis functions is used.

In a RBF network there are three types of parameters that need to be determined to optimise the fit between $y(x)$ and the data: the weights w_i, the centers $\mathbf{c}_i$, and the RBF width parameters β_i. The most common method to find these parameters has two phases. Firstly, using unsupervised learning (i.e., clustering) the position of the centers and the widths of the RBFs are determined. Then, an optimal choice of weights w_i that optimises the accuracy of the fit is done by least squares minimisation.

A widely applied simplified procedure to fit RBF networks to the data, which skips the unsupervised learning phase, consists of choosing the centers c_i to coincide with the known points x_i, and choosing the widths β_i according to some heuristic based on the distance to nearest neighbors of the center c_i (local model), or to fix all widths to the same value which is taken proportional to the maximum distance between the chosen centers (global model). The bias w_0 can be set to the mean of the function values b_i at the known data-points (i.e., function values of the points in the training set), or set to 0. Under these conditions, the weights w_i can be determined by solving the system of N simultaneous linear equations in w_i obtained by requiring that the unknown function interpolates exactly the known data-points:

$$y(\mathbf{x}_i) = b_i,\ i = 1 \ldots N.$$

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

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

2.2 Generalisation of RBFNs to Arbitrary Representations

To generalise RBFNs we need to generalise: (i) the class of functions used as approximants of the unknown function; (ii) the training procedure to determine the function within that class that best fits the data-points; (iii) the model query procedure that given a query point whose value is unknown it returns its predicted value.

Following the geometric methodology for the generalisation, we first need to rewrite any of the above three elements only as a function of the Euclidean distance, then substitute the Euclidean distance with a generic metric obtaining a formal generalisation of the original algorithm, and finally specify the formal algorithm to a distance rooted in the target representation to obtain the specific instance of the algorithm for the target representation. If all these points are possible, then the generalisation of the algorithm has been successful. In the following, we show that indeed the geometric methodology can be applied successfully to generalise RBFNs.

Let M be a metric space endowed with a distance function d. A radial basis function $\phi : \mathbf{R}^n \to \mathbf{R}$ whose value depends only on the distance from some point $c \in \mathbf{R}^n$ so that $\phi(\mathbf{x}) = \phi(\|\mathbf{x} - \mathbf{c}\|)$ can be naturally generalised to a function $\phi : \mathbf{M} \to \mathbf{R}$ whose value depends only on the distance from some point $c \in M$ *in the metric space* so that $\phi(\mathbf{x}) = \phi(d(\mathbf{x}, \mathbf{c}))$. For example, the generalised

Gaussian functions are obtained by replacing the Euclidean distance with the generic metric d in the original definition: $\phi(x) = \exp(-\beta d(\mathbf{x}, \mathbf{c})^2)$.

A set of configurations endowed with a notion of edit distance is a metric space, as all edit distances meet the metric axioms. Consequently, a generalised radial basis function is well-defined on any set of configurations, or in other words, is a *representation-independent function*. For example, the set of binary strings $\mathbf{H}$ endowed with the Hamming distance hd form a metric space. Therefore, the generalised Gaussian functions, when the Hamming distance hd is specified as metric d, become well-defined functions $\phi : \mathbf{H} \to \mathbf{R}$, which map binary strings to real (note that both c and x are binary strings). The same generalised Gaussian functions are well-defined functions mapping permutations to real when the swap distance on permutations is specified as metric d.

The approximating model $y(\mathbf{x})$ which is a linear combination of radial basis functions can be generalised by considering a linear combination of generalised radial basis functions: $y(\mathbf{x}) = w_0 + \sum_{i=1}^{N} w_i \, \phi(d(\mathbf{x}, \mathbf{c}_i))$. As its components, the generalised approximating model is also representation-independent and it can be specified to any underlying solution representation by specifying as underlying metric d a distance function rooted in the target representation. Interestingly, a generalised approximating model is a way of representing a very large family of functions on general metric spaces parameterised on the center locations c_i, the weights w_i, the widths β_i, and by the specific underlying metric d. When the underlying metric space is finite (as it is in combinatorial optimisation problems), any function can in principle be approximated with arbitrary accuracy by a sum of this form, if a sufficiently large number N of radial basis functions is used [2].

The method to fit the model to the known data-points does not refer explicitly to their underlying representation but it depends solely on the distances between the known data-points, taking $g_{ij} = \phi(d(\mathbf{x}_j, \mathbf{x}_i))$, and on the known objective values b_i. Therefore, in effect, model-fitting is also representation-independent. In particular, the simplified model-fitting procedure which fixes the centers and the widths and determine the weights w_i by least squares minimisation can be done by solving the system $\mathbf{Gw} = \mathbf{b}$, regardless of the underlying representation. Notice however that when the distance function d is not embeddable in the Euclidean space, the radial basis functions which are positive definite on the Euclidean space are not necessarily positive definite with regard to the distance function d. In turns, the matrix G is not necessarily a positive definite matrix, hence the existence of the inverse matrix G^{-1} needed to determine the weights w_i is not guaranteed. This difficulty can be overcome by considering the pseudo-inverse of the matrix G which always exist, it is unique, and it corresponds to the traditional inverse when it exists. It can be shown that the weights w_i determined by solving the system $\mathbf{Gw} = \mathbf{b}$ using the pseudo-inverse are exactly those obtained by least squares minimisation.

As for querying the model with a point to obtain the predicted value, clearly the query point is represented using the underlying representation (e.g., the query point is a binary string when the model is specified to the Hamming

[2] To see this, consider the extreme case in which every point in space is associated with a radial basis function. In this case, it is always possible to choose the weights of the bases to fit the function values at each location.

distance on binary strings). However, importantly, the way of calculating the predicted value is representation-independent as it does not depend on the underlying representation but only on the distance between the query point and the center points.

In summary, the definition, learning and querying of RBFNs naturally generalise from Euclidean spaces to general metric spaces. The generalised model applies to any underlying representation once a distance function rooted on that representation is provided. In particular, this method can be used *as it is* to learn in principle any function mapping complex structured representations to reals. It is important to note that it is the generalised RBFNs learning that adapts to the target representation, rather than the other way around. In particular, there is no special requirement of the target representation of being shoehorn in a vector of features. This allows us to choose the most natural representation and distance for the task at hand which, as discussed earlier, is likely to make the learning easier. A further point to note is that the adaptation of the general model to the specific representation is done by *formal instantiation* of a generic metric to a distance function associated with the target representation, in particular the model adapts naturally to the target representation without introducing any arbitrary ad-hoc element, for example, to deal with more complex representations. Finally, this way of looking at learning algorithms is both formal and general and naturally bridges continuous and combinatorial spaces. Naturally, the fact that this generalised model is well-defined on any representation is logically independent from that it may work well on all representations and for problems occurring in practice. Therefore, it is important to test the framework experimentally for different representations and types of problems. In the following section, we present initial experiments for when the framework is applied to binary strings[3] on a standard test-bed. In future work, we will investigate how the framework performs when specified to more complex representations and to problems with expensive evaluation functions arising in practice.

3 Experiments

Experiments have been carried out using the well-known NK-Landscape problem [7], which provides a tunable set of rugged, epistatic landscapes over a space of binary strings, which we will consider having costly objective function. In our experiments, in order to evaluate the performance of the SMBO algorithm under different conditions of problem size and ruggedness, we use landscapes of size $n = 10, 15, 20, 25$ for $k = 2, 3, 4, 5$, for a total of 16 combinations.

We use a standard surrogate model based optimisation algorithm (see Algorithm 1). The surrogate model used is a RBFN model which is fitted to the available data-points using the simplified learning procedure presented in the previous section. The centers c_i of the radial basis functions are chosen to coincide with all available data-points. The widths of the radial basis functions are

[3] Binary strings are of course a special type of vectors. However, they are a valid representation to use as an illustrative example of application of the generalised SMBO to combinatorial spaces because their property of being vectors is not utilised.

assigned to the same value $\beta = \frac{1}{2D^2}$ where D is the maximum distance between all centers. With this setting of β, each radial basis function is "spread on the space" to cover all other centers so that each known function value at a center can potentially contribute significantly to the prediction of the function value of any point in space, and not only locally to function values of points near the given center (i.e., we force the surrogate model to be a global approximating function). The value of the bias term w_0 is set to the average function value of the known data-points, i.e., the average of vector b_i. In this way, the predicted function value of a point which is out of reach of the influence of all centers is by default set to the average of their function values. The coefficients w_i of the radial basis functions in the linear model are determined by least squares minimisation as described in the previous section.

The other settings of surrogate model based optimisation are as follows. We set the parameters as a function of the problem size n. The number of total available expensive function evaluations is set to n^2. So, essentially our aim is to find the best solution to the problem the algorithm can produce in quadratic time out of an exponential number of candidate solutions (i.e., 2^n). This setting is just a term of reference, as for different problems one may have a different number of solutions available with regard to the problem size. We set the size of the initial sample of data-points to two, and the number of sample points suggested by the surrogate model to $n^2 - 2$. This setting is consistent with the working hypothesis that the surrogate model is better than random sampling at suggesting promising solutions which are better than the known data-points as it uses as much as possible the surrogate model to make predictions. To search the surrogate model we use a standard generational evolutionary algorithm with tournament selection with tournament size two, uniform crossover with crossover rate 0.5 and bitwise mutation with mutation rate $1/n$. The population size and the number of generations are both set to $10n$, which provide the evolutionary algorithm with an abundant lot of trials to locate the optimal or a near-optimal solution of the surrogate model. If the predicted objective value of the best solution of the surrogate model is better than the best known objective value of the known data-points, then the model could extrapolate from the data, and that solution is evaluated in the expensive objective function. Otherwise, the surrogate model has failed at suggesting a promising solution which improves over the known best, and a solution sampled uniformly at random is evaluated with the expensive objective function in the attempt to gather more data about under-sampled regions of the problem and improve the accuracy of the surrogate model to help subsequent searches on the model.

To have a reference for the performance of the SMBO algorithm, we compared it with Random Search (RS), a standard (1+1) Evolutionary Algorithm ((1+1) EA), and with a generational Evolutionary Algorithm (EA) applied directly on the problem with the expensive objective function. The reason we included random search in the set of algorithms is because, whereas it is safe to assume that in practice evolutionary algorithms are better than random search in normal circumstances, with small samples random search can do relatively well. We gave all algorithms in the comparison exactly the same number of expensive objective functions, which is n^2 trials, and report the best solution found. The (1+1) EA

has a population of a single individual and uses bitwise mutation with bit-flip probability of $1/n$. The EA has a population of n individuals, it runs for n generations, it uses tournament selection with size two, bitwise mutation with bit-flip probability of $1/n$, and uniform crossover with crossover rate 0.5. For each considered combination of the parameters n and k, we generated a single fitness landscape and did 10 independent runs of the above algorithms on it. For each problem, we also estimated the global optimum by running an evolutionary algorithm with a very large population (1000 individuals) and very large number of generations (1000 generations).

Table 1. Results on the NK landscape benchmark. Mean and max over 10 independent runs of the best solution found by each algorithm, for all combinations of $k = 2, 3, 4, 5$ and $n = 10, 15, 20, 25$.

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

*Bold numbers are highest max and underlined numbers are second highest max.

Table 1 reports the results of the comparison. SMBO is consistently the best both in terms of average best solution and max best solution for all combinations of n and k considered. Furthermore, in 12 out of 16 cases, SMBO was able to find the (estimated) real optimum. As the problem size (n) increases, SMBO becomes better in the comparison in terms of difference in performance. As expected, as the ruggedness (k) of the problem increases, the performance of SMBO and the other search algorithms decreases with regard to the difference in performance to the estimated optimum. As for the other algorithms in the

comparison, the population-based EA generally does better than (1+1) EA and RS. In particular, the population seems to help with larger instances of the problem. Perhaps surprisingly, random search often does better than (1+1) EA. This is because (1+1) EA can get easily trapped in local optima, whereas the solution found by random search exhibits a large variance in quality so the max best solution found can be competitive "by a stroke of luck", especially with small sample size and in small problems.

In summary, analogously to the case of continuous space, the surrogate model on the Hamming space really helps at finding better solutions than using standard search algorithms. This makes very promising the application of this framework to more complex solution representations associated with combinatorial spaces.

4 Conclusions and Future Work

There are many potentially interesting applications of a surrogate model based optimisation framework that can naturally encompass more complex representations beyond the traditional vector-based representation. We advocate that allowing a natural representation for the problem at hand makes the task of learning the underlying surrogate model easier, hence more efficient, as no extra non-linearity due to shoe-horning the solutions in vectors is introduced. Also, a direct approach to representations greatly enlarges the scope of SMBO to complex representations (e.g., Genetic Programming trees) which cannot be naturally mapped to vectors of features.

In this paper, we have outlined a conceptually simple, formal, general and systematic approach to adapt a SMBO algorithm to *any* target representation. This approach has been derived using a geometric methodology to generalise search algorithms from continuous to combinatorial spaces that has been successfully applied in the past to generalise other types of search algorithms. This methodology requires to write the original continuous algorithm only in terms of Euclidean distances between candidate solutions, which then can be generalised by replacing the Euclidean distance function with a generic metric. Then the formal algorithm obtained can be formally specified to any target representation by employing as underlying metric a distance rooted on the target representation (e.g., edit distance). Multivariate interpolation and regression methods to build surrogate models are well-suited to this methodology as they are inherently based on spatial notions. We showed how radial basis function networks can be naturally generalised to encompass any representation. This is possible because both the approximating model and the learning of the parameter of the model can be cast completely in a representation-independent way, and rely only on distance relations between training instances and query instances.

As a preliminary experimental validation of the framework, 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 best in a comparison with other search algorithms. This shows that this framework has

potential to work well on other more complex representation associated with combinatorial spaces.

Much work remains to be done. Firstly, we will test the framework on standard problems for more complex representations, such as permutations, variable-length sequences, and trees. Then, we will test how the system performs on a number of challenging real-world problems. We will also experiment with different types of radial basis functions, beside the Gaussian, and in a more complex learning settings (i.e., learning the centers and the widths of the radial basis functions). Lastly, we will attempt the generalisation of more sophisticated interpolation and regression methods, including gaussian process regression, which is state of the art in machine learning.

References

1. L. Bajer and M. Holena. Surrogate model for continuous and discrete genetic optimization based on rbf networks. In *Intelligent Data Engineering and Automated Learning IDEAL 2010*, 2010.
2. P. Castillo, A. Mora, J. Merelo, J. Laredo, M. Moreto, F. Cazorla, M. Valero, and S. McKee. Architecture performance prediction using evolutionary artificial neural networks. In *Applications of Evolutionary Computing*, pages 175–183, 2008.
3. N. A. C. Cressie. *Statistics for Spatial Data (revised edition)*. Wiley, 1993.
4. L. C. Jain. *Radial Basis Function Networks*. Springer, 2001.
5. Y. Jin. A comprehensive survey of fitness approximation in evolutionary computation. *Soft Computing Journal*, 9(1):3–12, 2005.
6. D. R. Jones. A taxonomy of global optimization methods based on response surfaces. *Journal of Global Optimization*, 21:4:345–383, 2001.
7. S. Kauffman. *Origins of order: self-organization and selection in evolution*. Oxford University Press, 1993.
8. 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*, 20:1819–1831, 2006.
9. T. Mitchell. *Machine Learning*. McGraw Hill, 1997.
10. A. Moraglio. *Towards a geometric unification of evolutionary algorithms*. PhD thesis, University of Essex, 2007.
11. C. E. Rasmusen and C. Williams. *Gaussian Processes for Machine Learning*. the MIT Press, 2006.
12. S. Tong and B. Gregory. Turbine preliminary design using artificial intelligence and numerical optimization techniques. *Journal of Turbomachinery*, 114:1–10, 1992.
13. I. Voutchkov, A. 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*, 194:3535–3551, 2005.