Nested Maximin Latin Hypercube Designs in Two Dimensions

Husslage, B.G.M.; van Dam, E.R.; den Hertog, D.

Publication date:
2005

Document Version
Publisher's PDF, also known as Version of record

CentER

Discussion Paper

No. 2005–79

NESTED MAXIMIN LATIN HYPERCUBE DESIGNS IN TWO DIMENSIONS

By Bart Husslage, Edwin van Dam, Dick den Hertog

June 2005

ISSN 0924-7815

TILBURG UNIVERSITY

Nested maximin Latin hypercube designs in two dimensions

Bart Husslage* • Edwin van Dam† • Dick den Hertog

Department of Econometrics and Operations Research, Tilburg University,
P.O. Box 90153, 5000 LE Tilburg, The Netherlands

center.uvt.nl/phd_stud/husslage/ • *center.uvt.nl/staff/dam/* • *center.uvt.nl/staff/hertog/*

Abstract

In black box evaluation and optimization Latin hypercube designs play an important role. When dealing with multiple black box functions the need often arises to construct designs for all black boxes jointly, instead of individually. These so-called nested designs consist of two separate designs, one being a subset of the other, and are used to deal with linking parameters and sequential evaluations. In this paper we construct nested maximin designs in two dimensions. We show that different types of grids should be considered when constructing nested designs and discuss how to determine which grid to use best for a specific computer experiment. In the appendix to this paper maximin distances for different numbers of points are provided; the corresponding nested maximin designs can be found on the website *http://www.spacefillingdesigns.nl*.

Keywords: Circle packing, Latin hypercube design, linking parameters, non-collapsing, sequential simulation, space-filling.
JEL Classification: C90.

1 Introduction

Latin hypercube designs (LHDs) are extremely useful in the approximation of black box functions. Suppose that our aim is to approximate such a function on a box-constrained domain. By nature, a black box function is not given explicitly, however, we may perform function evaluations. As evaluations of the black box function often involve time-consuming computer simulations, we would like to construct an approximating model based on evaluations in a (small) number of points. See, e.g. Montgomery [10], Sacks et al [13], [14], Myers [12], Jones et al [7], Booker et al [1], and Den Hertog and Stehouwer [5]. We call such a set of evaluation points a *design*. As is recognized by several authors, such a design for computer experiments should at least satisfy the following two criteria (see Johnson et al [8] and Morris and Mitchell [11]). First of all, the design should be *space-filling* in some sense. When no details on the functional behavior of the response parameters are available, it is important to be able to obtain information from the entire design space. Therefore, design points should be "evenly spread" over the entire region. Secondly, the design should be *non-collapsing*. When one of the design parameters has (almost) no influence on the black box function value, two design points that differ only in this parameter will "collapse", i.e. they can be considered as the same point that is evaluated twice. For deterministic black box functions this is not a desirable situation. Therefore, two design points should not share any coordinate values when it is not known a priori which parameters are important. This can be accomplished by using Latin hypercube designs.

To obtain space-filling designs the evaluation points are chosen in such a way that the separation distance (i.e. the minimal distance among pairs of points) is maximized, leading to so-called maximin designs. Other space-filling designs, like minimax, IMSE, and maximum entropy designs, are also used in the literature. For a good survey of these designs see the book of Santner et al [15]. In this book it is also

*The research of B.G.M. Husslage is funded by the SamenwerkingsOrgaan Brabantse Universiteiten (SOBU).
†The research of E.R. van Dam has been made possible by a fellowship of the Royal Netherlands Academy of Arts and Sciences.

shown that maximin Latin hypercube designs generally speaking yield the best approximations. Only a few papers consider maximin designs, e.g. Trosset [18], Dimnaku et al [4], Locatelli and Raber [9], and Stinstra et al [16]. These papers describe heuristics to find approximate maximin designs. Morris and Mitchell [11] and Van Dam et al [3] consider maximin Latin hypercube designs.

In real-life problems there is a need for *nested designs*. We call a design *nested* when it consists of two separate designs, one being a subset of the other, see Van Dam et al [2]. Nested designs are useful when we have *linking parameters* or *sequential evaluations*.

To start with the first; consider a product that consists of two components, each of them represented by a black box function. To obtain proper approximating models a different number of function evaluations may be needed for each black box function. Moreover, in practice it may occur that the functions have an input parameter in common; such a parameter is called a *linking parameter*, see Husslage et al [6]. Evaluating a linking parameter at the same setting in both functions (i.e. component-wise) leads to an evaluation of the product. Not only do product evaluations provide a better understanding of the product, they are also very useful in the product optimization process. Another reason for using the same settings for (linking) parameters is due to physical restrictions on the simulation tools. Setting the parameters for computer experiments can be a time-consuming job in practice, since characteristics, like shape and structure, have to be redefined for every new experiment. Therefore, it is preferable to use the same settings as much as possible. By constructing nested designs we can determine the settings for linking parameters.

As an example of a real-life problem in which linking parameters play a role, we consider a collaborative optimization approach to optimize the design of a color picture tube, see Stinstra et al [17]. Such a tube consists of the main components screen, electron gun, and shadow mask. Stinstra et al [17] consider the collaborative design of several aspects of the shadow mask and the screen. Two of these aspects are the black functions describing Landing and Microphony. The Landing function measures the quality of the image, whereas the Microphony function measures how vulnerable the shadow mask is to external vibrations. Since the response parameters of both Landing and Microphony depend on the settings of the design parameters of the shadow mask, linking parameters play an important role, see Figure 1. As is argued by Husslage et al [6], the same settings should be used for these linking parameters as much as possible, giving rise to the need for nested designs.

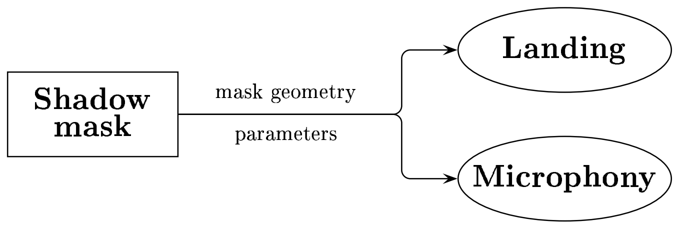

Figure 1: Linking parameters in tube design optimization.

Nested designs are also useful when dealing with sequential evaluations. In practice it is common that after evaluating an initial set of points, extra evaluations are needed. As an example, suppose we construct an approximating model for some black box function based on n_1 function evaluations. However, after validating the obtained model it turns out that an extra set of function evaluations is needed to build a proper model. We then face the problem of constructing a design on a total of, say, n_2 points, given the initial design on n_1 points. To anticipate on the possibility of extra evaluations, one can construct the two designs (on n_1 and n_2 points) at once, hence, by constructing a nested design.

We have just described why both Latin hypercube designs and nested designs are important. In this paper we will combine both types of designs and construct nested maximin Latin hypercube designs in two dimensions. We will focus on the problem of nesting two sets, X_1 and X_2, with $X_1 \subseteq X_2$, $X_i = \{(x_j, y_j) | j \in I_i\}$, and $|I_i| = n_i$, $i = 1, 2$. Hence, the index set $I_1 \subseteq I_2 = \{0, \ldots, n_2 - 1\}$ tells us which design points (x_j, y_j) are contained in both sets. We assume that all points (x_j, y_j) are contained in the box $[0, 1]^2$. We use scaling factors s_1 and s_2 to compare the minimal distances of the sets X_1 and X_2, respectively. Our aim is to determine the design points (x_j, y_j) and index set I_1 such that every set X_i is as much as possible space-filling with respect to the maximin criterion. To this end we define d_i as the (squared) minimal scaled Euclidean distance among all points in the set X_i, i.e. $d_i = \min_{j,k \in I_i, j \neq k} \frac{(x_j - x_k)^2 + (y_j - y_k)^2}{s_i}$. In this paper we will use $s_i = \frac{1}{n_i - 1}$. Then we have to maximize $d = \min\{d_1, d_2\}$ over all $I_1 \subseteq I_2$, with $|I_1| = n_1$, and $(x_j, y_j) \in [0,1]^2$, to find a nested maximin design.

In the rest of this paper we discuss different types of nested maximin designs and give examples for each of them. More results are provided in the appendix to this paper.

2 Nested maximin designs

When considering nested maximin designs there are several different types of designs we can distinguish, see Figure 2. A first division can be made by distinguishing between *unrestricted* (possibly collapsing) and *non-collapsing* designs. An unrestricted nested maximin design (consisting of two nested sets) can be found by solving the following mathematical problem:

$$\begin{array}{lll} \max & \min\limits_{\substack{j,k \in I_i \\ i=1,2;\, j \neq k}} & (n_i - 1)\left((x_j - x_k)^2 + (y_j - y_k)^2\right) \\ \text{s.t.} & I_1 \subseteq I_2 & \\ & |I_1| = n_1 & \\ & |I_2| = n_2 & \\ & 0 \leq x_j \leq 1, \quad j \in I_2 & \\ & 0 \leq y_j \leq 1, \quad j \in I_2. & \end{array} \tag{1}$$

Figure 3 gives an example of such an unrestricted nested maximin design where $(n_1, n_2) = (4, 9)$. In this figure the design points in X_1 are represented by black dots, the white dots represent the extra design points needed to complete X_2, hence, the black and white dots together make up the set X_2. In this particular example the nesting restriction does not reduce the maximin distances of the individual sets, i.e. they are both optimal. However, in general this reduction will occur for most combinations of n_1 and n_2.

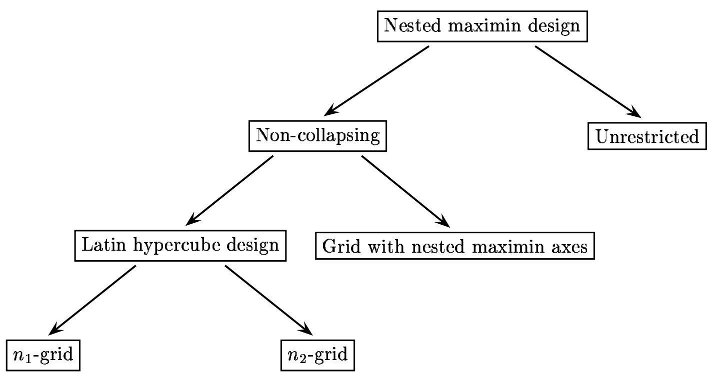

Figure 2: Several types of nested maximin designs.

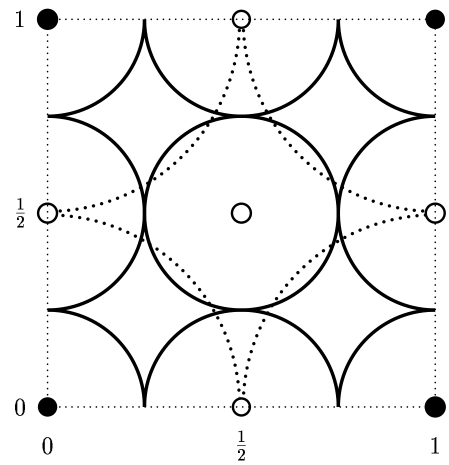

Figure 3: An unrestricted nested maximin design of $(n_1, n_2) = (4, 9)$ points, with $d = d_2 = 2$ and $d_1 = 3$.

As mentioned before, it is important to have non-collapsing designs when dealing with deterministic computer experiments. Therefore, we consider this type of designs in the rest of this paper. Unrestricted nested designs are discussed in another (forthcoming) paper. Note that by adding constraints to (1) to enforce that the x and y-levels are separated by some distance we will obtain a non-collapsing nested design. The problem now is to determine what value to take for this distance. We discuss two possibilities by distinguishing between *Latin hypercube designs* and *grids with nested maximin axes.*

2.1 Latin hypercube designs

There are two ways to use Latin hypercube designs (LHDs). We can either construct a Latin hypercube design based upon the first set (i.e. X_1), which we will call an n_1-grid, or we can construct a Latin hypercube design based upon the second set (i.e. X_2), which we will call an n_2-grid. Continuing our previous example where $(n_1, n_2) = (4, 9)$ the corresponding maximin Latin hypercube designs on the n_1-grid and n_2-grid are given in Figures 4 and 5, respectively.

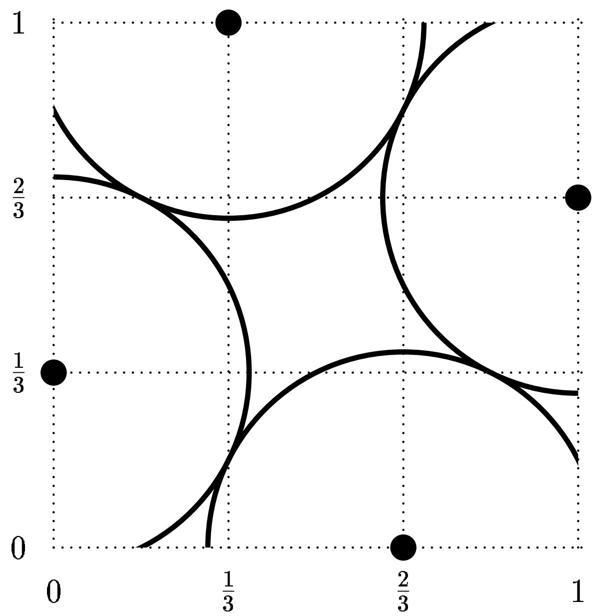

Figure 4: A maximin Latin hypercube design of 4 points, with $d_1 = 1.67$.

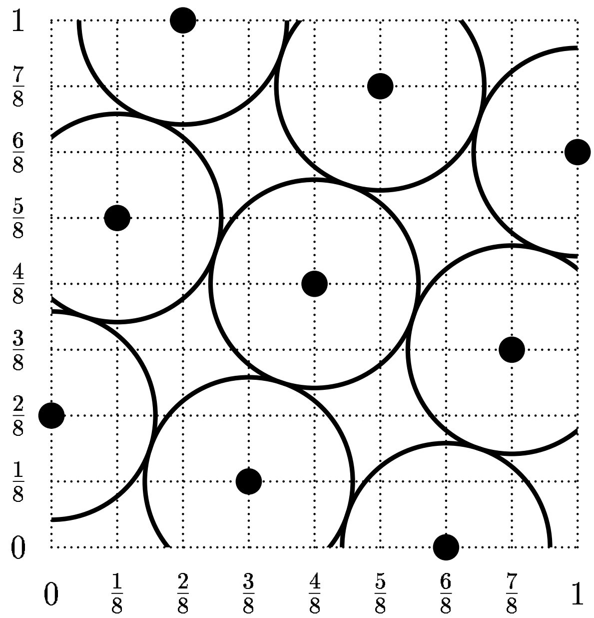

Figure 5: A maximin Latin hypercube design of 9 points, with $d_2 = 1.25$.

The n_2-grid To construct a nested Latin hypercube design on an n_2-grid we must choose n_1 points on this grid that make up the set X_1 and choose $n_2 - n_1$ extra points on the grid that (together with X_1) form the set X_2. Given the sets X_1 and X_2 we are interested in two measures: the "space-fillingness" of each set, represented by the d_i, and the "non-collapsingness" of each set on its axes. To find a nested space-filling design we maximize $d = \min\{d_1, d_2\}$. With respect to the non-collapsingness, note that an n_2-grid already gives optimal non-collapsingness for the set X_2. We therefore only have to add restrictions such that the projections onto the axes, i.e. the levels, of the design points in set X_1 will also be as space-filling as possible.

Now, let us first consider the case where $c_2 = \frac{n_2-1}{n_1-1} \in \mathbb{N}$. In this case we can easily maximize the non-collapsingness by limiting our choice of X_1-points to $\{0, \frac{1}{n_1-1}, \frac{2}{n_1-1}, \ldots, 1\}^2$, yielding equidistantly distributed projections of the design points onto the axes. See, for example, the nested maximin Latin hypercube design of $(n_1, n_2) = (16, 31)$ points (with $c_2 = 2$) in Figure 6. Using an extension of the branch-and-bound algorithm of Van Dam et al [3] we were able to find nested maximin Latin hypercube designs for n_2 up to 32 in case $c_2 \in \mathbb{N}$. Table 3 in the appendix gives the corresponding maximin distances.

For $c_2 \notin \mathbb{N}$ the situation is more complicated. It is then no longer possible to have equidistantly distributed projections onto the axes, since we are bounded to the n_2-grid and $n_1 - 1$ is no longer a divisor of $n_2 - 1$. From the one-dimensional case we know that when the X_2-levels are equidistantly distributed, like we have now on the n_2-grid, it is optimal to have $\lfloor c_2 \rfloor - 1$ or $\lceil c_2 \rceil - 1$ X_2-levels between the X_1-levels; see van Dam et al [2]. Hence, should the design collapse to one dimension, having chosen the X_1-points such that its levels fulfill above restriction will result in an optimal one-dimensional nested maximin design. Therefore, we require the X_1-levels to be separated by either $\lfloor c_2 \rfloor \frac{1}{n_2-1}$ or $\lceil c_2 \rceil \frac{1}{n_2-1}$. Note that there are multiple grids possible for the set X_1. Figure 7 gives an example of a nested maximin design on an n_2-grid where $(n_1, n_2) = (4, 9)$. The results found with the extended branch-and-bound algorithm can be found in Table 4 of the appendix.

The n_1-grid The idea here is the same as with the n_2-grid. We again demand to have $\lfloor c_2 \rfloor - 1$ or $\lceil c_2 \rceil - 1$ X_2-levels between the X_1-levels. Hence, the X_2-levels will be separated by either $\frac{1}{\lfloor c_2 \rfloor} \frac{1}{n_1-1}$ or $\frac{1}{\lceil c_2 \rceil} \frac{1}{n_1-1}$. See Figure 8 for an example of a nested maximin design on an n_1-grid where $(n_1, n_2) = (4, 9)$. More results, for n_2 up to 15, can again be found in the appendix, in Table 5.

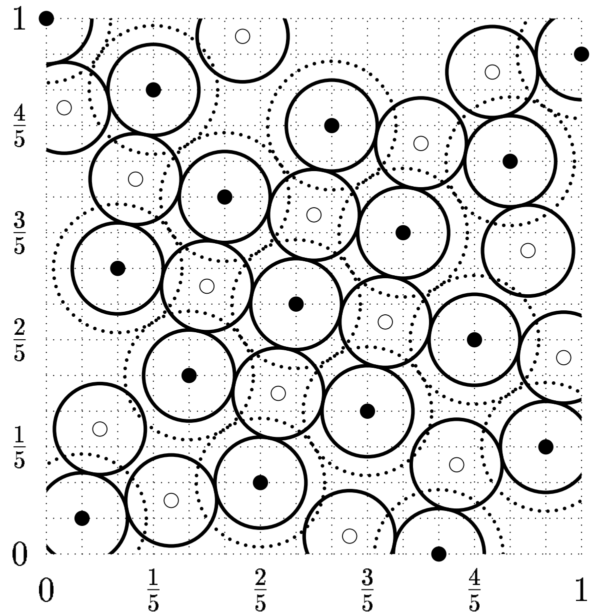

Figure 6: A nested maximin Latin hypercube design of $(n_1, n_2) = (16, 31)$ points, with $d = d_1 = d_2 = 0.8667$.

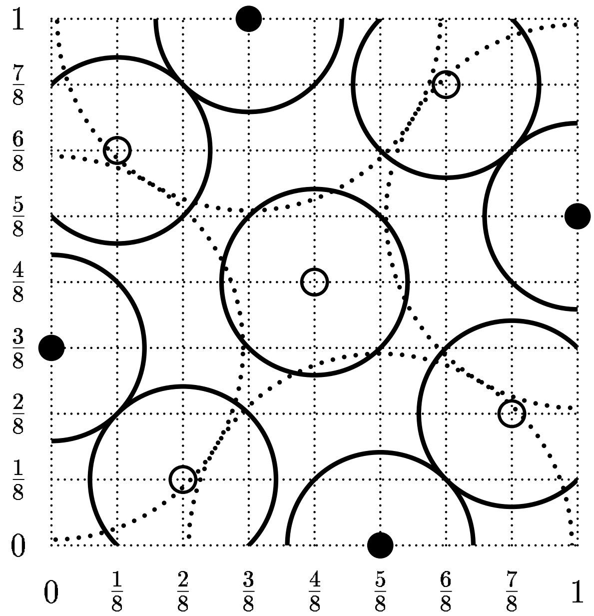

Figure 7: A nested maximin n_2-Latin hypercube design of $(n_1, n_2) = (4, 9)$ points, with $d = d_2 = 1.00$ and $d_1 = 1.59$.

2.2 Grids with nested maximin axes

When using a Latin hypercube design we clearly favor one of the sets, X_1 or X_2, by using an n_1-grid or an n_2-grid, respectively. If both sets are assumed to be of equal importance we would like to treat them equally as well. To deal with this problem we can use the (known) one-dimensional nested maximin designs (see van Dam et al [2]) on the axes and construct two-dimensional nested maximin designs on the grids obtained this way. Note that in this case the projection of the points onto the axes will always be space-filling, with respect to the maximin criterion. Furthermore, note that the one-dimensional maximin designs are (again) not unique, hence, there are multiple grids possible. See Figure 9 for an example of a nested maximin design of $(n_1, n_2) = (4, 9)$ points on a grid with nested maximin axes. Table 6 in the appendix gives the maximin distances for values of n_2 up to 15.

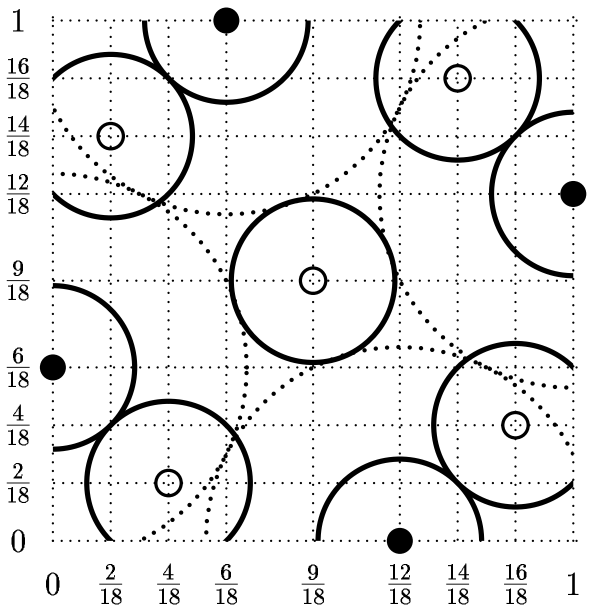

Figure 8: A nested maximin n_1-Latin hypercube design of $(n_1, n_2) = (4, 9)$ points, with $d = d_2 = 0.79$ and $d_1 = 1.67$.

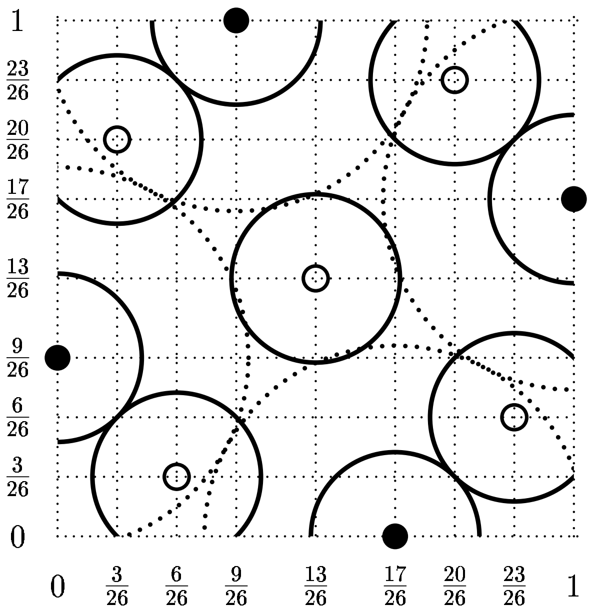

Figure 9: A nested maximin design of $(n_1, n_2) = (4, 9)$ points on a grid with nested maximin axes, with $d = d_2 = 0.85$ and $d_1 = 1.64$.

2.3 Comparing the different types of grids

In the previous sections we discussed three types of grids. The question now remains as when to use which type? As an example, consider Table 1, which summarizes the results of the previous sections for the case $(n_1, n_2) = (4, 9)$.

grid type	d	d_1	d_2	figure
nested n_2-grid	1.00	1.59	1.00	7
nested n_1-grid	0.79	1.67	0.79	8
grid with nested maximin axes	0.85	1.64	0.85	9

Table 1: Maximin distances for different types of nested grid-designs where $(n_1, n_2) = (4, 9)$.

When determining which grid to use there are a few aspects to consider. First, if we are more interested in the space-fillingness of a design we should choose the grid which yields the largest maximin distance, e.g. the nested n_2-grid in Table 1. Note, however, that the maximin distance does not only depend on the used grid, but also on the values of n_1 and n_2. Therefore, it may be wise to consider several different pairs (n_1, n_2) for each type of grid in order to find a satisfiable nested design. Besides the maximin distance there is also the non-collapsingness to consider, especially when it is not known a priori which parameters are important. Should the design collapse then we would like to have the one-dimensional design to be space-filling, e.g. by choosing a grid with nested maximin axes.

The reason why we are using a nested design may also affect our choice. For example, an n_1-grid is preferable for sequential evaluations, since we know for sure that the first set of design points will be evaluated (furthermore, this set should give us a good idea about the whole region, so should be as space-filling as possible), whereas the evaluation of an extra set of design points depends on the previously evaluated set. In the same setting, an n_2-grid is preferable when we demand that the final set of design points, hence X_2, should be a Latin hypercube design, as is often the case in practice. In the case of linking parameters the grid choice mostly depends on the question which of the two sets we consider to be most important, thus using an n_1-grid or an n_2-grid. A grid with nested maximin axes should be used when we have no preference for either one of the sets.

From above discussion it follows that the notion of what is the best nested grid-design clearly depends on the user's preference. Fortunately, there are some special cases, i.e. when $c_2 \in \mathbb{N}$, that make the comparison of the various nested grid-designs superfluous. In these cases we do not have to differentiate between different types, since they will all come down to the same nested design (and maximin distance).

Besides nested designs that maximize our objective function $d = \min\{d_1, d_2\}$ there are also some other interesting nested designs to consider: dominant nested designs. We will call a combination of distances (d_1, d_2) dominant if it is not possible to improve one of the distances, without deteriorating the other distance. For $c_2 \in \mathbb{N}$ and $n \leq 32$ we were able to compute all dominant nested designs. Besides the optimal ones in Table 3, Table 2 provides the pairs (n_1, n_2) which have more than one dominant combination. In this latter table the distances d_1 and d_2 of the optimal design are given first, followed by the distances of the other dominant design(s). Note that the dominant nested design of $(11, 21)$ points is also optimal, i.e. both designs have the same distances. For $(9, 17)$ and $(10, 19)$ points, however, the objective values of the dominant designs are equal (0.6250 and 0.5556, respectively), but the individual distances are smaller ($1.1250 < 1.2500$ and $1.0000 < 1.1111$, respectively). As an example, Figures 10 and 11 show the two other dominant nested designs of $(16, 31)$ points (the optimal one is given in Figure 6).

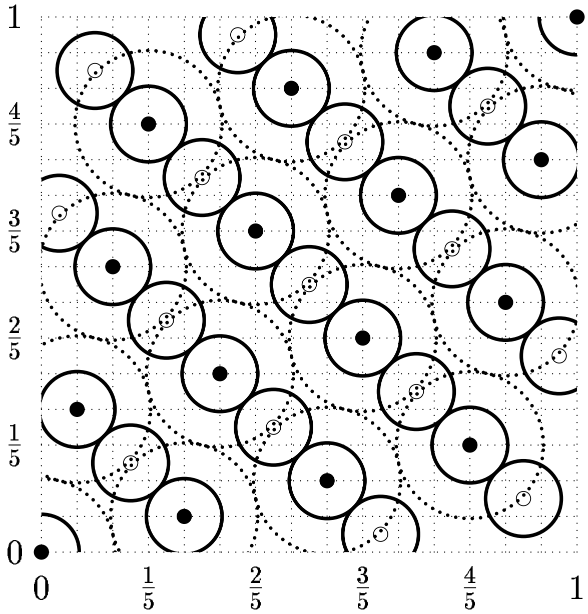

Figure 10: A dominant nested Latin hypercube design of $(n_1, n_2) = (16, 31)$ points, with $d = d_2 = 0.6000$ and $d_1 = 1.1333$.

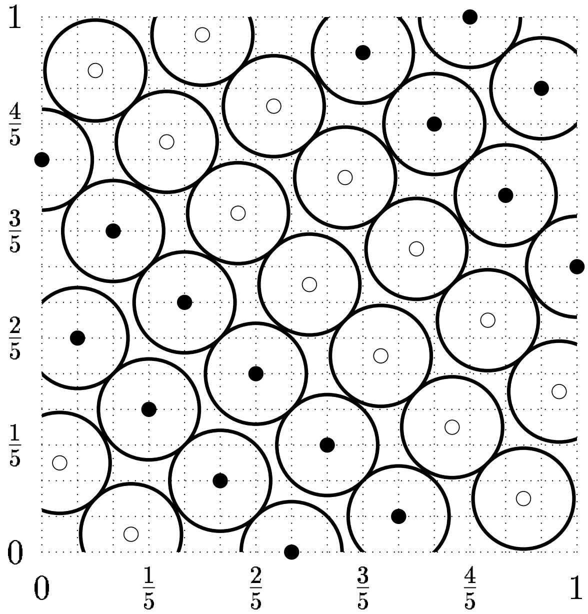

Figure 11: A dominant nested Latin hypercube design of $(n_1, n_2) = (16, 31)$ points, with $d = d_1 = 0.5333$ and $d_2 = 1.0667$.

n_1	n_2	dominant combinations	n_1	n_2	dominant combinations
4	10	(0.6667, 0.8889), (1.6667, 0.5556)	5	25	(1.2500, 0.8333), (0.5000, 1.0833)
4	16	(1.6667, 0.8667), (0.6667, 1.1333)	7	25	(1.3333, 0.7500), (0.3333, 1.0833)
6	16	(1.0000, 0.5333), (0.4000, 1.1333)	9	25	(1.0000, 1.0833), (1.2500, 0.8333)
9	17	(1.2500, 0.6250), (0.6250, 1.1250)	14	27	(1.0000, 1.0000), (1.3077, 0.6923)
4	19	(1.6667, 0.9444), (0.6667, 1.0000)	4	28	(1.6667, 0.9259), (0.6667, 0.9630)
7	19	(1.3333, 0.9444), (0.3333, 1.0000)	10	28	(0.8889, 0.9630), (1.1111, 0.7407)
10	19	(1.1111, 0.5556), (0.5556, 1.0000)	5	29	(1.2500, 0.9286), (0.5000, 1.0357)
11	21	(1.0000, 0.5000), (0.5000, 1.0000)	8	29	(1.1429, 0.8929), (0.7143, 0.9286)
8	22	(1.1429, 0.8095), (0.2857, 0.8571)	15	29	(0.9286, 0.9286), (1.2143, 0.6429)
12	23	(0.7273, 1.1818), (0.9091, 0.4545)	7	31	(1.3333, 0.8333), (0.3333, 0.8667)
4	25	(1.6667, 1.0417), (0.6667, 1.0833)	16	31	(0.8667, 0.8667), (1.1333, 0.6000), (0.5333, 1.0667)

Table 2: Pairs (n_1, n_2) with more than one dominant combination for $c_2 \in \mathbb{N}$, $n_2 \leq 32$.

3 Conclusions

A two-dimensional nested design consists of two separate designs, one being a subset of the other. Using these nested designs, instead of traditional designs of computer experiments, is useful when dealing with linking parameters or sequential evaluations, since nested designs are able to capture the dependencies between the two black boxes or evaluation stages (with respect to the design parameters). This paper focuses on constructing nested maximin Latin hypercube designs in two dimensions. The maximin criterion is used to find space-filling nested designs, i.e. designs with the design points spread over the entire design space. By choosing the design points on a grid we insure non-collapsingness, i.e. no two design points will have the same coordinate values. We distinguish between three types of grids: an n_1-Latin hypercube design, an n_2-Latin hypercube design, and a grid with nested maximin axes. Which grid to use is found to mainly depend on the nature of the computer experiment and the user's preference. For all three grids maximin distances are provided for values of n_2 up to 15. In the special case where $n_1 - 1$ is a divisor of $n_2 - 1$ there is no need to differentiate between different types of grids, since they all come down to the same nested design. For pairs (n_1, n_2) that satisfy this condition maximin distances up to $n_2 = 32$ are provided. All corresponding nested maximin designs can be found on the website *http://www.spacefillingdesigns.nl*.

References

[1] Booker, A.J., Dennis, J.E., Frank, P.D., Serafini, D.B., Torczon, V., and Trosset, M.W. (1999). A rigorous framework for optimization of expensive functions by surrogates, *Structural Optimization* 17, 1–13.

[2] Dam, E.R. van, Husslage, B.G.M., and Hertog, D. den (2004). One-dimensional nested maximin designs, *CentER Discussion Paper 2004-66*, 14 pp.

[3] Dam, E.R. van, Husslage, B.G.M., Hertog, D. den, and Melissen, J.B.M. (2005). Maximin Latin hypercube designs in two dimensions, *CentER Discussion Paper 2005-08*, 16 pp.

[4] Dimnaku, A., Kincaid, R., and Trosset, M.W. (2002). Approximate solutions of continuous dispersion problems, *2002 ISOLDE conference*, *http://www.math.wm.edu/˜trosset/*.

[5] Hertog, D. den, and Stehouwer, H.P. (2002). Optimizing color picture tubes by high-cost nonlinear programming, *European Journal on Operations Research* 140(2), 197–211.

[6] Husslage, B.G.M., Dam, E.R. van, Hertog, D. den, Stehouwer, H.P., and Stinstra, E.D. (2003). Collaborative metamodeling: Coordinating simulation-based product design, *Concurrent Engineering: Research and Applications* 11(4), 267–278.

[7] Jones, D., Schonlau, M., and Welch, W. (1998). Efficient global optimization of expensive black-box functions, *Journal of Global Optimization* 13, 455–492.

[8] Johnson, M.E., Moore, L.M., and Ylvisaker, D. (1990). Minimax and maximin distance designs, *Journal of Statistical Planning and Inference* 26, 131–148.

[9] Locatelli, M., and Raber, U. (2002). Packing equal circles in a square: A deterministic global optimization approach, *Discrete Applied Mathematics* 122(1-3), 139–166.

[10] Montgomery, D.C. (1984). *Design and analysis of experiments*, Second edition, John Wiley & Sons, New York.

[11] Morris, M.D., and Mitchell, T.J. (1995). Exploratory designs for computer experiments, *Journal of Statistical Planning and Inference* 43, 381–402.

[12] Myers, R.H. (1999). Response surface methodology – Current status and future directions, *Journal of Quality Technology* 31, 30–74.

[13] Sacks, J., Schiller, S.B., and Welch, W.J. (1989). Designs for computer experiments, *Technometrics* 31, 41–47.

[14] Sacks, J., Welch, W.J., Mitchell, T.J., and Wynn, H.P. (1989). Design and analysis of computer experiments, *Statistical Science* 4, 409–435.

[15] Santner, Th.J., Williams, B.J., and Notz, W.I. (2003). *The design and analysis of computer experiments, Springer Series in Statistics*, Springer-Verlag, New York.

[16] Stinstra, E.D., Hertog, D. den, Stehouwer, H.P., and Vestjens, A. (2003). Constrained maximin designs for computer experiments, *Technometrics* 45(4), 340–346.

[17] Stinstra, E.D., Stehouwer, H.P., and Heijden, J. van der (2003). Collaborative tube design optimization: An integral meta-modeling approach, *Proceedings of the Fifth ISSMO Conference on Engineering Design Optimization*, Como, Italy.

[18] Trosset, M.W. (1999). Approximate maximin distance designs, *Proceedings of the Section on Physical and Engineering Sciences*, 223–227, Alexandria, VA.

Appendix

n_1	n_2	d	d_1	d_2	n_1	n_2	d	d_1	d_2
2	3	1.0000	2.0000	1.0000	2	21	0.9000	2.0000	0.9000
2	4	0.6667	2.0000	0.6667	3	21	0.9000	1.0000	0.9000
2	5	0.5000	2.0000	0.5000	5	21	0.9000	1.2500	0.9000
3	5	0.5000	1.0000	0.5000	6	21	0.8500	1.0000	0.8500
2	6	1.0000	2.0000	1.0000	11	21	0.5000	1.0000	0.5000
2	7	0.8333	2.0000	0.8333	2	22	0.8571	2.0000	0.8571
3	7	0.8333	1.0000	0.8333	4	22	0.8571	1.6667	0.8571
4	7	0.6667	0.6667	1.3333	8	22	0.8095	1.1429	0.8095
2	8	0.7143	2.0000	0.7143	2	23	0.9091	2.0000	0.9091
2	9	1.0000	2.0000	1.0000	3	23	0.9091	1.0000	0.9091
3	9	1.0000	1.0000	1.0000	12	23	0.7273	0.7273	1.1818
5	9	1.2500	1.2500	1.2500	2	24	1.0870	2.0000	1.0870
2	10	0.8889	2.0000	0.8889	2	25	1.0833	2.0000	1.0833
4	10	0.6667	0.6667	0.8889	3	25	1.0000	1.0000	1.0833
2	11	1.0000	2.0000	1.0000	4	25	1.0417	1.6667	1.0417
3	11	1.0000	1.0000	1.0000	5	25	0.8333	1.2500	0.8333
6	11	1.0000	1.0000	1.0000	7	25	0.7500	1.3333	0.7500
2	12	0.9091	2.0000	0.9091	9	25	1.0000	1.0000	1.0833
2	13	0.8333	2.0000	0.8333	13	25	1.0833	1.0833	1.0833
3	13	0.8333	1.0000	0.8333	2	26	1.0400	2.0000	1.0400
4	13	0.8333	1.6667	0.8333	6	26	1.0000	1.0000	1.0000
5	13	0.6667	1.2500	0.6667	2	27	1.0000	2.0000	1.0000
7	13	0.8333	1.3333	0.8333	3	27	1.0000	1.0000	1.0000
2	14	1.0000	2.0000	1.0000	14	27	1.0000	1.0000	1.0000
2	15	0.9286	2.0000	0.9286	2	28	0.9630	2.0000	0.9630
3	15	0.7143	1.0000	0.7143	4	28	0.9259	1.6667	0.9259
8	15	0.7143	1.1429	0.7143	10	28	0.8889	0.8889	0.9630
2	16	1.1333	2.0000	1.1333	2	29	0.9286	2.0000	0.9286
4	16	0.8667	1.6667	0.8667	3	29	0.9286	1.0000	0.9286
6	16	0.5333	1.0000	0.5333	5	29	0.9286	1.2500	0.9286
2	17	1.0625	2.0000	1.0625	8	29	0.8929	1.1429	0.8929
3	17	1.0000	1.0000	1.0625	15	29	0.9286	0.9286	0.9286
5	17	0.8125	1.2500	0.8125	2	30	1.0000	2.0000	1.0000
9	17	0.6250	1.2500	0.6250	2	31	0.9667	2.0000	0.9667
2	18	1.0000	2.0000	1.0000	3	31	0.9667	1.0000	0.9667
2	19	1.0000	2.0000	1.0000	4	31	0.8667	1.6667	0.8667
3	19	1.0000	1.0000	1.0000	6	31	0.8667	1.0000	0.8667
4	19	0.9444	1.6667	0.9444	7	31	0.8333	1.3333	0.8333
7	19	0.9444	1.3333	0.9444	11	31	0.8667	1.0000	0.8667
10	19	0.5556	1.1111	0.5556	16	31	0.8667	0.8667	0.8667
2	20	0.9474	2.0000	0.9474	2	32	0.9355	2.0000	0.9355

Table 3: Maximin distances for nested designs on an LHD, $c_2 \in \mathbb{N}$.

n_1	n_2	d	d_1	d_2	n_1	n_2	d	d_1	d_2
3	4	0.6667	1.7778	0.6667	7	12	0.9091	0.9917	0.9091
4	5	1.2500	1.8750	1.2500	8	12	0.9091	1.0413	0.9091
3	6	1.0000	1.4400	1.0000	9	12	0.8595	0.8595	1.1818
4	6	0.9600	0.9600	1.0000	10	12	0.9669	0.9669	1.1818
5	6	0.8000	0.8000	1.0000	11	12	1.0744	1.0744	1.1818
5	7	0.8889	0.8889	1.3333	6	13	0.8333	1.0069	0.8333
6	7	1.1111	1.1111	1.3333	8	13	0.8333	0.9722	0.8333
3	8	1.1429	1.3061	1.1429	9	13	0.8333	1.0000	0.8333
4	8	0.7143	1.7755	0.7143	10	13	0.8333	1.1250	0.8333
5	8	0.7143	1.0612	0.7143	11	13	0.9028	0.9028	1.0833
6	8	0.8163	0.8163	1.1429	12	13	0.9931	0.9931	1.0833
7	8	0.9796	0.9796	1.1429	3	14	0.7692	1.1598	0.7692
4	9	1.0000	1.5938	1.0000	4	14	0.7692	1.7219	0.7692
6	9	1.0000	1.0156	1.0000	5	14	0.6154	1.3728	0.6154
7	9	0.9375	0.9375	1.2500	6	14	0.7692	1.1834	0.7692
8	9	1.0938	1.0938	1.2500	7	14	0.7692	1.1361	0.7692
3	10	0.8889	1.2346	0.8889	8	14	0.7692	1.0769	0.7692
5	10	0.6420	0.6420	0.8889	9	14	0.8047	0.8047	1.3077
6	10	0.8025	0.8025	0.8889	10	14	0.9053	0.9053	1.3077
7	10	0.8889	0.9630	0.8889	11	14	1.0059	1.0059	1.3077
8	10	0.8642	0.8642	1.1111	12	14	1.1065	1.1065	1.3077
9	10	0.9877	0.9877	1.1111	13	14	1.2071	1.2071	1.3077
4	11	0.8000	1.3500	0.8000	4	15	0.7653	0.7653	1.2143
5	11	0.8000	1.1600	0.8000	5	15	0.9286	1.3265	0.9286
7	11	0.8000	1.0800	0.8000	6	15	0.7143	0.8673	0.7143
8	11	0.9100	0.9100	1.0000	7	15	1.2143	1.5306	1.2143
9	11	0.8000	1.0400	0.8000	9	15	0.7143	1.1837	0.7143
10	11	0.9000	0.9000	1.0000	10	15	0.8265	0.8265	0.9286
3	12	0.9091	1.0083	0.9091	11	15	0.9184	0.9184	0.9286
4	12	1.1818	1.6116	1.1818	12	15	0.9541	0.9541	1.2143
5	12	0.7273	1.1240	0.7273	13	15	1.0408	1.0408	1.2143
6	12	0.9091	1.1983	0.9091	14	15	1.1276	1.1276	1.2143

Table 4: Maximin distances for nested designs on an n_2-LHD, $c_2 \notin \mathbb{N}$.

n_1	n_2	d	d_1	d_2	n_1	n_2	d	d_1	d_2
3	4	0.3750	1.0000	0.3750	7	12	0.7639	1.3333	0.7639
4	5	1.1111	1.6667	1.1111	8	12	0.7143	0.7143	1.0102
3	6	0.8681	1.0000	0.8681	9	12	1.0000	1.0000	1.1172
4	6	0.6667	0.6667	1.1111	10	12	0.8889	0.8889	1.0864
5	6	0.7813	1.2500	0.7813	11	12	1.0000	1.0000	1.1000
5	7	0.9375	1.2500	0.9375	6	13	0.9067	1.0000	0.9067
6	7	1.0000	1.0000	1.0800	8	13	0.7143	0.7143	1.1020
3	8	0.8750	1.0000	0.8750	9	13	0.8438	1.0000	0.8438
4	8	0.6265	0.6667	0.6265	10	13	0.8889	0.8889	0.9630
5	8	1.0938	1.2500	1.0938	11	13	0.8000	0.8000	0.9600
6	8	0.7000	1.0000	0.7000	12	13	0.9091	0.9091	0.9917
7	8	0.8333	0.8333	0.9722	3	14	0.8622	1.0000	0.8622
4	9	0.7901	1.6667	0.7901	4	14	0.8703	1.6667	0.8703
6	9	0.8000	1.0000	0.8000	5	14	0.7222	1.2500	0.7222
7	9	0.8333	0.8333	1.0000	6	14	0.7656	1.0000	0.7656
8	9	0.7347	1.1429	0.7347	7	14	0.8526	1.3333	0.8526
3	10	0.7200	1.0000	0.7200	8	14	0.6633	1.1429	0.6633
5	10	0.5000	0.5000	0.7813	9	14	0.6250	0.6250	1.0156
6	10	0.9000	1.0000	0.9000	10	14	0.8889	0.8889	1.0432
7	10	0.6250	1.3333	0.6250	11	14	0.8450	1.0000	0.8450
8	10	0.7143	0.7143	0.9184	12	14	0.9091	0.9091	1.0744
9	10	0.9141	1.0000	0.9141	13	14	1.0833	1.0833	1.1285
4	11	0.7716	1.6667	0.7716	4	15	0.8089	1.6667	0.8089
5	11	0.5556	1.2500	0.5556	5	15	0.7109	1.2500	0.7109
7	11	0.6944	1.3333	0.6944	6	15	0.8244	1.0000	0.8244
8	11	0.7143	0.7143	1.0204	7	15	0.9182	1.3333	0.9182
9	11	1.0000	1.0000	1.0156	9	15	0.6250	0.6250	0.9844
10	11	0.8889	0.8889	0.9877	10	15	0.8889	0.8889	1.1235
3	12	0.7639	1.0000	0.7639	11	15	0.9100	1.0000	0.9100
4	12	0.9931	1.6667	0.9931	12	15	0.7521	1.1818	0.7521
5	12	0.6111	1.2500	0.6111	13	15	0.8333	0.8333	0.9722
6	12	0.8922	1.0000	0.8922	14	15	1.0000	1.0000	1.0355

Table 5: Maximin distances for nested designs on an n_1-LHD, $c_2 \notin \mathbb{N}$.

n_1	n_2	d	d_1	d_2	n_1	n_2	d	d_1	d_2
3	4	0.4898	1.3061	0.4898	7	12	0.7856	1.2593	0.7856
4	5	1.1837	1.7755	1.1837	8	12	0.7255	0.7255	1.0624
3	6	0.8264	1.1901	0.8264	9	12	0.9124	0.9124	0.9995
4	6	0.7474	0.7474	1.2457	10	12	0.9339	0.9339	0.9726
5	6	0.9168	0.9452	0.9168	11	12	1.0204	1.0204	1.1224
5	7	0.9796	1.0612	0.9796	6	13	0.9194	0.9194	0.9614
6	7	1.2457	1.2457	1.2561	8	13	0.6655	1.0774	0.6655
3	8	0.9956	1.1378	0.9956	9	13	0.7653	1.0000	0.7653
4	8	0.6385	1.7297	0.6385	10	13	0.8844	0.9184	0.8844
5	8	0.9979	1.2029	0.9979	11	13	0.8651	0.8651	0.8979
6	8	0.8328	0.8804	0.8328	12	13	0.9449	0.9449	1.0307
7	8	1.0647	1.0647	1.0711	3	14	0.8248	0.9273	0.8248
4	9	0.8521	1.6420	0.8521	4	14	0.8048	1.5080	0.8048
6	9	0.9168	0.9168	0.9452	5	14	0.6456	1.3355	0.6456
7	9	0.8878	0.8878	1.0000	6	14	0.7666	0.9452	0.7666
8	9	0.9979	0.9979	1.1405	7	14	0.7855	1.1775	0.7855
3	10	0.7978	1.1080	0.7978	8	14	0.6770	1.1665	0.6770
5	10	0.5917	0.5917	0.8550	9	14	0.6740	0.6740	1.0576
6	10	0.9371	0.9600	0.9371	10	14	0.9216	0.9216	1.1643
7	10	0.7891	0.7891	0.8163	11	14	0.9552	0.9552	1.2418
8	10	0.8275	0.8512	0.8275	12	14	1.0495	1.0495	1.1219
9	10	0.9050	0.9050	1.0181	13	14	1.1484	1.1484	1.2441
4	11	0.7856	1.4648	0.7856	4	15	0.8117	1.6519	0.8117
5	11	0.6612	1.2066	0.6612	5	15	0.8089	1.2844	0.8089
7	11	0.7785	0.9343	0.7785	6	15	0.8028	0.9469	0.8028
8	11	0.8037	0.8037	1.1481	7	15	1.0933	1.4518	1.0933
9	11	0.9452	0.9452	1.0019	9	15	0.6348	0.6348	1.0489
10	11	0.9371	0.9371	0.9600	10	15	0.8647	0.8647	1.1573
3	12	0.8318	1.0019	0.8318	11	15	0.8596	0.8596	1.0598
4	12	1.0506	1.6482	1.0506	12	15	0.9360	0.9360	1.0295
5	12	0.6374	1.2187	0.6374	13	15	0.9846	0.9846	1.1487
6	12	0.9046	1.1448	0.9046	14	15	1.0749	1.0749	1.1576

Table 6: Maximin distances for nested designs on a grid with nested maximin axes, $c_2 \notin \mathbb{N}$.