7th World Congress on Structural and Multidisciplinary Optimization
COEX Seoul, 21 May – 25 May 2007, Korea

Generating Optimal Latin Hypercube Designs in Real Time

Felipe Antonio Chegury Viana [1], Vladimir Balabanov[2], Gerhard Venter [3], John Garcelon[4], Valder Steffen [5]

[1] PhD student, Federal University of Uberlândia, School of Mechanical Engineering, 38400-902 Uberlândia, MG, Brazil, chegury@mecanica.ufu.br
[2] VisualDOC Project Manager, Vanderplaats R&D, Inc., 1767 South 8th Street, Colorado Springs, CO 80906, USA, vladimir@vrand.com
[3] Associate Professor, Stellenbosch University, Department of Mechanical and Mechatronic Engineering, Private Bag X1 Matieland 7602, South Africa, gventer@sun.ac.za
[4] Senior R&D Engineer, Vanderplaats R&D, Inc., 126 Bonifacio Place, Suite F, Monterey, CA 93940, USA jhg@vrand.com
[5] Professor, Federal University of Uberlândia, School of Mechanical Engineering, 38400-902 Uberlândia, MG, Brazil, vsteffen@mecanica.ufu.br

1. Abstract

This paper presents a new methodology for creating good approximations to Optimal Latin Hypercube designs without using formal optimization. Using this methodology, the approximations to Optimal Latin Hypercube designs are obtained with minimal computational effort and in real time. The methodology exploits patterns of point locations for Optimal Latin Hypercube designs. Small building blocks with one or several points in each are used to recreate these patterns taking into account the dimensionality of the problem and the required number of design points.

2. Keywords: Optimal Latin Hypercube design, design of experiments, metamodels

3. Introduction

The location of experimental data points is very important for generating accurate metamodels, while maintaining a reasonable number of experimental data points. Design of Experiments [1] provides a tool that aids in the process of point selection. One popular design of experiments technique is the Optimal Latin Hypercube design [2-4]. The Optimal Latin Hypercube design is widely used for computer generated experiments. This design is model independent and has very good space filling properties. Optimal Latin Hypercube design is also flexible: it gives the user the option to select as many points as desired. Unfortunately, generating an Optimal Latin Hypercube design results in a difficult optimization problem that is traditionally solved by time consuming non-gradient based methods, such as Genetic Algorithms [2]. Solution times reported in the literature often exceed several hours for large number of points and large number of design variables [2-4]. Such a high computational cost limits the practical use of this important design of experiments.

This paper presents a new methodology for creating good approximations to Optimal Latin Hypercube designs without using formal optimization. Using this methodology, the approximations to Optimal Latin Hypercube designs are obtained with minimal computational effort and in real time. The methodology exploits patterns of point locations for Optimal Latin Hypercube designs. Small building blocks with one or several points in each are used to recreate these patterns taking into account the dimensionality of the problem and the required number of design points.

In its current form, the obtained Structured Latin Hypercube designs provided by this new methodology in two dimensional space cannot be improved using non-gradient optimization methods. In higher dimensional design spaces, the Structured Latin Hypercube designs generated using the proposed methodology are still good but can be improved using non-gradient optimization methods, which often adds significant computational cost. Even though the designs can be improved for higher dimensional problems, the proposed methodology provides a powerful and efficient tool for obtaining good approximations to Optimal Latin Hypercube designs in real time.

4. Latin Hypercube Design and Enhanced Stochastic Evolutionary Algorithm

The Latin Hypercube design 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 designs are constructed using a random process. Such a process results in many possible designs, each satisfying the Latin Hypercube condition of only one point per every level. However, the random process does not prevent the

possibility of creating a design, where all the points are located 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 to improve the space filling property of the design. In Optimal Latin Hypercube the points are pushed away from each other as much as possible. This last condition is 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 [2].
To introduce a valid new method for creating Optimal Latin Hypercube design, the new results should be compared to existing methods. We chose to compare our results to the ones obtained using Enhanced Stochastic Evolutionary Algorithm (ESEA). ESEA was introduced in [5] as an enhanced version of the Stochastic Evolutionary Algorithm, developed by Saab and Rao [6]. We chose ESEA because this algorithm proved to be relatively fast in coming up with 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.

It should be mentioned that ESEA uses the ϕ_p criterion to come up with the optimal design:

$$\phi_p = \left[\sum_{i=1}^{s} J_i d_i^{-p} \right]^{1/p} \quad , \tag{1}$$

where

p - is a positive integer (when using a large value of p, the ϕ_p criterion is equivalent to the maximum distance criterion [5]). We used p=50 for our calculations.

d_i - are the values of the distances between the points.

J_i - is the number of pairs of points in the design separated by d_i

s - is the number of distinct distance values.

The list of distances between the points ($d_1, d_2, \ldots, d_s$) is composed of the distinct distances between individual points, d_{ij} :

$$d_{ij} = \sum_{k=1}^{N} \left| x_{ik} - x_{jk} \right| \quad , \tag{2}$$

where

N - is the number of design variables (coordinates)

x_{ik} - is the k-th coordinate of the i-th point

5. Structured Latin Hypercube Design

This section describes an 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. The importance of such an 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 (at most, seconds). 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 approximation for Optimal Latin Hypercube design. In the former 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 methodology that we propose is based on the concept that an Optimal *N*-dimensional Latin Hypercube design can be constructed from an *N*-dimensional seed design. Instead of a formal description of the approach, a practical example will be used to explain the methodology of creating a Structured Latin Hypercube design.

Consider the case in which it is desired to construct an Optimal 16 × 2 Latin Hypercube design, i.e., 16 points in 2-dimensions.
First, a small Latin Hypercube design will be constructed to be used as a seed in the process. Figure 1 shows some examples of 2-dimensional seed designs. Figure 1(a) shows the seed used in this example. It is important to note that this seed can be as simple as just a 1 × *N* design (where *N* is the dimension of the problem, i.e. the number of design variables and 1 is the number of points used).

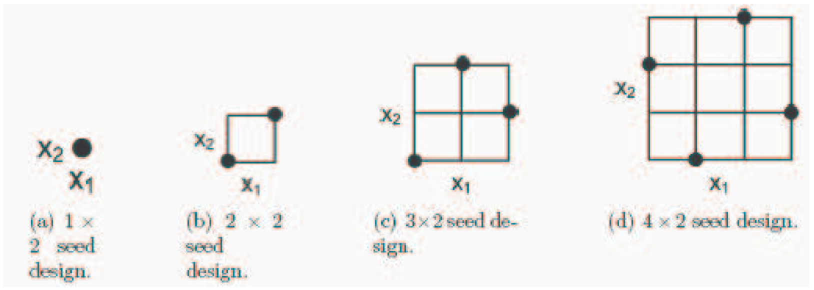

Figure 1. Examples of seed designs for 2 design variables

Second, the design space is divided into blocks in such a way that each dimension is divided into the same number of blocks. The result is that each block can be filled using the seed design (defined previously). It is clear that these processes are inter-dependent. The seed size, i.e., the number of points in the seed design, and the final design size will determine the number of blocks in each dimension. In general, the following relations must be observed:

$$LatinHypercubeSize = NumberOfBlocks \times \mathrm{SeedSize} \quad (3)$$

$$NumberOfBlocks = (NumberOfDivisions)^{NumberOfVariables} \quad (4)$$

$$SeedSize = \frac{LatinHypercubeSize}{NumberOfBlocks} \quad (5)$$

Figure 2 shows how the 16×2 Latin Hypercube mesh will be divided into blocks for the seed design from Figure 1(a). It is important to point out that the fact that each block has four rows and four columns of the Latin Hypercube mesh does not mean that each block will have four points at the end of the process. Instead, this is a way to ensure the minimal distance between points in the final Structured Latin Hypercube design

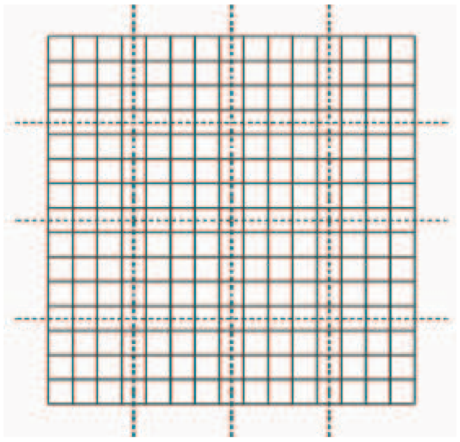

Figure 2. 16×2 Latin Hypercube mesh divided into blocks.

The seed design must be properly placed into each block. Figure 3 illustrates how this process works. The first step is to properly scale the "seed design" and then place it at the origin. Next, a set of shifts must be performed. The first one is to shift the seed to consecutive blocks along one of the dimensions. The second step is to shift the origin of the seed inside the mesh of the block. There is a coupling between these two processes. If the block shift is performed on the rows, the seed-origin shift must be performed on the columns, and vice-versa. This process is repeated until one of the dimensions is filled. After that, the whole set of points placed in that dimension can be used to feedback the "shifting" process that continues filling the next dimensions.
The biggest advantage of this approach is that there are very few calculations to perform. All operations can be viewed as translations of a *N*-points block in the *N*-dimensional hypercube.

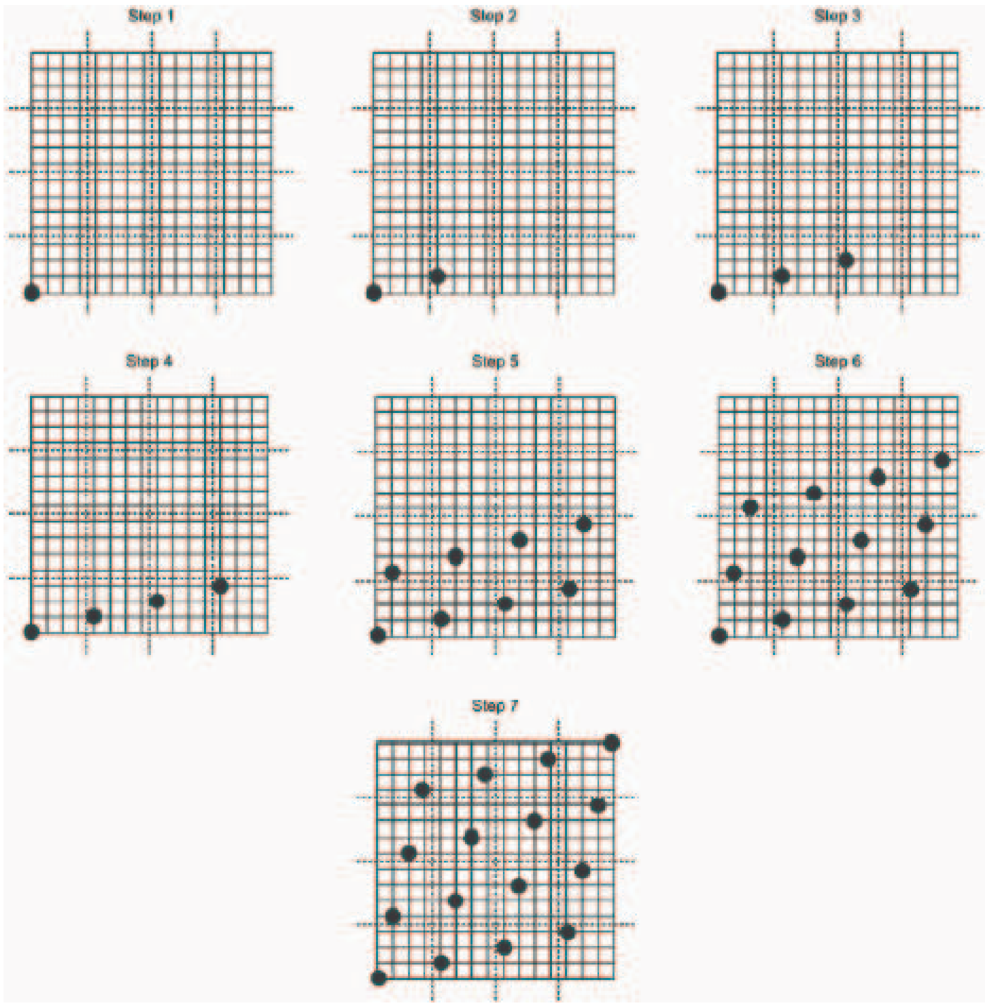

Figure 3. The process of creating the 16×2 Structured Latin Hypercube design

Although efficient for generating large designs, the previous algorithm fails to provide the flexibility for the number of points that the user may want. The approach described above is limited by the relationship

$$LatinHypercubeSize = NumberOfDivisions^{NumberOfVariables} \quad , \tag{6}$$

which follows directly from the Eqs. (4) and (5) for the smallest seed size of 1. It means that the described algorithm is restricted to have the number of points to be equal to the integer power of the number of design variables.
To overcome this serious deficiency we decided to use the fact that the algorithm is capable of generating well distributed points that fill the design space well. To generate the Structured Latin Hypercube design with any number of points, the first step is to generate the design that has at least the required number of points using the algorithm described above. If after generating the design we obtain the required number of points, the process is completed. If after applying the Structured Latin Hypercube algorithm the number of point is larger than required, a shrinking process is used to reduce the number of points. Points are removed one-by-one from the initial Structured Latin Hypercube design using certain criterion until the desired number of points is met. The simplest criterion that works well is to discard the points that are the farthest from the center of the hypercube. The reason that such criterion works is that this criterion preserves the kernel of the initially generated design formed by well distributed points.

6. Numerical Results

The appeal of the proposed methodology for generating Structured Latin Hypercube design is that virtually no computational time is required to create good Latin Hypercube designs. This empirical approach can be used either to quickly obtain a Latin Hypercube design with good space-filling properties or to generate a good starting point for a formal Optimal Latin Hypercube optimization procedure (like ESEA).
Table 1 shows the performance comparison of the ESEA procedure of generating Optimal Latin Hypercube designs starting from three different initial designs, which are as follows:

(a) "Worst case" – the worst design, where the points are located along the diagonal of the design space,
(b) "Structured case" – Structured Latin Hypercube design,
(c) "Random case" – a random design.
For all cases, the stopping criterion used was a maximum number of 100 iterations.
Table 1 also allows one to directly compare the values of the ϕ_p criterion for the Structured Latin Hypercube design with the Optimal Latin Hypercube design obtained with ESEA.

Table 1. Optimal Latin Hypercube designs generated using ESEA from three different initial designs

Design size	Quantity	Worst case	Structured case	Random case
225×2	Time [s]	143	68	147
	Iterations	251	101	237
	ϕ_p initial	0.55715	0.07052	0.51110
	ϕ_p final	0.07560	0.07052	0.07528
1024×2	Time [s]	11155	3868	7838
	Iterations	284	101	202
	ϕ_p initial	0.57433	0.03527	0.50697
	ϕ_p final	0.04218	0.03527	0.04258
256×4	Time [s]	313	372	469
	Iterations	283	336	424
	ϕ_p initial	0.27929	0.01658	0.03846
	ϕ_p final	0.01101	0.01087	0.01082
243×5	Time [s]	609	556	550
	Iterations	547	498	494
	ϕ_p initial	0.22320	0.01303	0.02668
	ϕ_p final	0.00686	0.00688	0.00679
1024×10	Time [s]	34283	27772	29421
	Iterations	601	489	518
	ϕ_p initial	0.22973	0.00440	0.01086
	ϕ_p final	0.00233	0.00234	0.00232

Note that for the cases of two design variables (225×2 and 1024×2) the optimization procedure was not able to improve the ϕ_p criterion of the Structured Latin Hypercube design. This indicates that for the 2D cases, the proposed empirical methodology of creating the Structured Latin Hypercube design produced the optimum results at almost no computation cost. For the higher dimensional cases, the formal optimization was able to make relatively small improvements to the ϕ_p criterion of the Structured Latin Hypercube design. This is an indication that the Structured Latin Hypercube design generated using the proposed methodology provides a reasonable approximation to the Optimal Hypercube designs in high dimensions.

7. Conclusions

The paper introduces an empirical methodology for creating Structured Latin Hypercube designs that are good approximations of Optimal Latin Hypercube designs. The approach is based on the idea that a simple "seed design" with a few points can be used to build a complete Latin Hypercube design. The main advantage of the proposed methodology is that it requires virtually no computational time. Test cases show that in two dimensional case (two design variables) the Structured Latin Hypercube designs obtained using the proposed methodology can not be improved using a formal ESEA optimization procedure. For higher dimensions ESEA was able to improve the ϕ_p criterion of the Structured Latin Hypercube design. However, in all the test cases the Structured Latin Hypercube design produced better ϕ_p criterion than the random Latin Hypercube design. As the time for generating the Structured Latin Hypercube design is very small we recommended to use this design as a good approximation of the Optimal Latin Hypercube design.

8. References

1. R.H. Myers and D.C. Montgomery, “Response Surface Methodology. Process and Product Optimization using Designed Experiments,” Wiley and Sons, Inc. New York, USA 1995.
2. S. Bates, J. Sienz, and V. Toropov. “Formulation of the Optimal Latin Hypercube Design of Experiments using a Permutation Genetic Algorithm, AIAA-2004-2011.
3. M.D. Morrsi and T.J. Mitchell, “Exploratory Designs for Computational Experiments,” Journal of Statistics Planning and Interference, 43:381-402, 1995.
4. K.Q. Ye, W. Li, and A. Sudjianto, “Algorithmic Construction of Optimal Symmetric Latin Hypercube Designs,” Jouranl of of Statistics Planning and Interference, 90:145-159, 2000.
5. R. Jinb, W. Chena, and A. Sudjianto. An Efficient Algorithm for Constructing Optimal Design of Computer Experiments. Journal of Statistical Planning and Inference, 134(1):268–287, September 2005.
6. Y. G. Saab and Y. B. Rao. Combinatorial Optimization by Stochastic Evolution. IEEE Transactions on Computer-Aided Design, 10:525–535, 1991.