SIAM J. SCI. COMPUT.
Vol. 37, No. 2, pp. A1020–A1039
© 2015 Society for Industrial and Applied Mathematics

Downloaded 07/29/15 to 157.193.135.172. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

A FUZZY HYBRID SEQUENTIAL DESIGN STRATEGY FOR GLOBAL SURROGATE MODELING OF HIGH-DIMENSIONAL COMPUTER EXPERIMENTS*

J. VAN DER HERTEN†, I. COUCKUYT†, D. DESCHRIJVER†, AND T. DHAENE†

Abstract. Complex real-world systems can accurately be modeled by simulations. Evaluating high-fidelity simulators can take several days, making them impractical for use in optimization, design space exploration, and analysis. Often, these simulators are approximated by relatively simple math known as a surrogate model. The data points to construct this model are simulator evaluations meaning the choice of these points is crucial: each additional data point can be very expensive in terms of computing time. Sequential design strategies offer a huge advantage over one-shot experimental design because information gathered from previous data points can be used in the process of determining new data points. Previously, LOLA-Voronoi was presented as a hybrid sequential design method which balances exploration and exploitation: the former involves selecting data points in unexplored regions of the design space, while the latter suggests adding data points in interesting regions which were previously discovered. Although this approach is very successful in terms of the required number of data points to build an accurate surrogate model, it is computationally intensive. This paper presents a new approach to the exploitation component of the algorithm based on fuzzy logic. The new approach has the same desirable properties as the old method but is less complex, especially when applied to high-dimensional problems. Experiments on several test problems show the new approach is a lot faster, without losing robustness or requiring additional samples to obtain similar model accuracy.

Key words. sequential design, active learning, high-dimensional, experimental design, fuzzy inference systems

AMS subject classifications. 62L05, 62L86, 62K20, 65D05, 62P30

DOI. 10.1137/140962437

1. Introduction.

To avoid many real-life experiments and countless prototypes, modern engineering problems rely heavily on highly accurate computer simulations to reduce costs, time, and (potentially) risks. The simulations are used to help the engineer understand the relation between inputs and the outputs of the system, and to identify interesting regions in the design space.

The downside of using high-accuracy simulations is that one simulation of a complex system with several inputs (commonly referred to as *variables*), and outputs (also called *responses*) can be very expensive in terms of computation time [18, 14]. These lengthy or expensive computations often make it impractical to use simulations directly for design exploration and gaining insight into the complex system behavior. Most optimization algorithms require many simulations in the search space which makes optimization a computationally expensive task.

An extra abstraction layer can be used to expedite the process. The simulator (which approximates the real world) is approximated by *surrogate models* (also known

*Submitted to the journal's Methods and Algorithms for Scientific Computing section March 27, 2014; accepted for publication (in revised form) January 21, 2015; published electronically April 24, 2015. This research was partially funded by the Interuniversity Attraction Poles Programme BESTCOM initiated by the Belgian Science Policy Office.

http://www.siam.org/journals/sisc/37-2/96243.html

†Ghent University - iMinds, IBCN, Gaston Crommenlaan 8, Gent, 9050 Belgium (joachim.vanderherten@intec.ugent.be, ivo.couckuyt@intec.ugent.be, dirk.deschrijver@intec.ugent.be, tom.dhaene@intec.ugent.be). The second and third authors were supported by the Fund for Scientific Research in Flanders (FWO-Vlaanderen).

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 07/29/15 to 157.193.135.172. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

as response surface models or metamodels). These computationally cheap replacement models can be used to analyze or optimize the complex system while minimizing the required number of expensive simulations. For this study, we make two assumptions: the simulator is deterministic which means that running the simulation twice with the same input parameters always produces the same results. Second, the complex system is treated as a gray or black box (little or nothing is known about the inner working of the system).

Surrogate models can be used for optimization: in this context a *local* surrogate model is constructed to guide an optimization algorithm towards an optimum. Afterwards, the model is no longer of use and discarded. This is not the case in *global* surrogate modeling, which aims to construct a model that approximates the behavior of the system over the entire domain. This surrogate model can afterwards be used instead of the expensive simulator.

The simulator can be defined as an unknown function $f : \mathbb{R}^d \to \mathbb{C}$, which maps a d-dimensional input vector of real inputs to a possibly complex output. This function is sampled at a discrete set of N data points: $P = \{ \boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_N \}$. These data points (called the experimental design) are evaluated by the simulator and their responses are denoted as $F = \{ f(\boldsymbol{p}_1), f(\boldsymbol{p}_2), \ldots, f(\boldsymbol{p}_N) \}$. Based on this information a surrogate model $\tilde{f}$ is chosen from a set of candidate approximation functions. This choice is usually guided by predefined quality criteria (such as cross validation). Since acquiring the responses is computationally expensive but necessary to build an accurate surrogate model, the goal is to keep the set P as small as possible while still obtaining good accuracy. The choice of the data points in P is of crucial importance for constructing an accurate surrogate model with a reduced number of points. Intuitively, the data points should be spread over the domain in such a way that they capture a maximum amount of information on the behavior of f. Since f is considered to be a black box, this is a difficult task.

The LOLA-Voronoi algorithm [8, 7], an earlier proposed hybrid iterative scheme that distributes the points to cover the design space and distributes the data density proportional to the nonlinearity of f has proven to be very useful in several studies in several fields [1, 10, 28, 23, 2, 11, 27]. Nonlinear regions are more difficult to model, so the additional data points in these regions greatly help the search for a good approximation $\tilde{f}$. The algorithm combines an approach that estimates the gradient in the data points (based on a local linear approximation—LOLA), and a Voronoi space-filling approach. The downside of the LOLA algorithm is that it becomes very computationally demanding for high-dimensional design spaces. In this paper we propose a new fuzzy-based approach to overcome this issue. This approach can replace LOLA without further modifications to the concept of LOLA-Voronoi.

2. Sequential design. The selection of data points can be determined by means of a *one-shot* approach: all points are chosen at once and simulated. These data are given to the modeling algorithm and a surrogate model is constructed. The locations of the points in the design space is called the design of experiments (DoE). One-shot designs of computer experiments are usually space filling meaning they try to cover the domain as equally as possible. Examples are (maximin) Latin hypercubes [24] and fractional designs [25].

Sequential designs turn the one-shot approach into an iterative process. The data acquired and/or the constructed models from previous iterations are analyzed in order to intelligently select locations for new data points. These additional points are evaluated and usually new models are constructed. Sequential design has two

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 07/29/15 to 157.193.135.172. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

important benefits over one-shot designs: first of all, it is impossible to have too few or too many points; the iterative process is halted when the objectives are reached (i.e., the surrogate model meets the predefined accuracy goals [15]). In a one-shot setting too few points means restarting the process, whereas too many points means wasting time due to evaluating an expensive simulator more than required. Second, the information provided by the intermediate simulator responses and constructed models can be used to identify regions that are difficult to model. This allows the sampling distribution to be guided towards these regions.

2.1. Exploration and exploitation. Any sequential design method faces the trade-off between *exploration* and *exploitation*. Exploration involves exploring the complete design space for key regions such as discontinuities, steep areas, optima, and stable regions that have not yet been identified. Usually exploration does not look at system responses and focuses on filling the design space as evenly as possible. Undersampling and oversampling no longer occur when exploring the design space sequentially. Examples of sequential exploration methods can be found in [6]. Exploitation on the other hand analyzes simulator responses and/or constructed models in sample regions that have been identified as interesting. One could sample near optima or discontinuities to capture the complex behavior, or sample in regions where intermediate surrogate models make large errors. Examples of methods that involve exploitation of the available experimental design information can be found in [9, 20].

These two concepts conflict with each other: exploration aims to look away from regions we already know and focus on unexplored areas, whereas exploitation does the opposite and gathers more information about irregularities that have been spotted previously. If a sequential design only focuses on exploitation, certain key regions are potentially missed as the sequential design strategy is stuck sampling a region that was identified previously. To reduce this risk we could specify a large initial space-filling design, but this might result in oversampling the design space. Only focusing on exploration disallows the sampling distribution to be modified towards interesting regions as we end up with a sequential space-filling design. Finding a balance between exploration and exploitation can be done in many different ways, and can also be application dependent.

3. Exploitation using local approximations. Exploitation includes the responses from previous points to guide the sequential design process to interesting regions in the design spaces. The definition of interesting regions depends entirely on the context of the surrogate modeling process: for instance, for optimization interesting regions are those (possibly) containing optima. In the context of accurate global surrogate modeling this means distributing a minimal amount of points to find a model which accurately represents the systems response over the entire design space.

Previously, the LOLA algorithm was introduced to guide the sampling process towards regions in the domain that may be more difficult to approximate [8, 7]. Often, systems have a very linear response in a large part of the design space, but have one or more regions that behave very nonlinearly. Sampling more densely in these "difficult" regions has proven to be a successful approach for global surrogate modeling. The LOLA algorithm first estimates the gradient at each point, which is the best local linear approximation of the system response. This approximation is compared to the true simulator responses for nearby points. To compute the gradient approximation $\boldsymbol{g}$ in a point $\boldsymbol{p}_r$, a subset of $P_r = P \setminus \boldsymbol{p}_r$ is defined, known as the neighborhood: $N(\boldsymbol{p}_r) = \{ \boldsymbol{p}_{r1}, \ldots, \boldsymbol{p}_{rv} \} \subset P_r$. This set is used to solve the following least squares

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 07/29/15 to 157.193.135.172. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

problem:

$$\begin{pmatrix} p_{r1}^{(1)} - p_r^{(1)} & p_{r1}^{(2)} - p_r^{(2)} & \dots & p_{r1}^{(d)} - p_r^{(d)} \\ p_{r2}^{(1)} - p_r^{(1)} & p_{r2}^{(2)} - p_r^{(2)} & \dots & p_{r2}^{(d)} - p_r^{(d)} \\ \vdots & \vdots & & \vdots \\ p_{rv}^{(1)} - p_r^{(1)} & p_{rv}^{(2)} - p_r^{(2)} & \dots & p_{rv}^{(d)} - p_r^{(d)} \end{pmatrix} \begin{pmatrix} g^{(1)} \\ g^{(2)} \\ \vdots \\ g^{(d)} \end{pmatrix} = \begin{pmatrix} f(\boldsymbol{p}_{r1}) - f(\boldsymbol{p}_r) \\ f(\boldsymbol{p}_{r2}) - f(\boldsymbol{p}_r) \\ \vdots \\ f(\boldsymbol{p}_{rv}) - f(\boldsymbol{p}_r) \end{pmatrix}. \tag{3.1}$$

LOLA requires $v \geq 2d$, so this system is never underdetermined. Using this gradient information to predict the values of other points in the neighborhood, we can compute the error between the linear approximation and the true simulator response at these points:

$$E(\boldsymbol{p}_r) = \sum_{i=1}^{v} |f(\boldsymbol{p}_{ri}) - (f(\boldsymbol{p}_r) + \boldsymbol{g}.(\boldsymbol{p}_{ri} - \boldsymbol{p}_r))|. \tag{3.2}$$

The error E is referred to as the *nonlinearity score*. The region surrounding $\boldsymbol{p}_r$ is nonlinear if the error is large, as a linear prediction will be insufficient to capture the system response locally. Theoretically, LOLA chooses new points in locations near points with high nonlinearity scores. In practice however, LOLA is combined with a Voronoi exploration based component (see section 5.1).

3.1. How to determine the neighborhood. A key issue in LOLA is how to determine the neighborhood $N(\boldsymbol{p}_r)$ of a point $\boldsymbol{p}_r$, which is referred to as the *reference point*. Determining this set is essentially a multiobjective optimization problem which optimizes two criteria:

1. *Cohesion*: A neighbor should be as close to the reference point as possible, as we are constructing a local approximation.
2. *Adhesion*: The neighbors should be as far away from each other as possible, in order to cover the space surrounding the reference point.

Clearly, it is impossible to maximize both. If the neighbors are very close to the reference point (high cohesion), they are close to each other as well (high adhesion). Points further away can have better adhesion, but can result in a bad local approximation. Unfortunately, there is no known general solution to place an arbitrary number of points in an ideal configuration on a (hyper)sphere [5].

The original LOLA algorithm [7] solves this optimization problem by comparing a neighborhood with an optimal configuration known as the cross-polytope. This configuration always has $2d$ points (which explains the constraint of $v \geq 2d$). This configuration is intuitive: for one-dimensional problems this means one neighbor on each side of the reference point, for two dimensions this is a square, etc. For each $\boldsymbol{p}_r$, all possible sets of v points are constructed and compared to the cross-polytope. The set which resembles the cross-polytope is chosen as $N(\boldsymbol{p}_r)$. When new points are available, each point in the neighborhood is removed and replaced by a new point. If this results in a better configuration, then the neighborhood is updated. This solution is very elegant and leads to quasi-optimal configurations in terms of cohesion and adhesion. Because in a cross-polytope configuration the vectors $\boldsymbol{p}_{ri} - \boldsymbol{p}_r$, $i = 1 \dots v$, are orthogonal and the method generates neighborhoods resembling a cross-polytope, this results in a well-conditioned system for (3.1).

However, the downside of this approach is its complexity: $\mathrm{O}\big(2^{2d} N N_{\mathrm{new}}\big)$ (N_{new} represents the amount of new samples the algorithm proposes for evaluation). Two optimizations to the algorithm were proposed, affecting mostly the N and N_{new} com-

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 07/29/15 to 157.193.135.172. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

ponents. The "too far" heuristic excludes certain points from addition to the neighborhood as they are mathematically unable to improve the neighborhood. This makes the algorithm very powerful and usable for low-dimensional problems. Issues appear, however, when using LOLA to build global surrogate models for problems of higher dimensionality. As d becomes larger the neighborhood size increases, which causes each new point to result in many new candidate neighborhoods that need to be evaluated. Additionally, due to the *curse of dimensionality* more points will be required to obtain sufficient information to construct an accurate surrogate model. Problems of four dimensions and higher will spend a very long time on determining where to choose new samples when using the LOLA algorithm for exploitation.

3.2. Novel approach to determine the neighborhood. Surrogate modeling of high-dimensional systems can be computationally very demanding as a lot of (expensive) data points are required to construct accurate models. The complexity of many modeling types, such as Kriging and radial basis function (RBF) models, scales badly with sample size and design space dimensionality. Having a sequential sampling algorithm that adds to the computational burden is undesirable. In this section a new approach to determining $N(\boldsymbol{p}_r)$ is introduced. This approach requires computing weights to include information about cohesion and adhesion. The weight computation is covered in section 4.

The original neighborhood selection procedure [7] is selective: no matter how many points surround the reference point, a fixed number of neighbors (v) is selected which means that, in some cases, valuable information is neglected. The new algorithm therefore includes all points within a certain range α of the reference point:

$$N(\boldsymbol{p}_r) = \{\, \boldsymbol{p} \mid \boldsymbol{p} \in P_r, ||\boldsymbol{p} - \boldsymbol{p}_r|| < \alpha \,\}. \tag{3.3}$$

We assume that each parameter was scaled to compatible ranges, and an appropriate distance metric is used. In section 5.2, a brief discussion on distances in high-dimensional spaces is given. The regulatory α parameter in (3.3) controls the part of the input space that is included in the gradient estimation. It defines the notion "local" for $\boldsymbol{p}_r$. It can be proportional to the average distance between points, or it can be time controlled. In this paper the following heuristic was used:

$$\alpha = \frac{2}{K} \sum_{j=1}^{K} ||\boldsymbol{n}_j - \boldsymbol{p}_r|| \tag{3.4}$$

with $\boldsymbol{n}_j$ the jth nearest neighbor of $\boldsymbol{p}$ in the input space. The heuristic represents twice the average distance to the K nearest neighbors. The parameter K is chosen as a function of dimensionality; for all experiments in this paper it was chosen to be $4d$.

When a point is very isolated, α will be large to include sufficient points in the gradient estimation to avoid an underdetermined system. In a dense region, a smaller alpha will only include points that are sufficiently close to obtain an accurate gradient. If points that are distant would be included, they could smooth out the gradient in the case of small nonlinearities. Unfortunately, in the case of a very isolated point, (3.4) still can result in $|N(\boldsymbol{p}_r)| < d$, which turns (3.1) into an underdetermined system. In this situation, α is raised to $||\boldsymbol{n}_d - \boldsymbol{p}_r||$ to include the d nearest neighbors.

Note how this definition of $N(\boldsymbol{p}_r)$ no longer selects points based on adhesion and cohesion as defined above, however, we still require including this information in our gradient estimation. This issue is covered by assigning weights to each neighbor. In the next section we come up with a strategy to assign the weights.

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 07/29/15 to 157.193.135.172. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

As we are no longer chasing the cross-polytope, we risk instability when solving (3.1). However, due to the nature of experimental design, points will still be spread out over the design space as much as possible, leading to a surrounding configuration. Because of this property most of the vectors $\boldsymbol{p}_{ri} - \boldsymbol{p}_r$, $i = 1 \dots v$ have different directions, which results in a well-conditioned matrix.

4. Determining the neighbor weights. Attaching proper weights to each neighbor of $\boldsymbol{p}_r$ and solving (3.1) as a weighted least squares problem reintroduces the concept of cohesion and adhesion. The weights reflect how much influence each point in $N(\boldsymbol{p}_r)$ has in the gradient estimation. Points with high cohesion and low adhesion are preferred and are assigned high weights, while low cohesion and/or high adhesion result in low weights.

First, we mathematically define cohesion and adhesion $\forall \boldsymbol{p} \in N(\boldsymbol{p}_r)$:

$$C(\boldsymbol{p}_r, \boldsymbol{p}) = ||\boldsymbol{p} - \boldsymbol{p}_r||, \tag{4.1}$$

$$A(\boldsymbol{p}_r, \boldsymbol{p}) = \min_{\boldsymbol{q} \in P_r} ||\boldsymbol{q} - \boldsymbol{p}||. \tag{4.2}$$

High cohesion means points are very close, which corresponds to lower values for C, compared to other points in $N(\boldsymbol{p}_r)$. On the other hand, low adhesion corresponds to large values for A. For simplicity, $C(\boldsymbol{p}_r)$ and $A(\boldsymbol{p}_r)$ are vectors which represent cohesion and adhesion values for all neighbors of $\boldsymbol{p}_r$. In section 4.2 a system based on fuzzy logic is defined to determine the neighbor weights. The concept of a Mamdani fuzzy inference system (FIS) is first explained in the next section.

4.1. Mamdani FIS. An FIS maps inputs to outputs, using fuzzy set theory. Common types include the Mamdani and the Sugeno FISs. In this section we briefly explain the concept of the first type.

Fuzzy sets and concepts were introduced as a way to represent data imprecisely. An example is the weight of a person: we can express the weight numerically using a number with a unit, but we can also treat a person's weight as a *linguistic variable* with *linguistic values*. Someone can be skinny, normal, or heavy. Typical for these type of statements is the lack of clear boundaries: when is someone no longer skinny but normal? Usually there is a gray zone between the linguistic values. Mathematically this is expressed by means of fuzzy sets. A crisp[1] set A has a simple membership function $\chi_A : A \to \{0, 1\}$: an element is either a member or not. For a fuzzy set, the membership function is less strict and takes the form of $\mu_A : A \to [0, 1]$. Usually a membership value of zero indicates complete nonmembership, whereas one represents complete membership. Values in-between indicate intermediate degrees of membership.

Built on the theory of fuzzy sets, an FIS consists of a fuzzifier, an inference engine, and a defuzzifier. The fuzzifier maps crisp inputs of linguistic variables to fuzzy set memberships, using provided membership functions. These membership degrees are fed into a rule-based inference engine, which processes rules of the form "if-then." To process the rules, we need to be able to process operations such as AND and OR within the rules. In fuzzy logic, these operations are known as fuzzy combinations. Many possible operations have been proposed; in this paper the minimum t-norm and maximum t-conorm are used.

The output of these rules (of which some might not be activated, depending on the input) is combined (usually by applying a fuzzy OR) and defuzzified. A

[1] A traditional set.

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 07/29/15 to 157.193.135.172. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

popular method for defuzzification is the centroid method. For more information about Mamdani FISs, the reader is referred to [21]. More information on fuzzy logic and the t-(co)norms can be found in [13].

4.2. Fuzzy-based neighbor weight assignment. Although we defined crisp values for cohesion and adhesion with (4.1) and (4.2), it is clear that handling these two quantities as linguistic variables is much more convenient. In fact, reasoning with crisp values for cohesion and adhesion makes the problem complex. In what follows, an FIS $\mathcal{S}$ is proposed to assign weights to each point in $N(\boldsymbol{p}_r)$. The system has two input parameters, cohesion and adhesion, and produces a weight as output.

For the cohesion input parameter, one fuzzy set referred to as "high" with membership function $\mathcal{C}_{\text{high}}$ is defined

$$\begin{aligned} \mathcal{C}_{\text{high}}: \quad & [0,\alpha] \to [0,1], \\ & x \mapsto \frac{1}{1+\exp\left(-\sigma_c x\right)}. \end{aligned} \tag{4.3}$$

Points that have a small distance to the reference point $\boldsymbol{p}_r$ have a high membership degree (corresponding to high cohesion) of the fuzzy set, points that are far away do not.

For adhesion, two fuzzy sets with membership functions $\mathcal{A}_{\text{low}}$ and $\mathcal{A}_{\text{high}}$ exist:

$$\begin{aligned} \mathcal{A}_{\text{low}}: \quad & [0, A_{\max}] \to [0,1], \\ & x \mapsto \exp\left(\frac{-(x-A_{\max})^2}{2(A_{\max}\sigma_{al})^2}\right); \\ \mathcal{A}_{\text{high}}: \quad & [0, A_{\max}] \to [0,1], \\ & x \mapsto \exp\left(\frac{-x^2}{2(A_{\max}\sigma_{ah})^2}\right). \end{aligned}$$

$A_{\max}$ is the maximum adhesion value for $A(\boldsymbol{p}_r)$. σ_c, σ_{al}, and σ_{ah} are the hyperparameters of the membership functions. Larger values for the σ values result in wider Gaussian membership functions, which means higher membership values to the sets. It is possible to define a single adhesion membership function and define the other as the negation (as for the cohesion), but two membership functions allow more control for the adhesion parameter. For the output, 3 triangular membership functions are defined (low, average, and high) as shown in Figure 1(a). The following rules complete the FIS definition:

1. IF cohesion is high AND adhesion is low THEN weight is high.
2. IF cohesion is high AND adhesion is high THEN weight is average.
3. IF cohesion is NOT high AND adhesion is low THEN weight is average.
4. IF cohesion is NOT high AND adhesion is high THEN weight is low.

The system processes the cohesion and adhesion for each $\boldsymbol{p} \in N(\boldsymbol{p}_r)$ and computes the membership degree for the fuzzy sets $\mathcal{C}_{\text{high}}$, $\mathcal{A}_{\text{low}}$, and $\mathcal{A}_{\text{high}}$ (this step is referred to as *fuzzification*). Next, all rules are evaluated to assign a degree of membership to the triangular output member functions. The output degree of membership corresponds to the result of the evaluation of the rule expressions. For example, after fuzzification a point has a membership degree of 0.5 in $\mathcal{C}_{\text{high}}$, 0.15 in $\mathcal{A}_{\text{low}}$, and 0.45 in $\mathcal{A}_{\text{high}}$. According to the first rule, this point has a degree of membership of 0.15 in the fuzzy set high.[2] The output membership function is then clipped by the obtained degree of

[2]This is due to the choice of minimum as t-norm and maximum as t-conorm. For more information on norms and fuzzy logic, the reader is referred to [13].

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

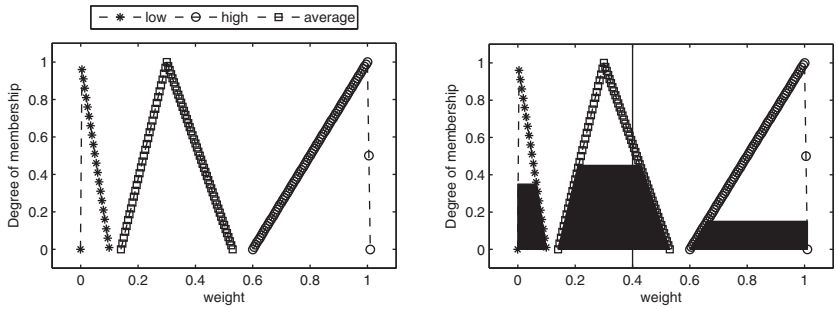


FIG. 1. *Figure* (a) *shows the output membership functions low, average, and high for* $\mathcal{S}$. *In* (b), *the clipped functions for the example with input membership degrees of* 0.5 *in* $\mathcal{C}_{high}$, 0.15 *in* $\mathcal{A}_{low}$, *and* 0.45 *in* $\mathcal{A}_{high}$ *are shown. The vertical line indicates the weight obtained by defuzzification of the resulting output membership distribution by the centroid method.*

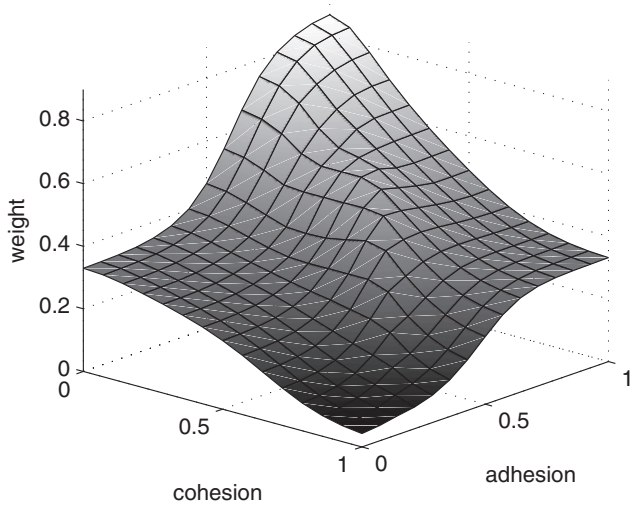


FIG. 2. *Example of the response surface of the FIS* $\mathcal{S}$ *for* $\alpha = 1$, $A_{max} = 1$, $\sigma_c = 0.3$, $\sigma_{al} = 0.27$, *and* $\sigma_{ah} = 0.3$. *The cohesion and adhesion on the axes correspond to* (4.1) *and* (4.2), *not the response of the membership functions.*

membership (y-axis in Figure 1). By applying a fuzzy OR over the obtained output membership values, the final output membership distribution is obtained (Figure 1(b) shows the effect for the example). The centroid defuzzification method is then used to convert the result into a crisp value for the weight (x-axis in Figure 1).

Figure 2 shows the response surface of $\mathcal{S}$. Highly cohesive points with low adhesion are preferred, as opposed to low cohesive points with high adhesion. It is possible to use different membership functions (for example, sigmoid instead of Gaussian membership functions), or to define more fuzzy sets both for inputs as well as the output. Throughout the rest of this paper, we use the FIS defined above.

An illustration of how weights are assigned by $\mathcal{S}$ is shown in Figure 3. The ideal configuration of the points in two dimensions (the cross-polytope) is illustrated in Figure 3(a). Each point is assigned an equal weight, which is not surprising as all cohesion and adhesion values are identical. Figure 3(b) shows a more complex

Downloaded 07/29/15 to 157.193.135.172. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

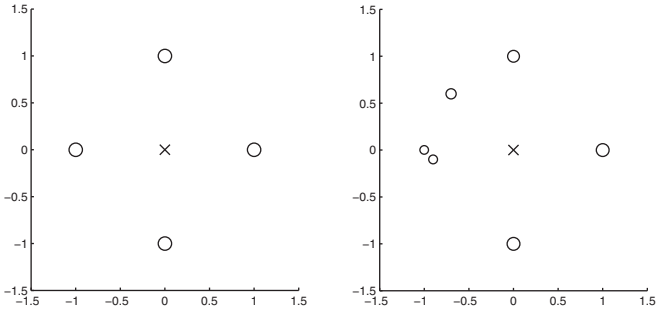


FIG. 3. *Illustration of weight assigments with $\mathcal{S}$ using the same constants as Figure* 2. *The cross indicates the reference point $\boldsymbol{p}_r$, the size of each neighbor the weight.* (a) *shows the ideal (cross-polytope) configuration: all weights are equal. In* (b), *the left side has three points: the weights have been distributed amongst them.*

situation. Instead of one sample at $(-1, 0)$ we now have three points: two close to each other, and a third one somewhat further away. In this case, the weight is divided amongst the two neighboring points. If the weight of these points would be added, they would roughly add up to the weights of the previous case, which means that more points are contributing to the gradient estimation. The third point on the left at $(-0.7, 0.6)$ is further away and provides information about a different direction. Therefore it is assigned a higher weight, although it doesn't have the same impact as the other stand-alone points. In fact, it has taken over some weight from the two points at $(-1, 0)$ and the point at $(0, 1)$.

5. New hybrid sequential design method. In section 3 a previously introduced exploitation method was discussed. A key complexity issue for high-dimensional problems was identified and a new approach to construct the neighborhoods was introduced. Section 4 introduced a fuzzy-based mechanism to assign weights to each neighbor, based on cohesion and adhesion. All these concepts are now brought together into a new approach that can take the place of the LOLA algorithm in LOLA-Voronoi.

5.1. Fuzzy LOLA. The weights computed by $\mathcal{S}$ can be used to solve (3.1) as a weighted least squares[3] problem to estimate the gradient. After obtaining $\boldsymbol{g}$, we can compute the nonlinearity score. An overview of this new approach, known as *fuzzy local linear approximation* (FLOLA), is given in Algorithm 1. Because the size of the system (3.1) is not dependent on the size of the set P, only the for-loop contributes to the complexity of the algorithm: this results in a complexity of $\mathrm{O}(N)$ which is a massive improvement compared to LOLA. Furthermore, the for-loop allows parallel computation since each iteration is independent. The biggest costs are many distance calculations to determine α, $A(\boldsymbol{p}_r)$, and $C(\boldsymbol{p}_r)$. This is solved by computing a distance matrix once prior to the for-loop: this matrix contains all required information for computations inside the loop. Distances matrices tend to occupy a lot of memory in the case of many points, which is unlikely for this algorithm in the context of surrogate modeling as each evaluation is expensive. Due to the limited size of the set P, the size of the matrix is always manageable.

[3]Note that the weights are computed for each point $\boldsymbol{p}$ separately. This essentially means we turned the gradient estimation into a moving least squares problem.

Downloaded 07/29/15 to 157.193.135.172. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 07/29/15 to 157.193.135.172. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

ALGORITHM 1. FLOLA: THIS EXPLOITATION ALGORITHM COMPUTES A SCORE $\forall \boldsymbol{p} \in P$, INDICATING THE NONLINEARITY OF THE REGION SURROUNDING $\boldsymbol{p}$. NEW SAMPLES ARE CHOSEN IN THE NEIGHBORHOOD OF THE N_{NEW} HIGHEST RANKED SAMPLES.

Require: P, F, σ_c, σ_{al}, σ_{ah}, N_{new}
  initiate $\mathcal{S}$ (section 4.2)
  Calculate distance matrix for P
  for all $p_r \in P$ **do**
    Compute α (3.4)
    Initialize $N(\boldsymbol{p}_r)$ (3.3)
    Determine $C(\boldsymbol{p}_r)$ and $A(\boldsymbol{p}_r)$ (4.1), (4.2)
    Compute weights W by evaluating $\mathcal{S}$
    Estimate $\boldsymbol{g}$ (3.1), with respect to W
    Calculate error on gradient estimation (3.2)
  end for
  Pick N_{new} samples with highest nonlinearity score
  P_{new} = new samples in the neighborhood of these samples
  $P = P \cup P_{\text{new}}$

5.2. Including an exploration metric. Similarly to LOLA, the exploitation based algorithm FLOLA can be complemented with a Voronoi approximation based exploration component and form *FLOLA-Voronoi*. For each point $\boldsymbol{p}_r$, the nonlinearity score E_{fuzzy} is complemented with a measure V indicating an approximation of the relative Voronoi cell size of the reference point. For more information on approximating the size of a Voronoi cell, the reader is referred to [7]. The value of V is in the range $[0, 1]$ so E_{fuzzy} is first normalized and then added to V:

$$H_{\text{fuzzy}}(\boldsymbol{p}_r) = V(\boldsymbol{p}_r) + \frac{E_{\text{fuzzy}}(\boldsymbol{p}_r)}{\sum_{i=1}^{N} E_{\text{fuzzy}}(\boldsymbol{p}_i)}. \tag{5.1}$$

For clarity, the pseudocode of FLOLA-Voronoi is shown in Algorithm 2. The only difference with LOLA-Voronoi is the algorithm used to calculate $E_{\text{fuzzy}}(\boldsymbol{p}_r)$. The hybrid score H is then used to rank all currently available points according to the nonlinearity and the sample density of the surrounding region. The N_{new} highest ranked reference points are selected to assign new points in the next iteration. The position of the point is determined by considering local space fillingness. Usually the position maximizing the minimum distance from both the reference point as well as its neighbors is chosen.

The combination of both criteria guarantees we do not get stuck in one region of the design space and no large areas are left unexplored. However, the exploitation score pushes the strategy to sample nonlinear regions much denser when they are discovered. When these regions are sampled dense enough, the FLOLA score will be lower, and exploration will take over. This additional information on nonlinear regions helps the surrogate model to capture the nonlinear behavior accurately as more information is provided on irregularities. In (5.1), the exploration and exploitation components contribute equally. It is possible to use a different balance, or even change the balance dynamically as more samples become available. For more information, please refer to [26].

Setting $N_{\text{new}} = 1$ is optimal, as each sampling decision can be made with the latest information at hand. This means that when a new nonlinear region is discovered it

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 07/29/15 to 157.193.135.172. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

ALGORITHM 2. FLOLA-VORONOI: HYBRID SEQUENTIAL STRATEGY. COMBINES AN EXPLOITATION AND AN EXPLORATION SCORE (FLOLA AND VORONOI RESPECTIVELY) AND SELECTS A NEW CANDIDATE SAMPLE IN THE NEIGHBORHOOD OF THE N_{NEW} HIGHEST RANKED SAMPLES.

Require: P, F, σ_c, σ_{al}, σ_{ah}, N_{new}
  for all $p_r \in P$ **do**
    Calculate $E_{\text{fuzzy}}(\boldsymbol{p}_r)$ (3.2)
    Calculate $V(\boldsymbol{p}_r)$ (see [7])
    Compute $H_{\text{fuzzy}}(\boldsymbol{p}_r)$ (5.1)
  end for
  Sort P by H_{fuzzy}
  for $i = 1$ **to** N_{new} **do**
    $\boldsymbol{p}_{\text{new}} \leftarrow$ location near $\boldsymbol{p}_i$
    $P_{\text{new}} \leftarrow P_{new} \cup \boldsymbol{p}_{\text{new}}$
  end for

is exploited immediately. However, choosing to add more samples each iteration does not lead to undesired clusters or a bad design, since only one additional point can be placed in each Voronoi cell during one iteration. For high-dimensional problems this is recommended as fitting a surrogate model may be expensive.

5.3. Note on distances in high-dimensional spaces. Throughout the entire paper, distance between vectors $a, b \in \mathbb{R}^d$ was indicated as $||a - b||$, without specifying the distance metric. The most commonly used distance metric is the Euclidean distance which is essentially a Minkowski distance (5.2) for $p = 2$:

$$||a - b||_p = \left(\sum_{i=1}^{d} (|a_i - b_i|)^p \right)^{1/p}. \tag{5.2}$$

However, in high-dimensional spaces the Euclidean distance fails to provide a meaningful notion to the concept of proximity. This is known as the concentration of norms, and affects all Minkowski distances for $p \geq 1$ [12]. As a solution, fractional distances can be used. In fact this is a Minkovski distance with $p \in [0, 1]$. This does not solve the concentration effect but reduces the impact. During our experiments fractional distances are used for the high-dimensional problems with $p = \frac{1}{d}$, to test if they result in better designs. For problems of very high dimensionality, $p = \frac{1}{\lfloor \log(d) \rfloor + 1}$ can be used to avoid very difficult dth root computations (which is extremely slow).

6. Experimental setup. In previous studies [1, 10, 28, 23, 2, 11, 27] in several research fields, LOLA-Voronoi has proven to be an excellent algorithm for building sequential designs. The sampling distribution is modified to focus on nonlinear regions at the expense of a small computational cost for low-dimensional problems. For high-dimensional problems this cost quickly magnifies, which can be countered by using FLOLA-Voronoi. Throughout all experiments, the hyperparameters σ_c, σ_{al}, and σ_{ah} of the membership functions of FLOLA are fixed at 0.3, 0.27, and 0.3, respectively.

In this section, we will first show by means of simple two-dimensional problems that the new algorithm performs very similarly to LOLA-Voronoi and has the same desirable properties. Next, higher-dimensional problems are modeled to illustrate the performance gain of the new algorithm. Next for FLOLA-Voronoi and LOLA-Voronoi, four more sequential design strategies are tested: the first is a solely Voronoi-based sequential design which is a pure exploration based method. More recently, a new

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 07/29/15 to 157.193.135.172. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

algorithm known as the Delaunay-hybrid adaptive sequential design (DHASD) was proposed [3]. This method combines an exploitation and exploration metric based on a Delaunay triangulation, and dynamically balances between the two. The balancing strategy relies on several predefined parameters to avoid clustering. These parameters heavily influence the performance of the sampling strategy but need to be chosen by an expert in the function of the design space and the complexity of the problem at hand. This is a disadvantage for practical applications where nothing is known in advance. For our test cases, the parameters were chosen by trial and error.

An exploitation based model error strategy was included as well: this strategy evaluates the best models of previous generations on a dense grid. The outputs are compared and new samples are chosen in regions with the largest differences. Since evaluating surrogate models is cheap, evaluating a dense grid is not computationally demanding. Furthermore, this method requires the construction of the intermediate surrogate models, which can have a considerable cost. All other methods do not require this, i.e., FLOLA-Voronoi only needs the simulator responses. To conclude, random sampling has also been included in the experiments.

All problems start with a small initial design. Sequentially, samples are added while intermediate models are constructed to evaluate the accuracy that can be obtained with the current set of samples. This iterative process continues until a target accuracy of 0.05 is reached for the root relative square error (RRSE) on a dense preevaluated validation set:

$$\mathrm{RRSE}(x, \tilde{x}) = \sqrt{\frac{\sum_{i=1}^{N} (x_i - \tilde{x}_i)^2}{\sum_{i=1}^{N} (x_i - \bar{x})^2}}.$$

x_i represents the true simulator responses in all samples, $\tilde{x}_i$ the estimate by the surrogate model, and $\bar{x}$ the mean. The model type for each problem was chosen based on prior knowledge about the test cases. Often this information is not available; in these cases automatic model type selection approaches can be used as described in [4].

6.1. SUMO research platform. To perform the experiments, the surrogate modeling (SUMO) MATLAB toolbox[4] was used. Designed as a research platform for sequential sampling and adaptive surrogate modeling featuring high extensibility, this MATLAB toolbox makes it very easy to implement and compare this new sampling approach to other sequential design methods using several model types.

The workflow of the SUMO toolbox is illustrated in Figure 4. Starting point is an initial design, which is generally a sparse space-filling design such as Latin hypercubes or a fractional design. After evaluation of these points, the main modeling loop is initiated. A set of surrogate models is built and scored using a set of measures (i.e., cross validation, validation set, ...). Usually models have a set of hyperparameters that can be tuned using optimization algorithms. Examples of hyperparameters are the order of numerator and denominator for rational models, the network architecture for neural networks, etc. During the hyperparameter optimization step, new models are constructed until no further improvement can be made. If the target accuracy (in terms of the measures) has not been reached yet, the sequential design routine is called to select more points to be evaluated. When the system responses to these new

[4]The SUMO Toolbox R2014a, including an implementation of FLOLA-Voronoi, can be downloaded from http://www.sumo.intec.ugent.be, allowing full reproduction of all experiments in this article.

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 07/29/15 to 157.193.135.172. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

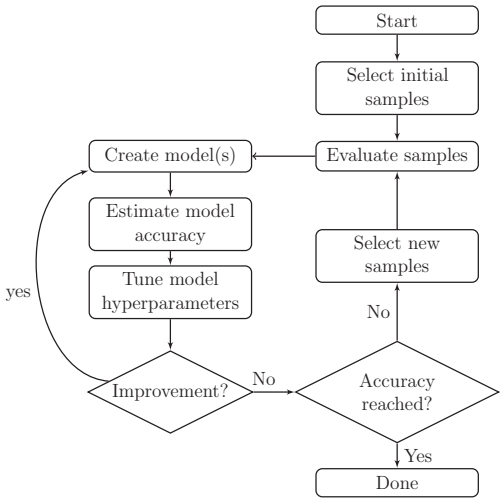


FIG. 4. *Flowchart of the SUMO toolbox workflow.*

points are available, the toolbox starts a new modeling loop. For more information on the architecture and the different components of the SUMO toolbox, please refer to [16].

6.2. Low-dimensional test cases. The goal of the low-dimensional experiments is to illustrate the equivalence of LOLA and FLOLA. As our goal is to evaluate the performance of sequential design strategies, a minimal initial design consisting of a Latin hypercube of 10 points combined with a 2-level factorial design was used. Each iteration, a single point is added to this set. For all test cases, each experiment was repeated ten times to reduce noise by random factors in the SUMO toolbox (for example randomization in the hyperparameter optimization process). A visual representation of each test case is given in Figure 5.

6.2.1. Case 1: Peaks. The first test case is a two-dimensional problem known as Peaks. The surface is flat, with a few Gaussian distributions in the center of the domain. This function is very useful to illustrate the concept of (F)LOLA-Voronoi: as a large part of the input domain is flat, an increased focus on the nonlinear region will result in fewer samples required to reach the target accuracy. Kriging with a Gaussian correlation function was used as the surrogate model type: due to the nature of this model type it is very suitable for modeling the Gaussian distributions.

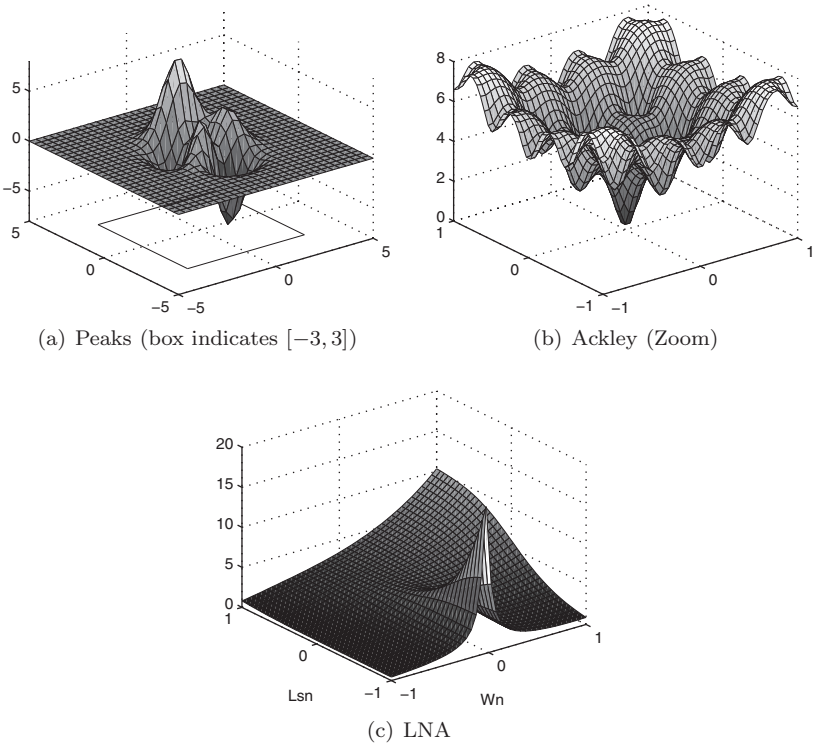
Three cases on different domains are considered: $[-3,3]^2$, $[-5,5]^2$, and $[-8,8]^2$. The first case is zoomed in on the nonlinear region. As the input range grows, the quasi-flat surface surrounding the nonlinear central region grows, and (F)LOLA-Voronoi is expected to be more efficient.

6.2.2. Case 2: Ackley function. Ackley's path, a function well known from optimization is used as a second test case. For a d-dimensional problem, it is defined as

$$F(\boldsymbol{x}) = -20\exp\left(-0.2\sqrt{\frac{1}{d}\sum_{i=1}^{d} x_i^2}\right) - \exp\left(\frac{1}{d}\sum_{i=1}^{d}\cos(2\pi x_i)\right) + 20 + e$$

with $x_i \in [-2,2]$. The function is modeled in two dimensions with RBFs.

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.



(a) Peaks (box indicates $[-3, 3]$) (b) Ackley (Zoom)

(c) LNA

FIG. 5. *Illustrations of the low-dimensional test cases.*

6.2.3. Case 3: Low-noise amplifier (LNA). This test case consists of a real-world problem from electronics. An LNA, which is a simple RF circuit, is the typical first stage of a receiver, providing the gain to suppress noise of subsequent stages. The performance of an LNA can be determined by means of computer simulations where the underlying physical behavior is taken into account. For this experiment we chose to model the input-noise current $\sqrt{\overline{i_{in}^2}}$, as a function of two (normalized) parameters: the inductance L_{sn} and the MOSFET width W_n. The relation to the real parameters is defined as

$$W = 100 \cdot 10^{-6} \cdot 10^{W_n} \text{ m},$$
$$L_s = 0.1 \cdot 10^{-9} \cdot 10^{L_{sn}} \text{ H}.$$

The input domain of this test case is smooth with a steep ridge for $W_n = 0$. A full description of the LNA problem can be found in [17]. The chosen model type for this problem is artificial neural networks, trained with Levenberg–Marquard backpropagation with Bayesian regularization (300 epochs). The network topology and initial weights are optimized by a genetic algorithm.

6.2.4. Results. Results of the low-dimensional test cases are shown in Table 1. For the Peaks test cases, the results confirm the expectation: FLOLA- and LOLA-Voronoi clearly perform better compared to the other methods in all three cases. As the quasi-flat region surrounding the nonlinear central region grows, the advantage over the other methods increases. This observation is confirmed by the LNA test

Downloaded 07/29/15 to 157.193.135.172. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 07/29/15 to 157.193.135.172. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

TABLE 1

Summary of the results for the two-dimensional test cases. Each problem was modeled with different sampling strategies until an RRSE of 0.05 *over a preevaluated validation set was reached. Each experiment ran ten times to cancel out noise by random factors. The* 95% *confidence intervals are shown between brackets.*

	Peaks $[-3, 3]$	Peaks $[-5, 5]$	Peaks $[-8, 8]$	Ackley	LNA
FLOLA-Voronoi	71 (67, 73)	99 (93, 105)	**145** (140, 149)	259 (244, 273)	63 (58, 68)
LOLA-Voronoi	**65** (61, 68)	**97** (87, 105)	147 (142, 153)	262 (244, 278)	**62** (56, 69)
DHASD	79 (75, 83)	115 (106, 123)	192 (179, 205)	352 (326, 379)	88 (79, 97)
Voronoi	96 (93, 98)	229 (219, 238)	643 (589, 698)	**243** (233, 251)	103 (95, 112)
Model Error	104 (98, 110)	281 (253, 308)	851 (768, 934)	262 (250, 273)	118 (105, 132)
Random	105 (94, 116)	270 (240, 300)	1042 (842, 1242)	430 (400, 458)	165 (136, 194)

case: the steep ridge is sampled much more densely by both methods, which leads to a satisfying model with 40% fewer samples than required for the next best method.

Only the Ackley test case behaves somewhat differently: FLOLA- and LOLA-Voronoi and model error result in very comparable results, but the pure exploration method (Voronoi) performs slightly better for this test case. This is not unexpected: the Ackley function is nonlinear over the entire interval. There is no benefit of balancing between exploration and exploitation as the sample density should be more or less the same. The exploitation scores do not provide an advantage and sometimes influence the strategy to pick a sample which is not in the largest Voronoi cell. The negative impact of the exploitation score for this test case is quite limited, and can be further reduced by improving the selection of the new candidate in the highest ranked cell.

DHASD is a recent approach, and was presented as an alternative for LOLA-Voronoi with the ability to generate better designs. In our study the method performs quite average. Possibly, better results can be obtained by adjusting the parameters of the method to result in better balancing between exploration and exploitation: since there is no automatic way to do this it is a serious disadvantage of the method, especially when nothing is known about the system in advance. Another possible cause for the performance of DHASD could be related to the combination with different surrogate model types: the method has only been tested in combination with Kriging [3]. However, the Peaks problem was modeled with Kriging and DHASD requires more samples.

Although the FLOLA algorithm is less complex and lifts the strict constraints on neighborhoods of the LOLA algorithm, the low-dimensional experiments indicate the capability of the new algorithm to produce comparable results in terms of the number of samples required to reach a predefined target accuracy.

6.3. High-dimensional test cases. To illustrate the speed of the new algorithm, two high-dimensional test cases are considered. Global surrogate modeling of high-dimensional problems is not easy, due to the curse of dimensionality. Adding samples one by one and reconstructing the models would be a very lengthy process. To avoid this, samples are added in batches. After each modeling iteration, the sample selection strategy is run once, and a batch of new candidates is selected for evaluation ($N_{\text{new}} > 1$). Many model types are not able to reach a very strict accuracy for high-dimensional problems: at some point their accuracy will not improve much, and the exact location of the samples does not have a big impact. Futhermore, due to the maximum walltime for jobs on the UGent HPC infrastructure, each run was given a time limit of 72 hours which may be too short for some strategies to obtain the target accuracy (RRSE of 0.05). To obtain a better comparison between the

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 07/29/15 to 157.193.135.172. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

sequential design strategies, the number of samples required for an RRSE of 0.1 is included as well. To evaluate the performance, the running times of the the LOLA, FLOLA, and Voronoi components are recorded separately each time the respective algorithm is run. The DHASD method was excluded from the high-dimensional test cases. Delaunay triangulation of high-dimensional datasets with many points is an infeasible lengthy process. Additionally, figuring out the parameters of DHASD by trial and error is difficult. Because the input spaces are high dimensional, the experiments with FLOLA-Voronoi were performed with both Euclidean (l_2) and fractional ($l_{1/d}$) distances to study the impact of the distance metrics. The LOLA-Voronoi algorithm was not modified to use fractional distances, as many optimizations of the LOLA algorithm rely on the use of Euclidean distance. As for the low-dimensional test cases, each experiment was repeated ten times to reduce noise by random factors in the SUMO toolbox.

6.3.1. Case 1: Hartmann six dimensions (6D). As a first test case, a six-dimensional Hartmann function was chosen. The function is not very complex so it can be modeled in reasonable time, but it does feature some areas that are more difficult to model. As initial design, 400 points were generated using a Monte Carlo approach as described in [6]. The sample selection batch size was set to 50 samples. As model type, least-squares (support vector machines) SVMs [29] were chosen as they have a fixed number of 2 parameters: one kernel parameter and the amount of noise. These parameters were optimized with the DIRECT algorithm [19].

6.3.2. Case 2: Styblinski–Tang eight dimensions (8D). The Styblinski–Tang function is a test function from optimization. In d dimensions, it is defined as

$$f(\boldsymbol{x}) = \frac{\sum_{i=1}^{d} x_i^4 - 16x_i^2 + 5x_i}{2}$$

for $-5 \leq x_i \leq 5$. The central region is quite flat, but towards the bounds of the interval the function is suddenly steep. It is expected usage of (F)LOLA-Voronoi will be advantageous as the bounds will be sampled more densely. A 2-level factorial design complemented with a Latin hypercube of 244 points generated by the TPLHD algorithm [30] was used as initial design. Each iteration 50 samples are added by the sequential design strategy. The chosen model type was again least-squares SVMs optimized with the DIRECT algorithm.

6.3.3. Results. The number of samples required to reach the target accuracy is shown in Table 2. For the Hartmann test case, FLOLA- and LOLA-Voronoi clearly outperform the other methods. This confirms that the results of the low-dimensional experiments hold for higher-dimensional problems. Surprisingly, model error sampling performs worse than random sampling for this test case. Figure 6(a) indicates the average runtime for each of the LOLA, FLOLA, and Voronoi components over the 10 runs for both distance metrics, as a function of the evaluated samples available before the sampling iteration. For Euclidean distance, the new algorithm is a lot faster compared to LOLA. When selecting new samples with 3000 evaluated samples, computing the scores with LOLA takes 15 minutes, compared to only a few seconds with FLOLA. This is also reflected in the total runtime of the experiment: on average, a run with FLOLA-Voronoi takes 3 hours to complete, compared to 13 hours with LOLA-Voronoi! When using fractional distances, computing distance matrices becomes a lot more expensive because of the nth root. The impact on FLOLA is

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 07/29/15 to 157.193.135.172. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

TABLE 2

Summary of the required sample size to reach two different target accuracies for high-dimensional test cases. Each experiment ran ten times to cancel out noise by random factors. The 95% confidence intervals are shown between brackets. In (b)*, some runs did not finish due to a time constraint of 72 hours. In those cases, the number of selected samples is shown as is the average accuracy at that point.*

(a) RRSE of 0.1 on validation set

	Hartmann 6D	Styblinski-Tang 8D
FLOLA-Voronoi (l_2)	1530 (1479, 1582)	**3340** (3307, 3374)
FLOLA-Voronoi ($l_{1/d}$)	1881 (1827, 1935)	3359 (3285, 3433)
LOLA-Voronoi	**1520** (1469, 1572)	4255 (4084, 4426)
Voronoi (l_2)	2265 (2217, 2313)	3525 (3476, 3573)
Model error	2241 (2106, 2376)	7255 (6917, 7593)
Random	2266 (2185, 2347)	4155 (4046, 4264)

(b) RRSE of 0.05 on validation set

	Hartmann 6D	Styblinski-Tang 8D
FLOLA-Voronoi (l_2)	**3200** (3134, 3268)	**6899** (6744, 7054)
FLOLA-Voronoi ($l_{1/d}$)	3676 (3591, 3761)	$> 5500 (RRSE \approx 0.08)$
LOLA-Voronoi	3296 (3234, 3358)	$> 6900 (RRSE \approx 0.07)$
Voronoi (l_2)	4606 (4506, 4706)	7750 (7632, 7868)
Model error	5311 (4318, 6304)	$> 10000 (RRSE \approx 0.07)$
Random	4951 (4818, 5084)	$> 10000 (RRSE \approx 0.06)$

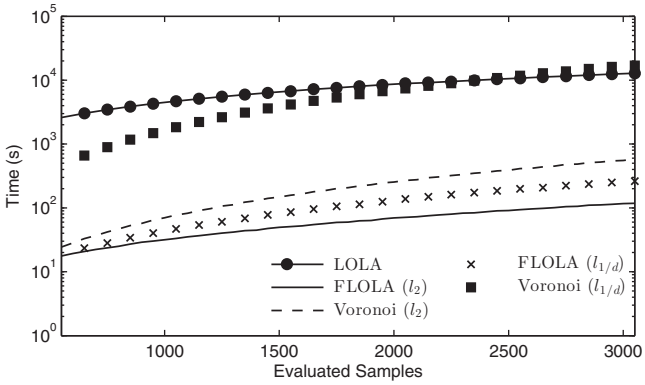
limited, but the performance of the Voronoi approximation is heavily affected when using fractional distances and becomes very slow.

For the eight-dimensional Styblinski–Tang function not a single run with LOLA-Voronoi, model error, or random sampling managed to reach the strict target accuracy (RRSE = 0.05) within the time constraint. Runs with LOLA-Voronoi were ended after 72 hours with approximately 6900 samples selected; a large part of the time was spent on sample selection. The average accuracy of runs with LOLA-Voronoi at this point was 0.07. Model error managed to select up to 10000 points. Unfortunately, this was not sufficient to reach the target accuracy. FLOLA-Voronoi with fractional distances also failed to reach the target accuracy, mainly due to the poor performance of the Voronoi component with fractional distances. Only (Euclidean) Voronoi sampling and FLOLA-Voronoi managed to reach the target accuracy in time, the latter using 10 percent fewer samples.

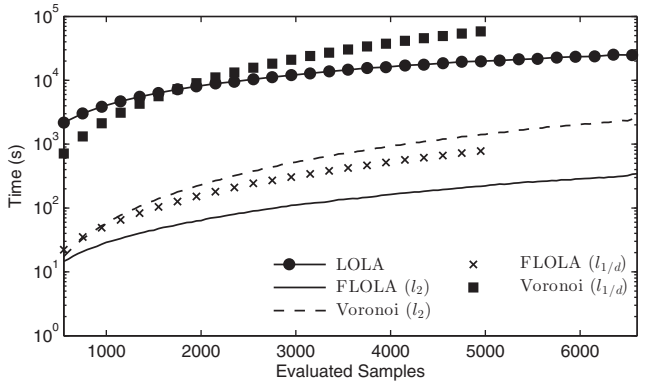
FLOLA-Voronoi with Euclidean distance is the most efficient method to reach the target accuracy for both problems. It is a lot faster compared to LOLA-Voronoi, and requires fewer samples compared to all other methods. The usage of fractional distance slows the algorithm down and does not seem to provide a benefit. However, the fractional distance does seem to have a slight impact when modeling the eight-dimensional test case. Figure 7 shows the evolution of the RRSE as more samples are added each iteration. Clearly, FLOLA-Voronoi with fractional distance brings down the error faster, which means initially the space is covered better. However, when 1800 samples have been selected, the version based on Euclidean distance has caught up and both methods have similar errors. At this point, the input space has been saturated up to a level which prevents fractional distance from being better at covering the input space. This effect is likely to be more present for problems of higher dimensionality; in this case the usage of FLOLA-Voronoi with fractional distance may be appropriate to obtain a qualitative model faster when samples are very expensive and the additional runtime of the algorithm is not an issue. Furthermore, the compu-

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 07/29/15 to 157.193.135.172. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php



(a) Hartmann 6D



(b) Styblinski-Tang 8D

FIG. 6. *Runtime of LOLA, FLOLA, and Voronoi for the high-dimensional test cases. For the Styblinski–Tang function, the experiments of FLOLA-Voronoi with fractional distance were interrupted because the time limit was reached.*

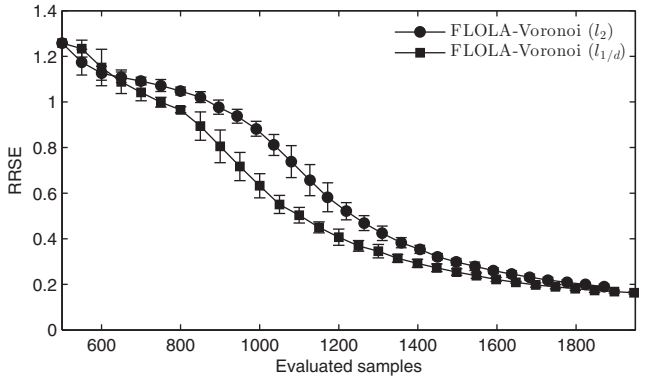


FIG. 7. *Comparison of FLOLA-Voronoi with Euclidean and fractional distance at the beginning of the modeling process of the Styblinski–Tang 8D function, for all ten runs (whiskers indicate the standard deviation). Clearly, the fractional distance is able to bring down the error of the model faster, but the version with Euclidean distance catches up at around 1800 samples.*

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 07/29/15 to 157.193.135.172. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

tation of the fractional distance matrix can benefit greatly from GPU computation. An implementation in CUDA [22] runs approximately 5x faster than the optimized CPU implementations used for these experiments.

7. Conclusion. The LOLA-Voronoi method has proven in the past to outperform other sequential design methods for several model types and problems. It does not require intermediate models to make sampling decisions and has been applied to multiple real-world test cases from different problem domains by users of the SUMO toolbox in several studies [1, 10, 28, 23, 2, 11, 27]. The performance of this method comes at the cost of computational complexity, which grows rapidly as the dimensionality of the problem increases.

This paper presents a novel approach which replaces the computationally complex LOLA algorithm with a fuzzy variant, FLOLA. Experiments show similar results indicating the new approach has the benefits of the original algorithm, but as it has a complexity of $\mathrm{O}(N)$ the overall time to build a global surrogate model of a high-dimensional problem reduces dramatically. Recent developments such as adaptive balancing of both components [26] are also applicable to this new algorithm.

Currently, new candidate points (samples) are chosen in the design space near the highest ranked samples, based on the maximin distance to existing samples. Better options to improve this local space fillingness will be investigated in further work. The reduced complexity allows the construction of global surrogate models of problems with higher dimensionality. In this study, the use of the fractional distances did not offer a lot of advantages. Only in an eight-dimensional problem was a slight advantage in the beginning of the modeling process noticed. A more thorough study of the impact of high-dimensional spaces on aspects of the surrogate modeling process based on Euclidean distance (model types and measures) is the subject of further research.

Acknowledgments. The authors would like to thank NXP Semiconducters and Jeroen Croon in particular for providing the LNA simulator code. Experiments were carried out using the STEVIN Supercomputer Infrastructure at Ghent University, funded by Ghent University, the Flemish Supercomputer Center (VSC), the Hercules Foundation, and the Flemish Government department EWI. Ivo Couckuyt and Dirk Deschrijver are postdoctoral research fellows of the Research Foundation Flanders (FWO).

REFERENCES

[1] J. Aernouts, I. Couckuyt, K. Crombecq, and J.J.J. Dirckx, *Elastic characterization of membranes with a complex shape using point indentation measurements and inverse modelling*, Internat. J. Engrg. Sci., 48 (2010), pp. 599–611.

[2] S. Aerts, D. Deschrijver, W. Joseph, L. Verloock, F. Goeminne, L. Martens, and T. Dhaene, *Exposure assessment of mobile phone base station radiation in an outdoor environment using sequential surrogate modeling*, Bioelectromagn., 34 (2013), pp. 300–311.

[3] A. Ajdari and H. Mahlooji, *An adaptive exploration-exploitation algorithm for constructing metamodels in random simulation using a novel sequential experimental design*, Comm. Statist. Simulation Comput., 43 (2014), pp. 947–968.

[4] I. Couckuyt, D. Gorissen, F. De Turck, and T. Dhaene, *Automatic surrogate model type selection during the optimization of expensive black-box problems*, in Proceedings of the 2011 Winter Simulation Conference, IEEE, Piscataway, NJ, 2011, pp. 4269–4279.

[5] H.T Croft, K.J. Falconer, and R.K. Guy, *Unsolved Problems in Geometry*, Springer, New York, 1991.

[6] K. Crombecq, I. Couckuyt, D. Gorissen, and T. Dhaene, *Space-filling sequential design strategies for adaptive surrogate modelling*, in Proceedings of the First International Con-

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

ference on Soft Computing Technology in Civil, Structural and Environmental Engineering (CSC 2009), Civil-Comp. Proc. 92, Civil-Comp. Press, UK, 2009, 50.

[7] K. Crombecq, D. Gorissen, D. Deschrijver, and T. Dhaene, *A novel hybrid sequential design strategy for global surrogate modeling of computer experiments*, SIAM J. Sci. Comput., 33 (2011), pp. 1948–1974.

[8] K. Crombecq, D. Gorissen, L. De Tommasi, and T. Dhaene, *A novel sequential design strategy for global surrogate modeling*, in Proceedings of the 2009 Conference on Winter Simulation, Austin, TX, IEEE, Piscataway, NJ, 2009, pp. 731–742.

[9] P. de Oliveira Castro, E. Petit, J.C. Beyler, and W. Jalby, *ASK: Adaptive sampling kit for performance characterization*, in Euro-Par 2012 Parallel Processing, Lecture Notes in Comput. Sci. 7484, Springer, Berlin, 2012, pp. 89–101.

[10] D. Deschrijver, K. Crombecq, H.M. Nguyen, and T. Dhaene, *Adaptive sampling algorithm for macromodeling of parameterized-parameter responses*, IEEE Trans. Microwave Theory Tech., 59 (2011), pp. 39–45.

[11] D. Deschrijver, F. Vanhee, D. Pissoort, and T. Dhaene, *Automated near-field scanning algorithm for the EMC analysis of electronic devices*, IEEE Trans. Electromagn. Compat., 54 (2012), pp. 502–510.

[12] D. Francois, V. Wertz, and M. Verleysen, *The concentration of fractional distances*, IEEE Trans. Knowledge Data Engrg., 19 (2007), pp. 873–886.

[13] R. Fullér, *Neural Fuzzy Systems*, Åbo Akademi, Åbo, 1995.

[14] K. Goethals, I. Couckuyt, T. Dhaene, and A. Janssens, *Sensitivity of night cooling performance to room/system design: Surrogate models based on CFD*, Building Environ., 58 (2012), pp. 23–36.

[15] D. Gorissen, I. Couckuyt, E. Laermans, and T. Dhaene, *Multiobjective global surrogate modeling, dealing with the* 5*-percent problem*, Eng. Comp., 26 (2010), pp. 81–89.

[16] D. Gorissen, K. Crombecq, I. Couckuyt, P. Demeester, and T. Dhaene, *A surrogate modeling and adaptive sampling toolbox for computer based design*, J. Mach. Learn. Res., 11 (2010), pp. 2051–2055.

[17] D. Gorissen, L. De Tommasi, K. Crombecq, and T. Dhaene, *Sequential modeling of a low noise amplifier with neural networks and active learning*, Neural Comput. Appl., 18 (2009), pp. 485–494.

[18] D. Gorissen, W. Hendrickx, K. Crombecq, and T. Dhaene, *Adaptive distributed metamodeling*, in Proceedings of 7th International Meeting on High Performance Computing for Computational Science (VECPAR 2006), Lecture Notes in Comput. Sci. 4395, M. Dayde et al., eds., Springer, Berlin, 2007, pp. 579–588.

[19] D.R. Jones, C.D. Perttunen, and B.E. Stuckman, *Lipschitzian optimization without the Lipschitz constant*, J. Optim. Theory Appl., 79 (1993), pp. 157–181.

[20] Y. Lin, *An Efficient Robust Concept Exploration Method and Sequential Exploratory Experimental Design*, Ph.D. thesis, Georgia Institute of Technology, Atlanta, GA, 2004.

[21] E.H. Mamdani and S. Assilian, *An experiment in linguistic synthesis with a fuzzy logic controller*, Internat. J. Man Mach. Stud., 7 (1975), pp. 1–13.

[22] CUDA Nvidia, *Programming Guide*, http://scholar.google.be/scholar?hl=en&q=nvidia+cuda (2008).

[23] B. Rosenbaum and V. Schulz, *Comparing sampling strategies for aerodynamic Kriging surrogate models*, ZAMM Angew. Math. Mech., 92 (2012), pp. 852–868.

[24] T. Simpson, D. Lin, and W. Chen, *Sampling strategies for computer experiments: Design and analysis*, Internat. J. Reliabil. Appl., 2 (2002), pp. 209–240.

[25] T. Simpson, J.D. Poplinski, P.N. Koch, and J.K. Allen, *Metamodels for computer-based engineering design: Survey and recommendations*, Eng. Comput. (Lond.), 17 (2001), pp. 129–150.

[26] P. Singh, D. Deschrijver, and T. Dhaene, *Balanced sequential design strategy for global surrogate modeling*, in Proceedings of the 45th Conference on Winter Simulation, Austin, TX, IEEE, Piscataway, NJ, 2013, pp. 2172–2179.

[27] P. Singh, D. Deschrijver, D. Pissoort, and T. Dhaene, *Adaptive classification algorithm for EMC-compliance testing of electronic devices*, Electron. Lett., 49 (2013), pp. 1526–1528.

[28] D.W. Stephens, D. Gorissen, K. Crombecq, and T. Dhaene, *Surrogate based sensitivity analysis of process equipment*, Appl. Math. Model., 35 (2011), pp. 1676–1687.

[29] J.A.K. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor, J. Vandewalle, J.A.K. Suykens, and T. Van Gestel, *Least Squares Support Vector Machines*, Vol. 4, World Scientific, River Edge, NJ, 2002.

[30] F.A.C. Viana, G. Venter, and V. Balabanov, *An algorithm for fast optimal Latin hypercube design of experiments*, Internat. J. Numer. Methods Engrg., 82 (2010), pp. 135–156.

Downloaded 07/29/15 to 157.193.135.172. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.