Communications in Applied and Industrial Mathematics,
ISSN 2038-0909, **1**, 2, (2010) 110–126

DOI: 10.1685/2010CAIM545

Adaptive sampling with a Lipschitz criterion for accurate metamodeling

Alberto Lovison[1], Enrico Rigoni[2]

[1] *Dipartimento di Matematica Pura ed Applicata, Facoltà di Scienze MM.FF.NN.*
Università degli Studi di Padova, Italy
lovison@math.unipd.it

[2] *ESTECO–Enginsoft Tecnologie per l'Ottimizzazione*
AREA Science Park, Trieste, Italy
rigoni@esteco.it

Communicated by Alessandro Chessa

Abstract

Lipschitz Sampling, unlike standard space filling strategies (Minimax and Maximin distance, Integrated Mean Squared Error, Eadze-Eglais, etc.) for producing good metamodels, incorporates information from output evaluation in order to estimate in some sense the local complexity of the function at hand. The complexity indicator considered is a suitable definition of local Lipschitz constant. New points are proposed to be evaluated where the product of the local Lipschitz constant by the distance from the nearest already evaluated point is maximum.
Benchmarks are proposed on standard test functions in comparison with standard space filling strategies. Smaller prediction errors are obtained by Lipschitz sampling when the function considered shows sudden variations in some part of the domain and varies more slowly in other regions.

Keywords: `Metamodeling, Design of Experiments, Minimax and Maximin Distance Design, Lipschitz constant`

AMS Subject Classification: `05A16, 65N38, 78M50`

1. Introduction.

This paper deals with the problem of approximating a nonlinear unknown function, representing the outcome of some physically interesting experiment. We assume that no measurement error is involved, i.e., repeated experiments with the same input settings return exactly the same function values. In the *design of experiments* (DoE) literature, these are referred to as *computer experiments*, because usually those are numerical codes mimicking some relevant physical phenomenon. Sometimes this pro-

Received 2010 30 09, in final form 2011 02 14
Published 2011 03 12

DOI: 10.1685/2010CAIM545

cess is known as *metamodeling*, because the aim is to substitute the numerical code, the model of the physical process, with a numerically inexpensive surrogate metamodel, i.e., a model of the model [1–6].

We are not interested here in a specific approximation method, which could be a neural network, a sum of radial basis functions, a gaussian process, indifferently, but rather on the positions where the test points should be placed in the experimental domain. In other words we want to find the optimal design of a computer experiment with respect to the approximation error over the whole domain. For the benchmarks, we will employ *gaussian processes*, because they have an universal character (an equivalence with radial basis functions and neural networks can be established) and they are commonly used and referred to in literature [2,6].

One obviously argues that the optimal set of input points depends on the unknown function to be modeled. Indeed, the standard way to tackle this problem is to find a DoE which could serve for a whole class of functions, independently on the specific case. The answer is a DoE essentially built on a criteria of spatially uniform distribution, which is defined in one stage, without involving the actual function values in the process.

Our proposal is a sequential procedure which defines a local complexity criterion based on the function values accumulated insofar and then selects new input values accordingly. A number of benchmarks is conducted for low dimensional examples, showing that in general a blind sequential space filling DoE is outperformed by the strategy proposed here.

2. Metamodeling

Metamodeling, also referred to as Response Surface Methodology, or Surrogate Function, is a practice adopted in recent times for extracting relevant information from the outcomes of costly or lengthy experiments.

The scope is to determine the functional relation between the inputs and the outcome of an experiment, e.g., the output of a FEM code, or the concentration of the synthesized compound in a chemical process, or the human development index for a possible world scenario of resources and consumption [7] et cetera. The canonical procedure consists in extracting the main effects (first and second order contributions) of the inputs on the output (see [8–10]), or furthermore, as in the case we are interested in, to extract a the full nonlinear functional relation, in some suitable form [11,12]. The substitution of the experiment (the real function) by a metamodel (the surrogate function) can be performed when the following conditions are met:

1. the numerical surrogate, i.e., the *metamodel*, is sufficiently flexible to reproduce the desired function behavior.

2. The function behavior should be predictable. The more the function is smooth, the more it is simple to reproduce it.
3. Finally it is necessary to have collected enough information on the function: the test points should be denser than the characteristic range of variation of the function. In analogy with Nyquist, it is necessary to have enough points to avoid aliasing phenomena.

We will focus mainly on the last issue.

Which are then the appropriate criteria for collecting the information? In literature this task is referred to as *exploration*. Indeed it is reasonable, if we do not know anything about the function we should fire our shots all around the experimental region. Quoting T. Santner [2],

> *"Because we don't know the true relation between the response and inputs, designs should allow one to fit a variety of models and should provide information about all portions of the experimental region."*

Moreover, because we assume that our experiments are deterministic, it is a waste to repeat the same experiment or to perform experiments on too close locations. The choice should be also independent on the particular features of the metamodel employed, because we also do not know which could be the best suited kind of metamodel for the problem at hand.

Therefore we could name our task to *fill the space uniformly*. This can be done easily, and without ambiguity in the one dimensional case, but it is far than trivial to accomplish such a program in the multivariate case.

In fact, there exists in literature a number of possible approaches, each one with its own motivations.

3. Space filling strategies

We could group the space filling DoEs in two main class. In the class of *sampling based* strategies we have among others:

1. regular grids,
2. random sequences,
3. quasi random sequences (e.g., Halton, Sobol', Hammersley) [13],
4. stratified samplings as latin hypercube designs [14],
5. randomized orthogonal arrays [2].

Criterion based strategies contain

1. uniform design [15],
2. maximin/minimax distance design [16],

DOI: 10.1685/2010CAIM545

3. tessellation (n-volume) based,
4. statistical/entropy based [1].

For the sampling based ones it is emphasized the special algorithm/strategy which generates the sequence of points. Important features characterizing one method if compared with the others are for instance the possible sample sizes in relation with the spatial dimension, the possibility to add further points at later stages or how they look like when they are projected on subspaces.

On the other hand, for the criterion based family the emphasis is shifted to the optimality principle which should be satisfied by the set of points, without mentioning how this optimality is reached. The fact is that there are several different ways to define precisely the idea of *"uniformity"*.

3.1. *Mixed strategies*

Trying to accomplish one of the optimality criteria is usually desirable, because the sampling strategies alone are either not satisfactory or are not useful in practical cases. E.g., regular grids could be the best choice, but require exponentially large samples as the dimension grows. On the other hand, finding optimal samples is medium sized dimensional spaces is a computationally tough problem (finding the best displacement of n points in a d–dimensional space is a problem with $n \times d$ degrees of freedom). Moreover, some of the criteria are also difficult to compute, because they require the computation of an integral over the domain, or need to find an extremal value for an auxiliary function. In the everyday practice, the most effective choice seems to adopt a *mixed strategy*, i.e., choosing the best realization of an optimality criterion among a sufficiently large number of candidates produced by a sampling strategy.

3.1.1. *Generator*

A fast and reliable strategy for generating candidate samples is *Latin Hypercube Design (LHD)* [14]. Latin Hypercube designs (LHD) are a special kind of *stratified sampling* [14] which have been conceived for numerical quadrature with a faster convergence speed than pure Monte Carlo. An LHD of n points is defined by partitioning each coordinate variable in n *strata* with the same probability and by extracting $d-1$ permutations $\sigma_j(\cdot)$ of n symbols. The first point of the design is defined by picking its first coordinate in the $\sigma_1(1)$th stratum, its second variable in the $\sigma_2(1)$th stratum and therefore the jth coordinate of the ith point has to be picked in the $\sigma_j(i)$th stratum of the jth variable. Values can be taken at the center

of the stratum or picked randomly according to the probability density prescribed.

LHD possess several desirable properties: they are *flexible* (i.e., an n–points sample can be defined for any $d < n$ dimension), they are easily generated, and they show *good marginalization*). In other words, the projection of the LHD on each coordinate follows the prescribed probability distribution, because there is only a point for every stratum by construction (see Figure 1). However, the points could not be disposed at best a priori, indeed

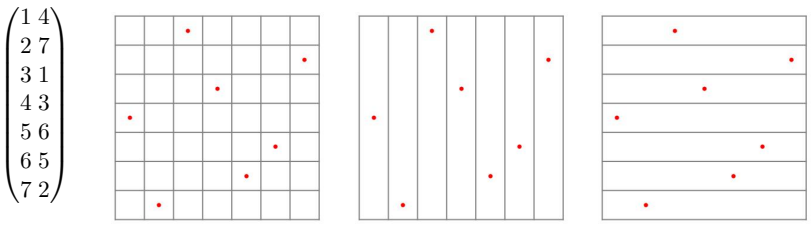

Fig. 1. A Latin Hypercube Design of n in two dimensions is defined by a permutation of n symbols. The projections of the design on the coordinate variables are uniformly distributed.

there possibly emerge clusters or undesired correlations among variables. Consider for instance the trivial permutation $\sigma(i) \equiv i$ and the (absolutely valid) LHD associated.

3.1.2. *Criterion*

A sound and tractable criterion is the *Maximin distance* [16]. A maximin distance design (MmD) is the set of n points $\overline{D}$ in a d–dimensional hypercube $\mathcal{H}_d$ which maximizes the minimum distance among pairs of points:

$$\overline{D} := \max_{\substack{D \subseteq \mathcal{H}_d \\ \#\{D\}=n}} \min_{x_1, x_2 \in D_p} \|x_1 - x_2\| \,.$$

In a sense, points are repelling each other, therefore clusterization phenomena are certainly avoided, however, a drawback is that, especially in large dimensions, points are likely to be scattered on the boundary of the domain, taking no points in its interior. When looking at the marginal distributions, i.e., at the projections on the individual variables, points are all collapsed on the intervals extrema.

Another serious problem already mentioned is the exponential computational complexity of the problem of finding such a design. In fact, although

DOI: 10.1685/2010CAIM545

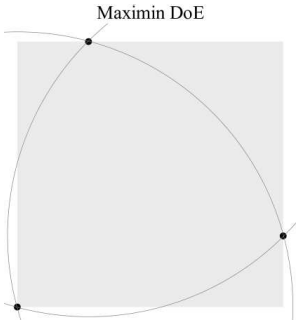

Fig. 2. A maximin distance design of 3 points in the square.

the criterion is quite easily computed, the number of variables is large ($n \times d$) and the problem presents several nontrivial symmetries.

3.1.3. *Mixing the criteria*

A mixed strategy, as suggested before, is maybe the most effective way of tackling the question and overcoming the difficulties discussed above. The coupling of LHD as a generator of candidate designs with the Maximin distance criterion is guaranteed to generate samples with good marginal properties and avoiding the correlations among variables. Moreover, because of the simplicity of the definitions, large sets of tentative samples can be generated and tested, giving with a good probability to attain an almost optimal design.

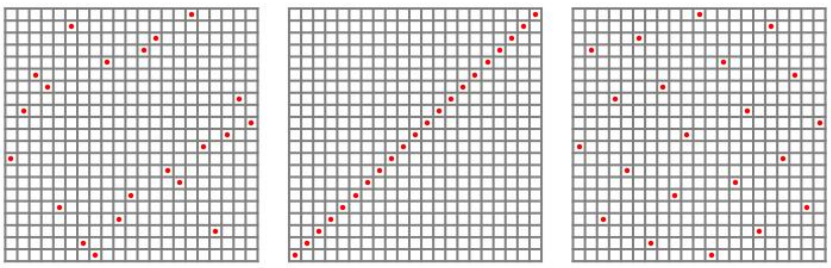

Fig. 3. Comparison of a purely random LHD (left), a pathologic LHD with extreme correlation (center) and the 21 points Maximin LHD (right).

All those features make the Maximin LHD one of the most used designs for fitting metamodels to computer experiments (see [2] and the references therein).

4. Adaptive samplings

Uniform space exploration may be the *best* approach for getting a good average performance over a broad class of unknown functions, but clearly it could be a limiting choice when considering a specific problem.

Indeed, as long as the function values are computed and accumulated, the emerging specific behavior of the unknown function could reasonably be involved in the decision process of the domain exploration. In fact, it is a waste proceeding in evaluating new sites without taking into consideration the information accumulated, especially when function evaluations are costly or lengthy. Some subregion of the experimental domain could reveal an incremented difficulty in being modeled, thus requiring a more intense exploration than other less complicated regions. Clearly this intuition must be more precisely defined. The function plotted in figure 4 for instance,

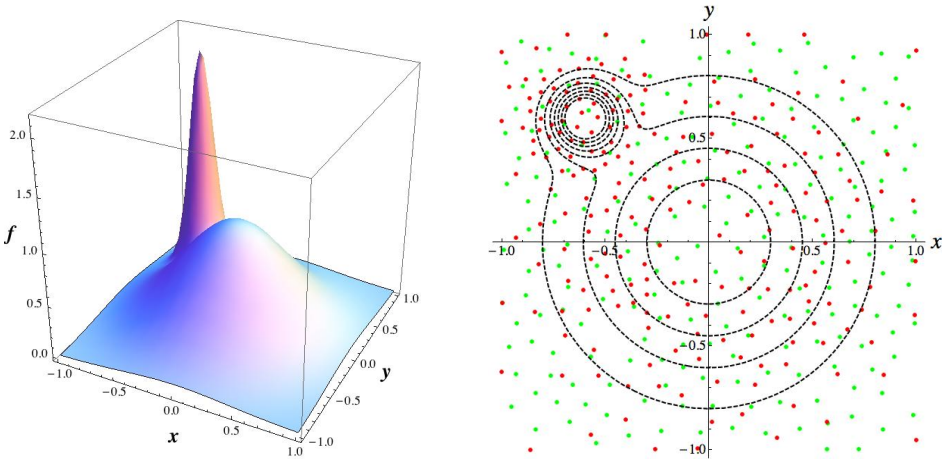

Fig. 4. An example of a smooth function with a region where sudden variations happen. Green dots are disposed according a space filling criterion. Red dots, on the other hand, concentrate in the peaked zone.

shows a general smooth behavior almost everywhere (some *low frequency* character) except for a small region where the function is markedly peaked. We would like to define a sampling strategy capable of *concentrating* in the *high frequency* zone, while spreading uniformly in the remaining zones, as exemplified by the sample of red points. For doing so we must define a *local indicator of complexity* for the function, which, on the basis of the function values accumulated, would reflect the needed sampling density for accurately reproducing the function behavior. We will define such an indicator noticing that the magnitudes of first and second derivatives should be pos-

DOI: 10.1685/2010CAIM545

itively correlated with the difficulty of reproducing the function behavior. Indeed, if derivatives are small the function cannot jump and wiggle too much.

5. A Lipschitz criterion

It is well known that in the study of partial differential equations, Lipschitz constants are employed for bounding the nonlinear character, and therefore the complexity, of the functions involved.

$$L_{f,D} := \sup_{\substack{x_1,x_2\in D \\ x_1\neq x_2}} \frac{|f(x_1)-f(x_2)|}{|x_1-x_2|} \tag{1}$$

The Lipschitz constant can be considered as a complexity indicator also for other reasons, for instance because it gives an upper bound on the number of oscillations of a given amplitude (see figure 5). Lipschitz constant also establishes bounds on the maximum and minimum value a function can assume in a given range: if the distance between two points x_1, x_2 is $d > 0$, the difference between the corresponding responses $f(x_1), f(x_2)$ cannot be larger than $L_f \cdot d$. On the other hand, phenomena such as the curvature,

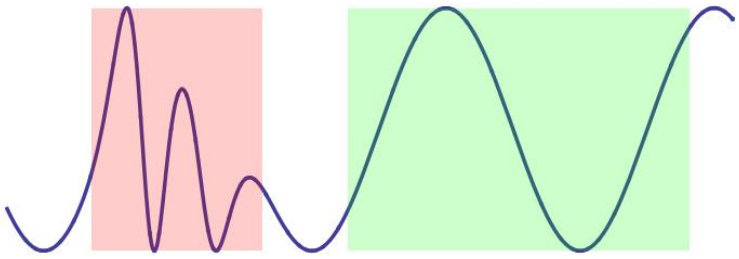

Fig. 5. A function with a subregion with a small Lipschitz constant (green) an another region with a larger one (red).

certainly related to the complexity of a response, are not captured by the Lipschitz constant, therefore alternative definitions of complexity, complementing Lipschitz, should be devised and compared. Moreover, little can be said about the smoothness of a function with finite Lipschitz constant. Indeed, Lipschitz functions are continuous and almost everywhere differentiable, but sharp angles (infinite curvature) can occur. These properties are involved and exploited in a family of global optimization algorithms referred to as *Lipschitz optimization* (see for instance [5,17] and the references therein).

We proceed by defining a compound complexity indicator which will be attached to every component of a suitable tessellation of the domain.

5.1. *Lipschtiz sampling algorithm*

In Algorithm 5.1 we state precisely what we mean by Lipschitz sampling. The procedure is illustrated with an example in figure 6, and is based on the definition of the merit function

$$merit(x) := L(x) \times Radius(x), \qquad x \in \mathcal{D}. \tag{2}$$

Here x is an arbitrary point in the domain $\mathcal{D}$, $L(x)$ is an estimate of the local Lipschitz constant and $Radius(x)$ is the distance from the closest point x_j in the starting dataset D. The Lipschitz constant is estimated for every Voronoï cell in the domain, by considering every possible linear interpolation of f containing the center of the cell and a subset of d adjacent nodes in the tessellation. Some of the details of the algorithm can be modified according to the sensibility and the experience of the experimenter, e.g., the specific space filling algorithm, or the number and the percentage of selected candidates. The percentage of selected points is related to the space filling character of the algorithm, which prevents the concentration in singular zones which could lead to missing some important features not captured in the starting sampling. The impact of the Lipschitz constant could also be controlled by adding an upper threshold.

As a space filling algorithm we chose in the benchmarks a sequential maximin distance strategy, consisting in taking sequentially the center of the largest empty ball contained in the domain.

5.2. *Computational cost*

The computational cost of the method is dominated by the Delaunay triangulation, which gives the sequence of candidate points. As reported in [18], the Delaunay triangulation of n points in dimension d costs $O\left(n^{floor(\frac{d}{2})}\right)$. Further issues related to the curse of dimensionality are discussed in the conclusions.

6. Benchmarks

We analyze the performances of Lipschitz sampling over three test functions in two dimensions: a sum of two bump functions, the Branin function and an Heaviside like step function.

DOI: 10.1685/2010CAIM545

Algorithm 5.1 Lipschitz sampling

Given f unknown in the domain $\mathcal{D} \subseteq \mathbb{R}^d$ and a starting set of points $D = \{x_1, \ldots, x_n\} \subseteq \mathcal{D}$
evaluate of f at the points $x_1, \ldots, x_n$ in D
repeat
  build the Delaunay/Voronoï tessellation corresponding to $x_1, \ldots, x_n$
  for all node $x_j \in D$ **do**
    compute all possible linear interpolations of f on the set of points composed by x_j and its adjacent nodes in the Delaunay triangulation. Set L_j as the maximum Lipschitz constant among the linear interpolations. Associate the constant to the Voronoï region of the node x_j
  end for
  by means of a space filling algorithm, define a set of $2m$ new candidate points in $\mathcal{D}$
  for all candidate points x **do**
    compute the merit function $merit(x) := L(x) \times Radius(x)$, where $L(x)$ is the Lipschitz constant of the Voronoï region to which x belongs, while $Radius(x)$ is the distance from the nearest node, i.e., the center of the region
  end for
  pick the m candidates with the largest merit function and evaluate f on them. Include the new points and their function values in the dataset and discard the remaining candidates
until the desired size of the DoE is reached

Experiments are performed as follows. Comparisons are made between a random DoE, an incremental maximin space filler DoE and Lipschitz sampling. Performances are computed on a validation dataset consisting in a 20×20 regular grid over the experimental region. The performance is the standardized mean squared error (SMSE), which is the averaged mean squared error, normalized on the variance of the response in the validation dataset, and is plotted in logarithmic scale. Training datasets of increasing sizes 10, 25, 40, 55, . . . , 190, 205, 220 are successively built for the space filler and for Lipschitz. A Gaussian process metamodel [6] is fitted to the dataset and the SMSE is evaluated on the validation dataset. The performance of the Random DoEs is averaged over 15 replications.

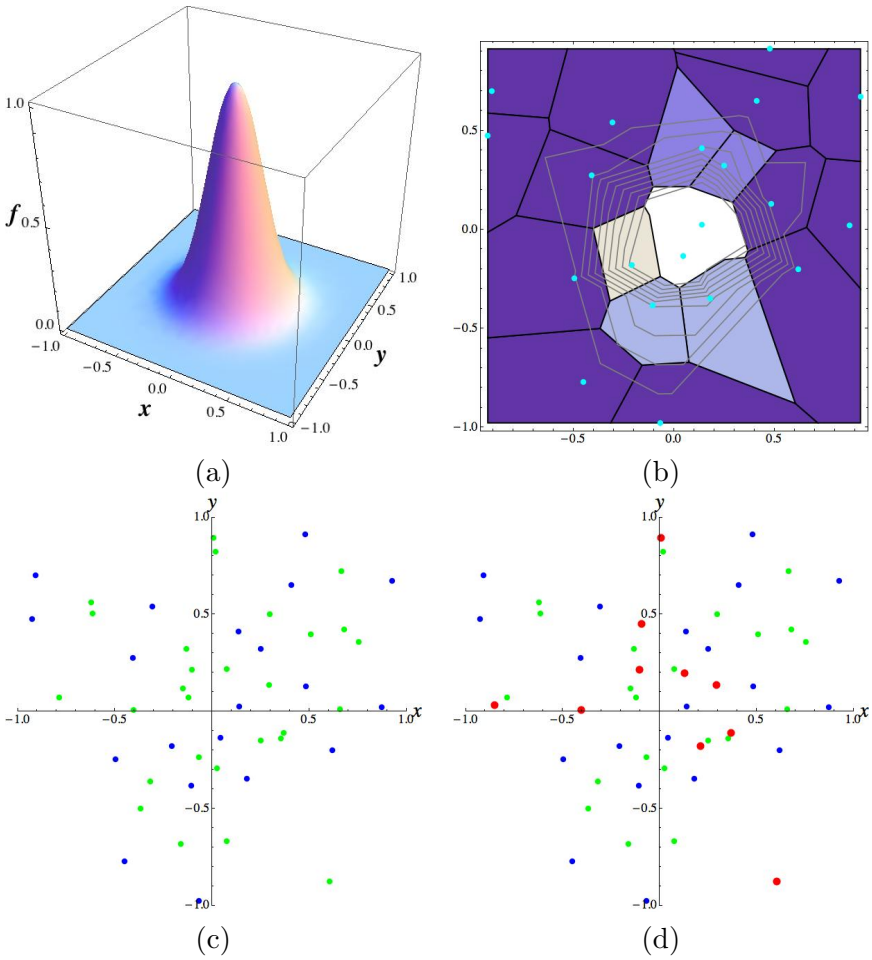

Fig. 6. Lipschitz sampling. The function (a) is evaluated on a starting arbitrary sample at first (greend dots in (b)) and the domain is tessellated in Voronoï cells accordingly. Next a set of candidate points (green dots in (c)) is built according to a spacefilling criterion (here a sequential maximin strategy). A Lipschitz complexity indicator is computed for each candidate multiplying the Lipschitz constant of the region to which it belongs by the distance from the nearest node of the tessellation. Finally the candidates with the highest values for the indicator are selected (red dots in (d)).

6.1. *Doubly bumped function*

This example consists in the sum of two similar bump functions:

(3)
$$f(x_1, x_2) := \exp\left(-5\frac{x_1^2 + x_2^2}{2}\right) + 2\exp\left(-100\frac{(x_1 + 0.6)^2 + (x_2 - 0.6)^2}{2}\right),$$

DOI: 10.1685/2010CAIM545

This function is used for testing the capabilities of an optimization algorithm to find the global maximum, or for studying robust optimization issues, i.e., estimating when a nominal optimal configuration retains its optimality under perturbation. According to intuition, Lipschitz sampling (red dots) concentrates around the sharp peak (see figure 7).

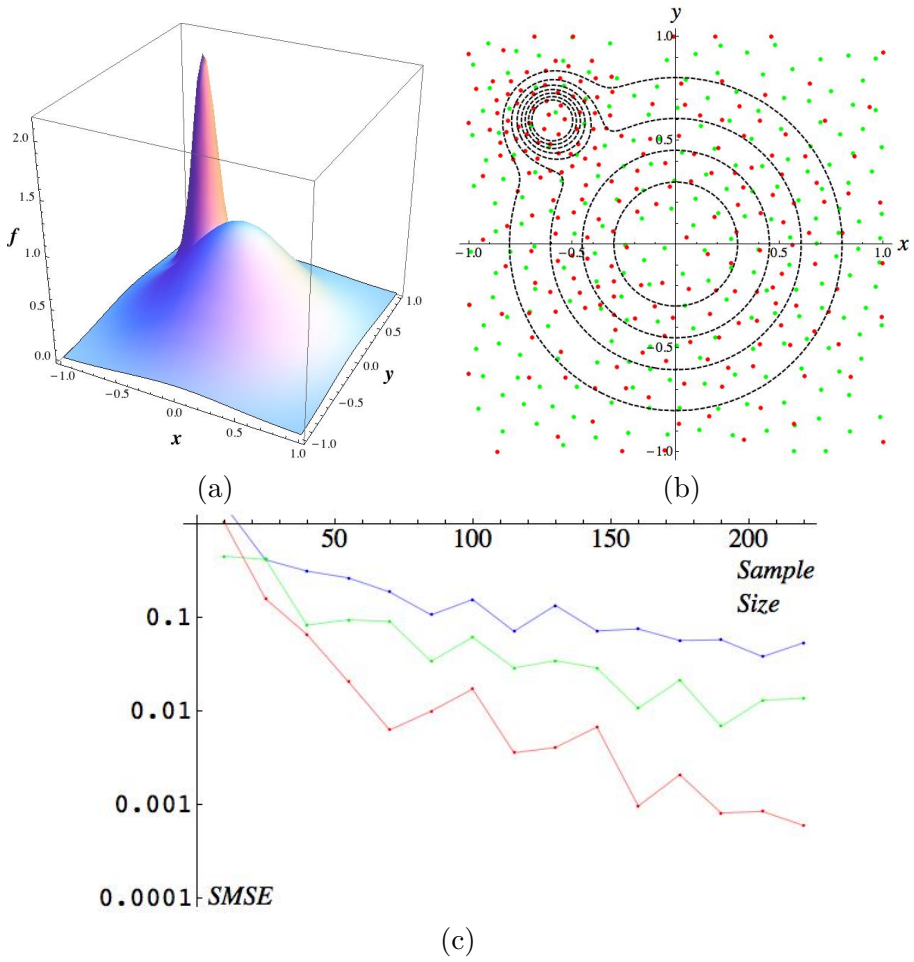

Fig. 7. Benchmarks on the double bump function (3). Performances are evaluated by computing the standardized mean squared error on a regular grid of 20×20 validation points (panel (c)). The function is approximated by fitting a gaussian process metamodel on a training sample of increasing size. Comparisons are made among the average of 15 replication of a random DoE (blue line in panel (c)), an incremental maximin distance DoE (green dots in panel (b), green line in (c)) and Lipschitz sampling (red dots in (b) and red line in (c)).

6.2. *Branin function*

The Branin function (see figure 8) is often used for assessing the performance of global optimization algorithms.

$$(4)\quad f(x_1,x_2) := \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos(x_1) + 10.$$

where $(x_1, x_2) \in [-5, 10] \times [0, 15]$.

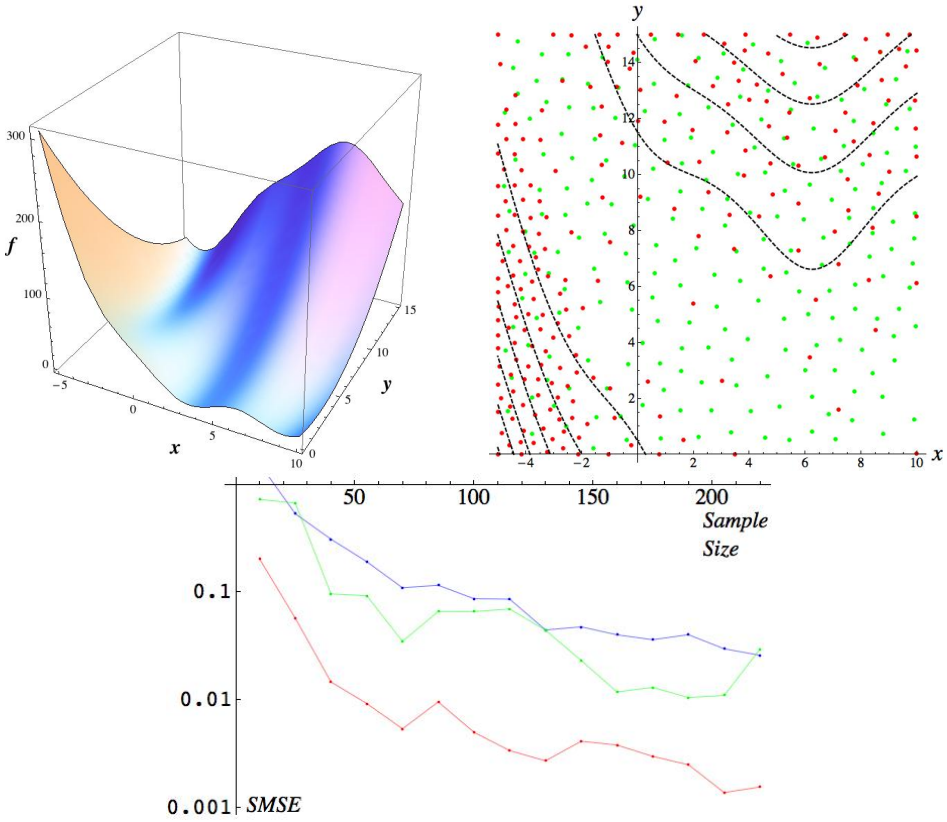

Fig. 8. Benchmarks on the Branin function (4). Performances are evaluated by computing the standardized mean squared error on a regular grid of 20×20 validation points (panel (c)). The function is approximated by fitting a gaussian process metamodel on a training sample of increasing size. Comparisons are made among the average of 15 replication of a random DoE (blue line in panel (c)), an incremental maximin distance DoE (green dots in panel (b), green line in (c)) and Lipschitz sampling (red dots in (b) and red line in (c)).

DOI: 10.1685/2010CAIM545

6.3. *Heaviside like step function*

Let

$$h(x) := \begin{cases} 1 & \text{if } x_1 < -x_2^2, \\ -1 & \text{otherwise,} \end{cases} \tag{5}$$

(see figure 9). In panel (b) we notice that the design gets denser and denser along the singularity curve, while the remaining part of the domain is not completely abandoned, but is progressively filled, though more slowly.

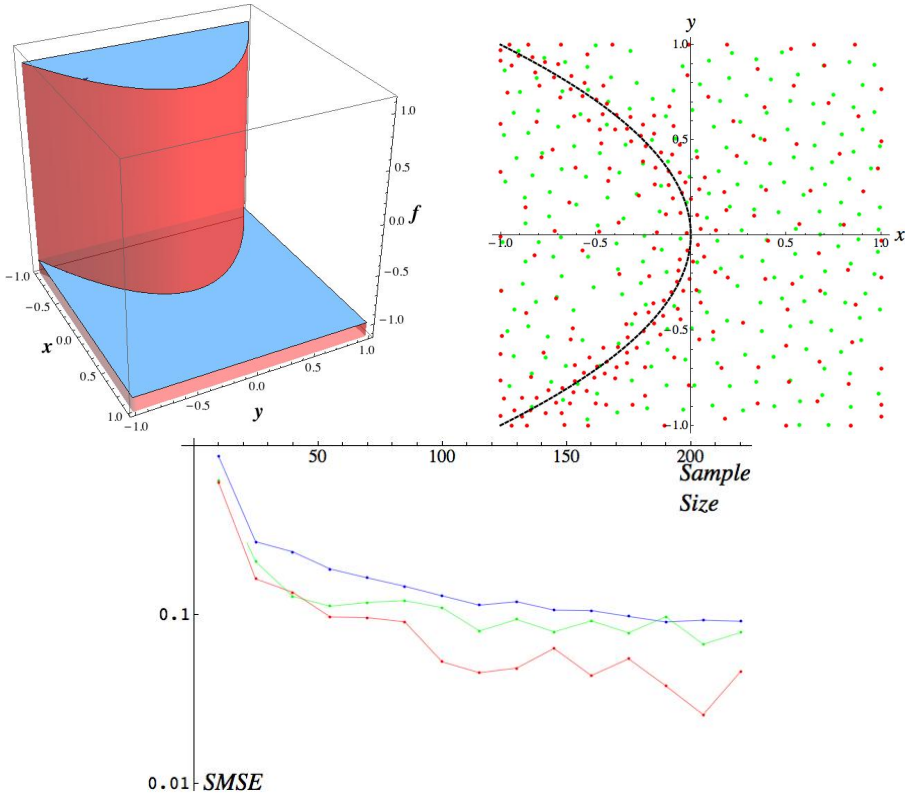

Fig. 9. Benchmarks on the Heaviside like step function (5). Performances are evaluated by computing the standardized mean squared error on a regular grid of 20×20 validation points (panel (c)). The function is approximated by fitting a gaussian process metamodel on a training sample of increasing size. Comparisons are made among the average of 15 replication of a random DoE (blue line in panel (c)), an incremental maximin distance DoE (green dots in panel (b), green line in (c)) and Lipschitz sampling (red dots in (b) and red line in (c)).

6.4. *Conclusions and future work*

As it can be observed in all of the examples examined, exploiting the information on the problem gained during the exploration process is worthwhile, even with the simple criterion proposed here. In particular, the algorithm is effective in refining regions where sudden variations take place, where for instance some important change in the behavior of the phenomenon modeled occurs, like in the case of phase transitions. We plan to complement the features of Lipschitz sampling by means of other indicators, like estimates of the curvature. Another crucial issue seems the curse of dimensionality. Indeed we tested our strategy in small or medium sized problems. In large dimension, $\geqslant 10$, it is very problematic to fill the space satisfactorily with moderately sized samples: the sequential maximin distance space filler, e.g., places the largest part of the points on the boundary of the domain. The Maximin LHD can overcome this problem, but it remains difficult to augment the size of the sample: it is easy for instance to double the size, but other choices are problematic. We think that in large dimensional cases it should be necessary to employ techniques of sensitivity analysis for screening the input variables [11,12]. Indeed usually the sparsity of effects occurs, i.e., there is a marked hierarchy among the importance of the input variables on the response, and therefore it is wasteful trying to fill in all directions indifferently.

We finally mention the problem of modeling several responses at the same time. This is a quite typical issue in industrial design, where approaches focussing on a single performance indicator could be misleading or even dangerous. When the restraint system of a car is designed, e.g., there are more than ten different injury indicators to be considered, and trying to improve the system behavior with respect to one of them usually deteriorates some of the others.

For the problem we are considering, i.e., improving globally the accuracy of metamodels for several responses, we define a compound complexity indicator involving all the Lipschitz constants of the different responses. We normalize the individual merit functions and take as an overall merit function the max of the normalized merit functions:

$$(6) \qquad merit(x_c) = \max_{j=1,\ldots,nOutput} \frac{L_i(x_c)}{\langle L_i \rangle} \cdot Radius(x_c), \qquad \forall x_c \text{ candidate.}$$

This strategy should perform better than simply considering in turn a response at a time and adding new points accordingly. If one or more responses shows a nontrivial correlation, improving one of them improves also the merit functions of the correlated ones, so during the successive steps the efforts of the algorithms can concentrate on the remaning responses.

DOI: 10.1685/2010CAIM545

The normalization presents also some non trivial features. One could normalize dividing by the global Lipschitz function computed on the domain, or by the average of the Lipschitz constants over the adjacent cells, or over an hypervolume of prescribed size. The choice of normalizing by the global Lipschitz constant could penalize excessively very singular responses, for instance.

In our implementation we adopted a normalization of the local Lipschitz constants by means of the average over the set of candidate points, i.e.,

$$\tilde{L}_i(x_c) := \frac{L_i(x_c)}{\langle L_i \rangle}, \qquad \text{where} \quad \langle L_i \rangle = \frac{\sum_{x_c} L_i(x_c)}{\#\{x_c\}}. \tag{7}$$

Lipschitz sampling has been included in the multiobjective optimization software modeFRONTIER® [19].

REFERENCES

1. J. Sacks, W. J. Welch, T. J. Mitchell, and H. P. Wynn, Design and analysis of computer experiments, *Statistical Science*, vol. 4, pp. 409–423, May 1989.
2. T. J. Santner, B. J. Williams, and W. I. Notz, *The Design and Analysis of Computer Experiments*. Springer Series in Statistics, Springer, 2003.
3. A. Lovison and E. Rigoni, Extracting optimal datasets for metamodelling and perspectives for incremental samplings, *Math. Comput. Simulation*, vol. 81, pp. 681–692, Nov 2010.
4. R. Jin, W. Chen, and A. Sudjianto, On sequential sampling for global metamodeling in engineering design, in *Proceedings of DECT'02*, (Montreal CANADA), September 29-October 2 2002.
5. D. R. Jones, A taxonomy of global optimization methods based on response surfaces, *J. Global Optim.*, vol. 21, no. 4, pp. 345–383, 2001.
6. C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. MIT Press, 2006.
7. R. J. Lempert, S. W. Popper, and S. C. Bankes, Shaping the next one hundred years: New methods for quantitative, long-term policy analysis, tech. rep., RAND, Santa Monica CA, 2003.
8. C. F. J. Wu and M. Hamada, *Experiments: Planning, Analysis and Parameter Design Optimization*. New York: John Wiley & Sons, 2000.
9. G. E. Box and N. R. Draper, *Empirical model building and response surfaces*. New York: John Wiley & Sons, 1987.
10. R. H. Myers and D. C. Montgomery, *Response Surface Methodology: Process and Product Optimization Using Designed Experiments*. New York: John Wiley & Sons, 2nd ed., 2002.

11. M. Schonlau and W. J. Welch, Screening the input variables to a computer model via analysis of variance and visualization, *Screening Methods for Experimentation in Industry, Drug Discovery and Genetics*, pp. 308–327, 2005.
12. I. M. Sobolprime, Sensitivity analysis for nonlinear mathematical models, *Mathematical Modeling & Computational Experiment*, vol. 1, pp. 407–414, 1993.
13. I. M. Sobolprime and Y. L. Levitan, A pseudo-random number generator for personal computers, *Comput. Math. Appl.*, vol. 37, no. 4-5, pp. 33–40, 1999.
14. M. D. McKay, W. J. Conover, and J. R. Beckman, A comparison of three methods for selecting values of input variables in the analysis of output from a computer code, *Technometrics*, vol. 21, no. 2, pp. 239–245, 1979.
15. K. T. Fang, D. K. J. Lin, P. Winker, and Y. Zhang, Uniform design: Theory and application, *Technometrics*, vol. 42, no. 3, pp. 237–248, 2000.
16. M. E. Johnson, L. M. Moore, and D. Ylvisaker, Minimax and maximin distance designs, *J. Statist. Plann. Inference*, vol. 26, no. 2, pp. 131–148, 1990.
17. Y. D. Sergeyev and D. E. Kvasov, Global search based on efficient diagonal partitions and a set of lipschitz constants, *SIAM J. Optim.*, vol. 16, no. 3, pp. 910–937 (electronic), 2006.
18. C. Barber, D. Dobkin, and H. Huhdanpaa, The quickhull algorithm for convex hulls, *Transactions on Mathematical Software (TOMS)*, vol. 22, Dec 1996.
19. A. Lovison, Lipschitz sampling for improving metamodels in modeFRONTIER, Tech. Rep. 2, ESTECO - EnginSoft Tecnologie per l'Ottimizzazione, 2008.