Efficient methods for optimal space filling in model reduction techniques

V. Buljak
Faculty of Mechanical Engineering, Department of Strength of Materials
University of Belgrade, Serbia

T. Garbowski
Institute of Structural Engineering
Poznan University of Technology, Poland

ABSTRACT: Model reduction techniques generate low-dimensional models to parametrize PDEs in order to allow for efficient evaluation of highly non-linear problems in many-query and real-time context. Computing time in these methods is divided into a time consuming "off-line" phase needed to "train" a surrogate model, which is further used in "on-line" phase providing accurate results in a much faster way. The accuracy of these methods is strictly connected to the number of points in which the system response is previously computed and on their distribution in a parameter space. For a given number of analysis decided to be "invested" for the design of surrogate model, the accuracy can be further improved by the adequate distribution of them in a parameter space. Two methods of robust samples distribution are proposed here. A first method is based on interactive nodes aimed to optimize the distribution of them. A second is based on an optimal Latin hypercube design. Both presented methods are flexible with respect to the number of nodes involved and they can provide uniform distribution for any arbitrary number of them. Furthermore, they allow also for taking into account different importance of divers parameters.

1 INTRODUCTION

Inverse analysis and parametric studies require a large number of time consuming analyses that differ between each other just by a few parameters which are changing from one simulation to another. For these purposes recently became popular the use of model reduction techniques aimed to design low-order models capable of evaluating responses of complex systems within drastically reduced computing time. Survey of different applications of model reduction techniques based on Proper Orthogonal Decomposition (POD) is given in (Ruckelynck et al. 2006). Similar methods based on Proper Generalized Decomposition (PGD) are developed and applied in computational fluid dynamics and in computational fracture mechanics (see e.g. Rozza et al. 2008, Nouy 2010).

Mechanical characterization of materials at present means primarily identification of the parameters entering into constitutive models implemented in computer codes oriented to non-linear inelastic structural analysis. These parameters are frequently estimated based on experiments, like instrumented indentation test, performed on site on a working components (see e.g. Buljak et al. 2013, Buljak and Maier 2012). In order to solve resulting inverse problems economically one needs to make a recourse to a certain reduced basis technique. Frequently the number of parameters to assess within the inverse analysis procedure is rather large (see e.g. Gajewski and Garbowski 2014, Garbowski et al. 2012, Garbowski et al. 2011) and the identification procedure by traditional means (i.e. FEM used for test simulation) involves significant computing time.

In general, all model reduction techniques have in common a computational strategy which consist in an a priori time consuming computation done once-for-all, which precedes a routine repeated computation that makes the use of previously generated results. By increasing the number of simulations done as a part of a preliminary "training" it is obvious that the error of surrogate model will be decreased, but for any given number of them, the optimization of error is strictly connected to their distribution. Therefore the location of experimental data points is very important for generating accurate meta-models, while maintaining a reasonable number of them.

This paper presents a revised methodologies for obtaining optimal space filling designs with minimal computational effort and having vary good space fill-

ing properties. First approach is based on so-called bubble mesh methods (see Section 2) which produce the most optimal possible design, however requires longer computational time. The second method is based on enhanced Structured Optimal Latin Hypercube design (see Section 3) which provides almost-optimal solution in real time.

2 INTERACTIVE NODES METHOD

The problem of finding the most uniform distribution of sampling points in the parameter space is addressed here by a novel method based on dynamic simulation of interactive nodes, inspired by so-called bubble mesh method for generation of finite element meshes, proposed by Shimada and Gossard (1998) and Shimada et al. (2000). The approach takes any arbitrary distribution of sampling points, attributing a unitary mass to each of them, and introducing repulsive forces which are inversely proportional to the distances between the nodes. Once the forces acting on each node are defined, a dynamic simulation is performed with explicit time integration scheme that should result in a distribution of nodes which corresponds to the equilibrated state. Since the forces are inversely proportional to the distances between the nodes, the equilibrated state of forces will correspond to the uniform distribution of the nodes (i.e. sampling points). In order to have the nodes eventually resting in equilibrated position also a damping force needs to be introduced which is taken here as linearly proportional to the velocity.

The sequence of operative stages proposed in the present context that eventually leads to the uniform distribution of sampling points in the parameter space can be outlined as follows: (i) a random distribution of M sampling points is taken as a starting distribution that covers normalized parameter space (i.e. each of N parameters is normalized to be within 0 and 1 limits); (ii) repulsive forces are calculated acting on each of M points resulting both from other points and from "walls" (i.e. boundaries) of the domain, in order to keep the points inside the domain; (iii) dynamic simulation with explicit integration scheme is performed on a set of nodes in order to find the equilibrated distribution; (iv) mapping the final distribution of nodes to cover 0 to 1 normalize space, as the equilibrated distribution will be moved away from "walls" due to the repulsive forces from them.

For any single node to be in the equilibrium it is required to be surrounded by nodes which are on equal distances from it. This is clearly satisfied for any node in the regular grid since then, along any axis there are two equidistant nodes on each side resulting in equal forces with opposite directions (see Fig. 1). In order to keep this distribution in the equilibrium, it is important not to have other forces from nodes which are farther than the neighboring four. This is achievable by assuming the setup of forces which are inversely

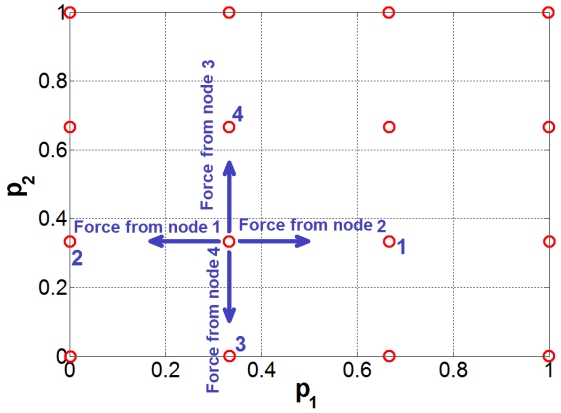

Figure 1: Set of forces acting on nodes of regular grid

proportional to the distance between the nodes but are active only if the nodes are on a shorter distance than some previously defined as critical one. This distance can be calculated for the case of regular grid, namely:

$$d_{cr} = \frac{1}{1 + \sqrt[N]{M}} \tag{1}$$

Presented scheme turns out to work reasonably well also for multidimensional cases with arbitrary number of nodes, as it will be demonstrated within the examples in Section 4.

3 OPTIMAL LATIN HYPERCUBE DESIGN METHOD

Another efficient method which can be employed for optimal space filling is a technique called Latin Hypercube Design (LHD). The LHD in N variables and in M points is constructed as follows. Each of the N design variables is divided into M equally spaced levels and only one point is allowed to occupy each level. Often Latin Hypercube design is constructed using a random procedure. Such a process results in many possible designs, each satisfying the Latin Hypercube condition of only one point per every level. The total number of distinct designs can be computed by the formula:

$$N_D = (M!)^{N-1} \tag{2}$$

where: N_D is a number of possible designs; M is a number of points; and N is a number of design variables (LHD dimensionality). It can be easily noticed that for very simple design as e.g. 16×2 there is approximately $2 \cdot 10^{13}$ admissible distributions of points. Obviously, random procedure does not prevent the possibility of creating a design, where all the points are located e.g. along the diagonal of the design space, thus resulting in poor statistical qualities of the experimental design. To overcome this problem, Optimal Latin Hypercube was introduced by McKay et al. (1979) and Iman and Conover (1980) to improve the space filling property of the design. In Optimal Latin Hypercube the points are being "pushed away" from each other as much as possible. This

last condition proved to be quite an obstacle for real-time creation of the Optimal Latin Hypercube designs. This problem is often solved using evolutionary algorithms, such as Genetic Algorithms (see e.g. Bates et al. (2004)) or Enhanced Stochastic Evolutionary Algorithm (ESEA) proposed by Jinb et al. (2005). ESEA is an enhanced version of the Stochastic Evolutionary Algorithm, developed by Saab and Rao (1991). The ESEA is algorithm proved to be relatively fast in generating the Optimal Latin Hypercube designs, unlike the Genetic Algorithm implementation. Such efficiency is explained by the use of the element-exchange algorithm and efficient evaluation of the optimality criterion:

$$\phi_p = \left[\sum_{i=1}^{M-1} \sum_{j=i+1}^{M} d_{ij}^{-p} \right]^{1/p} \tag{3}$$

where p is a positive integer, M is the number of points in the design, and d_{ij} is the inter-point distance between all point pairs in the design. The general inter-point distance between any point pair $\mathbf{x}_i$ and $\mathbf{x}_j$ can be expressed as follows:

$$d_{ij} = d\left(\mathbf{x}_i, \mathbf{x}_j\right) = \left[\sum_{k=1}^{N} \left| x_{ik} - x_{jk} \right|^t \right]^{1/t} \tag{4}$$

where N is a number of design variables. Here $p = 50$ and $t = 1$ are used. Minimizing the value of performance measure ϕ_p leads to the maximization of point-to-point distance in the design.

3.1 *Structured LHD*

Another empirical approach to create a well structured design (rather than based on random number generation) that is reasonably close to Optimal Latin Hypercube design without performing optimization was proposed by Viana et al. (2010). The importance of such approach resides in the fact that it gives the capability to create a Latin Hypercube design that has better space filling properties than the standard Latin Hypercube design, using minimum computational time. The resulting design can be used either as an initial design for the Optimal Latin Hypercube generator algorithm (such as Genetic Algorithm or ESEA) or as a good approximate Optimal Latin Hypercube design. In this case, one should take into account a compromise between obtaining best possible Optimal Latin Hypercube design and the computational cost to generate such design.

The approach is quite simple and it is based on the assumption that the simple N-dimensional Latin Hypercube can be constructed from a N-dimensional seed design. The procedure can be divided basically into two steps: (a) first, a small LHS is constructed to be used as a seed in the process. Figure 2 shows some examples of 2-dimensional seed designs. (b) second, the design space is divided into blocks, in such a way that each dimension is divided in the same number of blocks. The result is that each block can be filled using the seed design (defined previously) but enhanced by additional levels (see Fig. 3). The seed design must be done a priori and later properly placed into each of the blocks. The biggest advantage of this approach is that there are no calculations to perform. All operations can be viewed as translations of an N-points block in the N-dimensional hypercube.

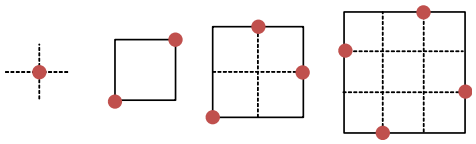

Figure 2: Seed design: examples for 2 design variables

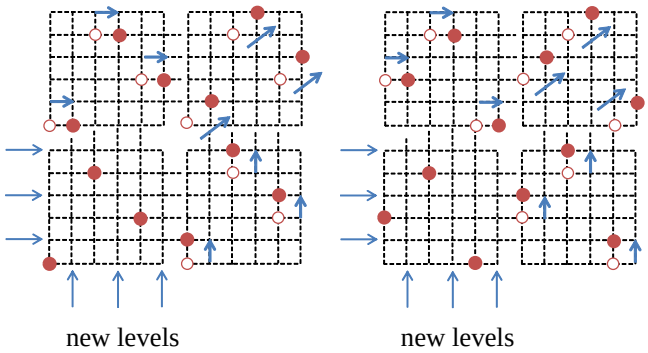

Figure 3: 12×2 SLHD: two examples of 3-node seed in 2 design variables

3.2 *Seed design*

The presented methodology for optimal space filling has, however, one important disadvantage: it is not clear how to select a priori the best seed design to get overall optimal or close to optimal Structured Latin Hypercube Design. One possible choice would be to select the seed design which has the smallest ϕ_p value, however, this may lead to non-optimal global solution once the full design is assembled from the blocks of seeds. Table 1 shows all 24 possible seed design consisting of 4 nodes in 2 dimensional parameter space and corresponding ϕ_p values for single seed design (4×2) as well as for 16×2 and 96×2 SLHD. It is clear that the best seed design is not always resulting in the best SLHD, in contrary it may provide rather poor quality solution.

Moving to higher dimensional spaces the number of possible seed designs is growing according to formula (2), where total number of points M is now substituted with the number of nodes in a seed design S, so the formula reads:

$$N_{SD} = (S!)^{N-1} \tag{5}$$

where S is much smaller than M.

Figure 4 shows all possible 3-node seed design in 2D space and an example of simple transition of a

Table 1: All possible 4-node seed designs in 2D space and corresponding performance of 4×2, 16×2 and 96×2 LHD

seed design	$\phi_{p(4\times2)}$	$\phi_{p(16\times2)}$	$\phi_{p(96\times2)}$
	2.16845	5.57352	14.8611
	2.16845	10.6066	18.1251
	2.15093	5.52851	14.8113
	2.15093	6.89680	16.4113
	2.15093	5.52851	14.7326
	2.15093	5.52851	14.8096
	2.15093	5.52851	14.8096
	2.15093	5.52851	14.8159
	2.15093	6.89680	16.3114
	2.15093	5.52851	14.8238
	2.15093	5.52851	14.8238
	2.15093	6.89680	16.2019
	2.12132	**5.45240**	**14.5541**
	2.12132	10.6066	18.1251
	2.12132	5.45250	14.6315
	2.12132	5.45250	14.6315
	2.12132	6.80185	15.9881
	2.12132	6.80185	16.0868
	2.12132	6.80185	16.0868
	2.12132	5.45250	14.6074
	2.12132	5.45250	14.6074
	2.12132	6.80185	15.9692
	**1.37936**	6.80185	16.0843
	**1.37936**	6.80185	16.0843

selected solution to 3D space. In the case of 3-node seed in 2D space, the six possible designs transform to thirty-six designs if we move to 3D space. Table 2 presents all possible solutions of 3-node seed design in 3D space and corresponding values of ϕ_p of 3×3 or 96×3 SLHD. Again it is visible that not all the best seed design (bolded values in Tab. 2) result in good SLHD.

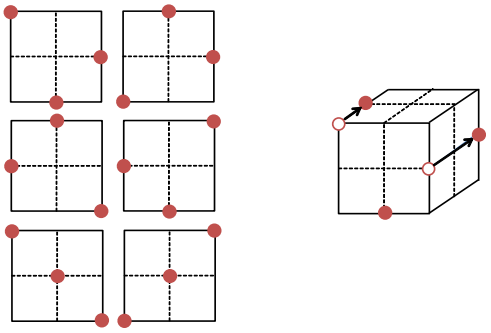

Figure 4: (a) all possible 3-node seed designs in 2D space; (b) transition from 3×2 seed to 3×3

Here presented examples provide an important knowledge about performance of selected seed designs and their influence on Structural Latin Hypercube Design. It is clear that for the small designs a systematic search within the admissible seed designs can be used to find the most optimal SLHD, however the number of solution grows exponentially with the space dimension. So for e.g. 4-node seed in 12-dimensional space the number of designs equals to $n_d = (4!)^{11} = 24^{11} \approx 1.5 \cdot 10^{15}$. Therefore for high-dimensional spaces the optimization by ESEA or GA should be performed or alternatively the smaller seed design should be selected.

4 EXAMPLES

In order to evaluate the performance of different space filling algorithms, three numerical examples on different scales are treated in what follows.

First exercise considers a problem of distribution of 90 samples in 2D space. This example illustrates the capability of space filling methods to provide rather uniform distributions for the problems in smaller scale (i.e. 2D in this case) when the number of nodes is not equal to the one required for forming a regular grid. Four different distributions are compared. The first one (Figure 5a) considers purely random distribution. The second one (Figure 5b) is the distribution which satisfies Latin Hypercube criterion, generated by random permutation procedure. The third distribution (Figure 5c) is the one resulting from Interactive Nodes Method described in Section 2. Fourth distribution (Figure 5d) is generated by the Structured Latin Hypercube Design procedure described in Section 3. All four resulting distributions are quantitatively compared by calculating values of ϕ_p (equation (3) with $p = 50$ and $t = 1$). Resulting values are

Table 2: All possible 3-node seed designs in 3D space and corresponding performance of 3×3, 96×3 LHD

point 1	point 2	point 3	$\phi_{p(3\times3)}$	$\phi_{p(96\times3)}$
$[1,1,1]$	$[2,2,2]$	$[3,3,3]$	0.5854	0.0465
$[3,1,1]$	$[2,2,2]$	$[1,3,3]$	0.5854	0.0667
$[2,1,1]$	$[3,2,2]$	$[1,3,3]$	0.5774	0.0381
$[3,1,1]$	$[1,2,2]$	$[2,3,3]$	0.5774	0.0379
$[1,1,1]$	$[3,2,2]$	$[2,3,3]$	0.5774	0.0467
$[2,1,1]$	$[1,2,2]$	$[3,3,3]$	0.5774	0.0467
$[1,3,1]$	$[2,2,2]$	$[3,1,3]$	0.5854	0.0850
$[3,3,1]$	$[2,2,2]$	$[1,1,3]$	0.5854	0.1379
$[2,3,1]$	$[3,2,2]$	$[1,1,3]$	0.5774	0.0449
$[3,3,1]$	$[1,2,2]$	$[2,1,3]$	0.5774	0.0449
$[1,3,1]$	$[3,2,2]$	$[2,1,3]$	0.5774	0.0594
$[2,3,1]$	$[1,2,2]$	$[3,1,3]$	0.5774	0.0594
$[1,2,1]$	$[2,3,2]$	$[3,1,3]$	0.5774	0.0407
$[3,2,1]$	$[2,3,2]$	$[1,1,3]$	0.5774	0.0436
$[2,2,1]$	$[3,3,2]$	$[1,1,3]$	0.5774	0.0438
$[3,2,1]$	$[1,3,2]$	$[2,1,3]$	**0.4173**	**0.0358**
$[1,2,1]$	$[3,3,2]$	$[2,1,3]$	**0.4173**	0.0380
$[2,2,1]$	$[1,3,2]$	$[3,1,3]$	0.5774	0.0424
$[1,3,1]$	$[2,1,2]$	$[3,2,3]$	0.5774	0.0405
$[3,3,1]$	$[2,1,2]$	$[1,2,3]$	0.5774	0.0437
$[2,3,1]$	$[3,1,2]$	$[1,2,3]$	**0.4173**	0.0360
$[3,3,1]$	$[1,1,2]$	$[2,2,3]$	0.5774	0.0440
$[1,3,1]$	$[3,1,2]$	$[2,2,3]$	0.5774	0.0425
$[2,3,1]$	$[1,1,2]$	$[3,2,3]$	**0.4173**	0.0380
$[1,1,1]$	$[2,3,2]$	$[3,2,3]$	0.5774	0.0498
$[3,1,1]$	$[2,3,2]$	$[1,2,3]$	0.5774	0.0565
$[2,1,1]$	$[3,3,2]$	$[1,2,3]$	**0.4173**	0.0408
$[3,1,1]$	$[1,3,2]$	$[2,2,3]$	0.5774	0.0540
$[1,1,1]$	$[3,3,2]$	$[2,2,3]$	0.5774	0.0539
$[2,1,1]$	$[1,3,2]$	$[3,2,3]$	**0.4173**	0.0444
$[1,2,1]$	$[2,1,2]$	$[3,3,3]$	0.5774	0.0499
$[3,2,1]$	$[2,1,2]$	$[1,3,3]$	0.5774	0.0565
$[2,2,1]$	$[3,1,2]$	$[1,3,3]$	0.5774	0.0540
$[3,2,1]$	$[1,1,2]$	$[2,3,3]$	**0.4173**	0.0408
$[1,2,1]$	$[3,1,2]$	$[2,3,3]$	**0.4173**	0.0444
$[2,2,1]$	$[1,1,2]$	$[3,3,3]$	0.5774	0.0539

listed in Table 3. Clearly, the worst distribution (i.e. the one with largest value of ϕ_p) is random distribution since this one is not offering any control over uniformity. Distribution that satisfies LHD criterion covers the space more uniformly than random one, and in terms of uniformity is somewhere in between the results obtained by the last two methods and a pure random distribution. The value of ϕ_p confirms what can be observed from Figure 5, namely that the last two distributions are much better, while IntNod one is slightly outperforming a SLHD one. Finally, the last column in Table 3 gives the ϕ_p value for the regular 10×9 grid.

Second exercise considers larger scale problem, namely the distribution of 2000 sampling points in 8D space. For this case there is rather large difference in required number of points to form regular grid of 2 points per axis and 3 points per axis, namely,

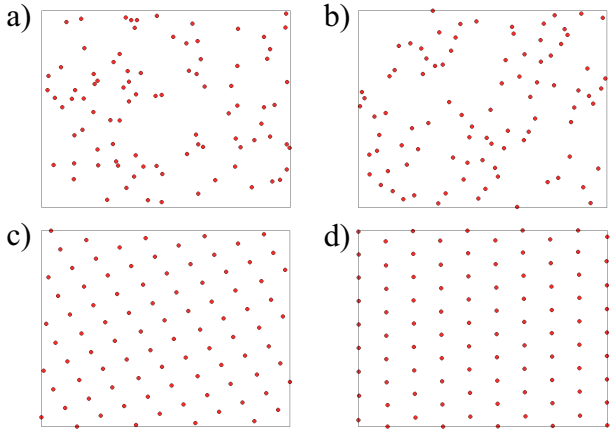

Figure 5: Distribution of 90 points in 2-dimensional space by: (a) random uniform generator, (b) Latin Hypercube Design, (c) interactive nodes algorithm, (d) Structured Latin Hypercube Design.

Table 3: The performance measurement of 5 methods of points distribution in 2-dimensional parameter space

RAND	LHD	IntNod	SLHD	regular grid
48.56	31.27	9.37	10.83	9.83

$2^8 = 256$ and $3^8 = 6561$. The example illustrates that quite uniform distribution is obtained with any arbitrary number of points and therefore the distribution can be improved by using smaller number of sampling points than the one required to "jump" from one regular grid to another. The problem is solved by using both IntNod method and SLHD and for the resulting distributions ϕ_p value is calculated. For this case the corresponding values of ϕ_p were equal to 1.88 and 2.66 for IntNod and SLHD, respectively.

The last example is of the largest scale: the distribution of 8000 sampling points in 14D space. This example intends to show the full potential of space filling methods where forming of regular grid involves fairly large number of sampling points and therefore is practically not manageable. In order to evaluate the uniformity of the resulting distribution a following criterion is introduced. To each node from the final distribution it is attributed the distance to the closest neighbor in normalize parameter space. A graph is then plotted with the identity of the node put on abscissa and the above mentioned distance to the ordinate. Should the distribution be uniform, line on the graph would be almost flat as each point will have at the same distance from it another sampling point. Figure 6 shows this graph for starting (Figure 6a) and final (Figure 6b) distribution computed by IntNod method, and the distribution computed by SLHD (Figure 6c). Practically the same flat line is obtained for both distributions computed by IntNod and SLHD methods evidencing the uniformity of the resulting distribution.

Tables 4 and 5 present the values of ϕ_p criterion for different designs, namely: random, LHS, OLHS with 1000 random permutation iterations, SLHD for different seed design and IntNod. It is clear from the results

that in case of high-dimensional parameter's spaces the IntNode has the best space filling properties.

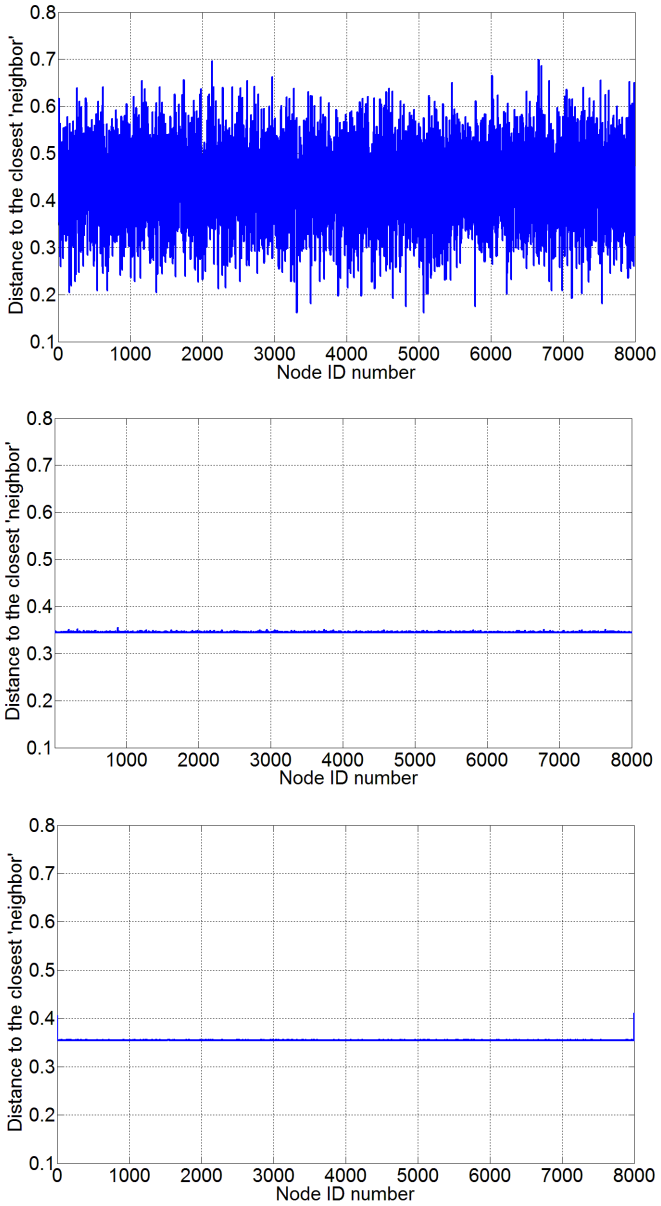

Figure 6: Distance between the closest neighbors; starting distribution (top figure), final distribution computed by IntNode (middle figure) and final distribution computed by SLHD (bottom figure)

5 CONCLUSIONS

The study presented in what precedes has shown the importance of the use of uniform space filling methods in combination with reduced basis (RB) techniques. Both of the tested methods turned out to produce uniformly distributed samples in computing times that are insignificant compared to the time required for the training of reduced basis model.

In general, the error of RB models is connected to the number of sampling points used in the training phase. Enlarging this number however, extends the time employed for the training. Results presented in this paper showed that for the same cost of the training in terms of computing time involved, by using space filling methods a further reduction of the error of RB model can be achieved.

Table 4: performance of different designs (2000 points in 8-dimensional space)

design	ϕ_p
regular ($2^8 = 256$)	1.1487
regular ($3^8 = 6561$)	2.4655
interactive nodes	1.8810
SLHD (3×8)	2.6618
SLHD (2×8)	2.9650
SLHD (1×8)	2.7029
OLHD (10^3 iter.)	5.3854
LHD	6.6741
random	7.9221

Table 5: performance of different designs (8000 points in 14-dimensional space)

design	ϕ_p
interactive nodes	1.1097
SLHD (3×14)	2.5219
SLHD (2×14)	1.5379
SLHD (1×14)	1.8348
OLHD (10^3 iter.)	2.6946
LHD	2.8439
random	3.4266

For small scale problems these methods are offering better control over the error and possibility to use virtually any number of sampling points, and not only those enough to form a regular grid. The importance of space filling methods is even more evident in large scale problems, where regular grids practically cannot be formed, as the number of sampling points required for it grows exponentially with the number of parameters. For such cases space filling methods are the only alternative to the random distribution of sampling points, and therefore the only possibility to construct RB model with controllable error.

The error of RB model depends also on problem at hand, and therefore it may occur that not the same density of the distribution is required in all of the regions of the parameter space. Research in progress concerns the extension of the presented approach to take into account the local error as a criterion for the distribution of the sampling points in the parameter space.

REFERENCES

Bates, S., J. Sienz, & V. Toropov (2004). Formulation of the optimal latin hypercube design of experiments using a permutation genetic algorithm. In *45th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference*.

Buljak, V., G. Cocchetti, & G. Maier (2013). Calibration of brittle fracture models by sharp indenters and inverse analysis. *International Journal of Fracture*. DOI: 10.1007/s10704-013-9814-4.

Buljak, V. & G. Maier (2012). Identification of residual stresses by instrumented elliptical indentation and inverse analysis.

Mechanics Research Communications 41, 21–29.

Gajewski, T. & T. Garbowski (2014). Calibration of concrete parameters based on digital image correlation and inverse analysis. *Archives of Civil and Mechanical Engineering*. http://dx.doi.org/10.1016/j.acme.2013.05.012.

Garbowski, T., G. Maier, & G. Novati (2011). Diagnosis of concrete dams by flat-jack tests and inverse analyses based on proper orthogonal decomposition. *Journal of Mechanics of Materials and Structures 6*(1-4), 181–202.

Garbowski, T., G. Maier, & G. Novati (2012). On calibration of orthotropic elastic-plastic constitutive models for paper foils by biaxial tests and inverse analyses. *Structural and Multidisciplinary Optimization 46*, 111–128.

Iman, R. & W. Conover (1980). Small sample sensitivity analysis techniques for computer models, with an application to risk assessment. *Communications in Statistics - Theory and Methods 17*, 1749–1842.

Jinb, R., W. Chena, & A. Sudjianto (2005). An efficient algorithm for constructing optimal design of computer experiments. *Journal of Statistical Planning and Inference 134(1)*, 268–287.

McKay, M., R. Beckman, & W. Conover (1979). A comparison of three methods for selecting values of input variables from a computer code. *Technometrics 21*, 239–245.

Nouy, A. (2010). A priori model reduction through proper generalized decomposition. *Computer Methods in Applied Mechanics and Engineering 199*, 1603–1626.

Rozza, G., D. Huynh, & A. Patera (2008). Reduced basis approximation and a posteriori error estimation for affinely parametrized elliptic coercive partial differential equations - application to transport and continuum mechanics. *Archives of Computational Methods in Engineering 15 (3)*, 229–275.

Ruckelynck, D., F. Chinesta, E. Cueto, & A. Ammar (2006). On the a priori model reduction: Overview and recent developments. *Archives of Computational Methods in Engineering 13(1)*, 91–128.

Saab, Y. & Y. Rao (1991). Combinatorial optimization by stochastic evolution. *IEEE Transactions on Computer-Aided Design 10*, 525–535.

Shimada, K. & D. Gossard (1998). Automatic triangular mesh generation of trimmed parametric surfaces for finite element analysis. *Computer Aided Geometric Design 15(3)*, 199–222.

Shimada, K., A. Yamada, & T. Itoh (2000). Anisotropic triangulation of parametric surfaces via close packing of ellipsoids. *International Journal of Computational Geometry and Applications 10(4)*, 301–324.

Viana, F., G. Venter, & V. Balabanov (2010). An algorithm for fast optimal latin hypercube design of experiments. *International Journal for Numerical Methods in Engineering 82*, 135–156.