Statistica Sinica **8**(1998), 965-977

SELECTING LATIN HYPERCUBES USING CORRELATION CRITERIA

Boxin Tang

University of Memphis and University of Western Ontario

Abstract: Latin hypercube designs have recently found wide applications both in design of experiments and in numerical integration. An important property of this class of designs is that they achieve uniformity in each univariate margin. In this article we study the use of correlation criteria to select a Latin hypercube. We introduce the polynomial canonical correlation of two vectors and argue that a design which has a small polynomial canonical correlation for each pair of its columns is preferred. An algorithm for reducing polynomial canonical correlations of a Latin hypercube is developed. The implementation of the algorithm is discussed, and its performance investigated. Comparison with Owen's algorithm is also made.

Key words and phrases: Canonical correlation, computer experiments, exploratory designs.

1. Introduction

Latin hypercube sampling was introduced by McKay, Conover and Beckman (1979) for numerically evaluating a multiple integral. Its key property is stratifying each univariate margin, due to which it has also found wide applications in computer experiments (Welch et al. (1992)). In this article, a Latin hypercube is considered as an $n \times m$ matrix of which each column is a permutation of $1, \ldots, n$. Each Latin hypercube enjoys the property of achieving uniformity in each of the m univariate margins. Our definition of a Latin hypercube is slightly different from that of a Latin hypercube sample and in fact is in agreement with that of a lattice sample (Patterson (1954)).

In many applications, other properties besides univariate uniformity are also required of a Latin hypercube. We then need to select a Latin hypercube using some appropriate criterion. A design with its points uniformly scattered in the design region is widely accepted especially in situations where little knowledge is known about the function to be modeled. The maximin distance criterion ((Johnson, Moore and Ylvisaker (1990)) has the effect of spreading design points uniformly in the design region and has been used for the selection of a Latin hypercube by Morris and Mitchell (1992) and Tang (1994). Morris and Mitchell approached the selection problem through an algorithm based on simulated annealing while Tang provided a theoretical result for a special case. Another work

along this line is that of Tang (1993), who constructed a subclass of Latin hypercubes using orthogonal arrays. These orthogonal array based Latin hypercubes achieve uniformity in each r-variate margin when an orthogonal array of strength r is used for construction.

In this article, we examine the use of correlation criteria for selecting a Latin hypercube. We introduce the polynomial canonical correlation of two vectors using the idea of canonical correlation, and argue that a Latin hypercube design with a small value of polynomial canonical correlation for each pair of its columns is preferred. Iman and Conover (1982) and Owen (1994) have considered reducing the (ordinary) correlations of a Latin hypercube. Main effect regression models are employed to motivate our approach, although other arguments for preferring Latin hypercubes with low correlations exist (Iman and Conover (1982), Owen (1994)). An algorithm for reducing the polynomial canonical correlations of a Latin hypercube is developed, its implementation discussed, and its performance examined. Our algorithm has some bearing on Owen's algorithm (1994).

We organize the article as follows. Section 2 introduces main effect regression models and polynomial canonical correlation. It is argued that a design with small polynomial canonical correlations is preferred. In section 3, we develop an algorithm for reducing the polynomial canonical correlations of a Latin hypercube, and also discuss the computing costs and implementation of the algorithm. The performance of the algorithm is investigated in Section 4. Some further remarks on the algorithm are given in Section 5.

2. Main Effect Model and Polynomial Canonical Correlation

2.1. Main effect polynomial regression

Consider the following main effect model in which for each regressor variable x_j, its main effect can be represented by a polynomial of order q

$$y = \beta_0 + \sum_{j=1}^{m} (\beta_{j1} x_j + \cdots + \beta_{jq} x_j^q) + \epsilon. \tag{1}$$

Here for simplicity the same order q is chosen and in fact our arguments apply quite generally to the case where polynomials of different orders are considered. Suppose n independent observations from the model are available. In the data setting, model (1) can be expressed as

$$y_i = \beta_0 + \sum_{j=1}^{m} (\beta_{j1} x_{ij} + \cdots + \beta_{jq} x_{ij}^q) + \epsilon_i,$$

for $i = 1, \ldots, n$. It is more convenient to work with the following centered version

$$y_i = \bar{y} + \sum_{j=1}^{m} \left[\beta_{j1} (x_{ij} - \bar{x}_j) + \cdots + \beta_{jq} (x_{ij}^q - \bar{x}_j^q) \right] + \epsilon_i, \tag{2}$$

where $\bar{y} = \sum_{i=1}^{n} y_i/n$, and $\bar{x}_j = \sum_{i=1}^{n} x_{ij}/n, \ldots, \bar{x}_j^q = \sum_{i=1}^{n} x_{ij}^q/n$. Corresponding to the model in (2), the full design matrix is given by

$$(\mathbf{1}, X_1, \ldots, X_m), \tag{3}$$

where $\mathbf{1}$ is the vector of all ones, and the sth column of the matrix X_j is given by the vector $(x_{1j}^s - \bar{x}_j^s, \ldots, x_{nj}^s - \bar{x}_j^s)^t$, $s = 1, \ldots, q$. Suppose that a Latin hypercube design is used to estimate the main effects $\beta_j = (\beta_{j1}, \ldots, \beta_{jq})^t$, $j = 1, \ldots, m$. It is desirable that the main effects β_j, $j = 1, \ldots, m$, can be uncorrelatedly estimated. (This does not exclude the possibility that the estimates of the components of each β_j be correlated.) We can achieve this if $X_1, \ldots, X_m$ are mutually orthogonal, that is, $(X_{j_1})^t X_{j_2} = \mathbf{0}$, for $1 \le j_1 < j_2 \le m$, where $\mathbf{0}$ denotes the zero matrix of order q. However, for a Latin hypercube design, the condition $(X_{j_1})^t X_{j_2} = \mathbf{0}$, for $1 \le j_1 < j_2 \le m$, can almost never be strictly satisfied. Instead, we would like to find a Latin hypercube such that for any $j_1 < j_2$, X_{j_1} and X_{j_2} are as orthogonal as possible. Rather than working with the entire matrix $(X_{j_1})^t X_{j_2}$, it is convenient to deal with a single numerical value that is able to assess the degree of orthogonality of the two matrices, X_{j_1} and X_{j_2}. The idea of canonical correlation will be employed to address this issue. Full details are given below.

2.2. Polynomial canonical correlation

We first introduce canonical correlation. For a detailed account, the reader can refer to, for example, Jobson (1992), vol. II, pp 181-190. Given two sets of vectors $U = (U_1, \ldots, U_p)$ and $V = (V_1, \ldots, V_q)$, where U_i and V_j are all vectors in R^n, the canonical correlation is the maximal correlation between linear combinations of U and linear combinations of V. That is, the canonical correlation of U and V is defined to be

$$\rho(U, V) = \max_{\alpha_1, \alpha_2} \mathrm{Cov}(\alpha_1^t U, \alpha_2^t V)/[\mathrm{Var}(\alpha_1^t U)]^{1/2}[\mathrm{Var}(\alpha_2^t V)]^{1/2}.$$

It is clear that $\rho(U, V) = 0$ implies that for any α_1, α_2, $\alpha_1^t U$ and $\alpha_2^t V$ are uncorrelated. Let S_{UV} be the covariance matrix of U and V, and S_{UU}, S_{VU} and S_{VV} are similarly defined. Then there is a simple formula for calculating the $\rho(U, V)$. In fact, $\rho(U, V)$ is the square root of the largest eigenvalue of the matrix (Jobson 1992))

$$S_{UU}^{-1} S_{UV} S_{VV}^{-1} S_{VU}. \tag{4}$$

Now for two vectors $U_1 = (u_1, \ldots, u_n)^t$ and $V_1 = (v_1, \ldots, v_n)^t$, let $U_j = (u_1^j, \ldots, u_n^j)^t$ and $V_j = (v_1^j, \ldots, v_n^j)^t$, for $j = 1, \ldots, q$. Then we refer to the canonical correlation between $U = (U_1, \ldots, U_q)$ and $V = (V_1, \ldots, V_q)$ as the polynomial canonical correlation of order q between the two vectors U_1 and V_1.

Now consider the model in (2) and the full design matrix in (3). It is evident that if the polynomial canonical correlation of order q between the two vectors $x_{j_1} = (x_{1j_1}, \ldots, x_{nj_1})^t$ and $x_{j_2} = (x_{1j_2}, \ldots, x_{nj_2})^t$ is zero, then $(X_{j_1})^t X_{j_2} = \mathbf{0}$. Thus it is desirable to find a Latin hypercube in which for any two columns, the polynomial canonical correlation of order q is minimized. It is the objective of this paper to develop an algorithm to reduce the polynomial canonical correlation of any pair of columns for a Latin hypercube design. Although the algorithm naturally generalizes to polynomial canonical correlation of any order, we will henceforth be concerned with polynomial canonical correlation of order 2, referred to as quadratic canonical correlation. The notion of (ordinary) correlation is in line with polynomial canonical correlation of order 1. In fact, for two vectors the polynomial canonical correlation of order 1, henceforth referred to as linear canonical correlation, is the absolute value of the (ordinary) correlation.

It is worth noting that though we consider the model where main effects are represented by polynomials, the idea also applies to the case where each main effect is a linear combination of other functions of the corresponding variable.

3. An Algorithm for Reducing Quadratic Canonical Correlations

3.1. The algorithm

Iman and Conover (1982) developed a method for reducing correlations of a Latin hypercube using the Cholesky decomposition. Owen (1994) proposed another method for this purpose using the Gram-Schmidt orthogonalization and showed that a considerable improvement can be achieved by his method. Here we describe Owen's algorithm, based on which we develop our algorithm for reducing the quadratic canonical correlations of a Latin hypercube. Let rank(x) be the ranks of the components of the vector x. (Here "ranks" takes the meaning as rank statistics.) The notation *takeout*(y, x) denotes the vector of residuals from the simple linear regression $y = \beta_0 + \beta_1 x + \epsilon$. Then for a given initial Latin hypercube $L = (l_1, \ldots, l_m)$, Owen's algorithm proceeds by alternating between the forward and backward steps in the following. The forward step is

• for $j = 1, \ldots, m-1$, and for $k = j+1, \ldots, m$, set $l_k =$rank(takeout(l_k, l_j)).

The backward step is

• for $j = m, \ldots, 2$, and for $k = j-1, \ldots, 1$, set $l_k =$rank(takeout(l_k, l_j)).

The central idea here is that for fixed j, k, we want to update l_k such that the new l_k has a small correlation with l_j. It is clear that $r_k =$ takeout(l_k, l_j) has zero correlation with l_j. As each column of a Latin hypercube must be a permutation of $1, \ldots, n$, and the vector r_k does not satisfy this requirement, we take $l_k =$ rank(r_k). The correlation between $l_k =$ rank(r_k) and l_j should be small.

The vector of residuals from the regression $y = \beta_0 + \beta_1 x + \beta_2 x^2 + \epsilon$ is denoted by $\mathrm{Res}(y, x)$. Using the idea of Owen's algorithm, we propose our algorithm for reducing the quadratic canonical correlations of a Latin hypercubes as follows.

Algorithm. For a given initial Latin hypercube $L = (l_1, \ldots, l_m)$, the algorithm proceeds by alternating between the forward and backward steps, where the forward step is

• for $j = 1, \ldots, m-1$, and for $k = j+1, \ldots, m$, set l_k =rank($\mathrm{Res}(l_k, l_j)$),

and the backward step is

• for $j = m, \ldots, 2$, and for $k = j-1, \ldots, 1$, set l_k =rank($\mathrm{Res}(l_k, l_j)$).

3.2. Some remarks on our algorithm

Obviously, the linear canonical correlation of two vectors does not exceed their quadratic canonical correlation. Therefore, if we reduce the quadratic canonical correlations of a Latin hypercube design, then its linear canonical correlations will automatically be controlled. Thus using our algorithm instead of Owen's is advantageous, at least in principle. Furthermore, our algorithm improves on Owen's only at a little extra computing cost. To see this, consider the two algorithms for fixed j and k. Computing the residuals from the regression $y = \beta_0 + \beta_1 x + \epsilon$ and from the regression $y = \beta_0 + \beta_1 x + \beta_2 x^2 + \epsilon$ requires $2n$ and $4n$ operations, respectively, which are both of order $O(n)$. This is so because for either regression the matrix $(X^t X)^{-1}$ in the least squares solution, $\hat{\beta} = (X^t X)^{-1} X^t y$, is the same for different j and k, depending only on n, and thus can be computed in advance. (This will be explained below.) However, either algorithm calls the subroutine "rank" requiring $O(n \log n)$ comparisons. Thus the cost for computing the ranks is higher than computing the residuals for whichever regression. For moderately large n, the difference in cost for computing the residuals from the two regressions, relative to the cost for computing the ranks, is negligible.

The actual implementation of our algorithm is rather easy. A subroutine "rank" is written to compute the ranks for a vector. As mentioned in the last paragraph, before starting the forward and backward steps, the computer code first calculates the matrix $(X^t X)^{-1}$ in the least squares solution, $\hat{\beta} = (X^t X)^{-1} X^t y$, which will be used throughout the progression of the algorithm. For the regression $y = \beta_0 + \beta_1 x + \beta_2 x^2 + \epsilon$ in our algorithm, we have

$$X^t X = \begin{bmatrix} n & \sum_{i=1}^n x_i & \sum_{i=1}^n x_i^2 \\ \sum_{i=1}^n x_i & \sum_{i=1}^n x_i^2 & \sum_{i=1}^n x_i^3 \\ \sum_{i=1}^n x_i^2 & \sum_{i=1}^n x_i^3 & \sum_{i=1}^n x_i^4 \end{bmatrix}.$$

Since $x_1, \ldots, x_n$ is a permutation of $1, \ldots, n$, we have $\sum_{i=1}^n x_i = \sum_{i=1}^n i = n(n+1)/2$, $\sum_{i=1}^n x_i^2 = \sum_{i=1}^n i^2 = n(n+1)(2n+1)/6$, and so on. This illustrates

why we can calculate $(X^tX)^{-1}$ in advance. The computer code is written in C programming language, and is available to the reader upon request.

4. Performance of the Algorithm

In this section, we study the performance of our algorithm given in Section 3. To do this, we need a criterion for evaluating a Latin hypercube. For a Latin hypercube with m columns $L = (l_1, \ldots, l_m)$, denote the quadratic canonical correlation between column i and column j by $\rho_{ij}(L)$. Since $q = 2$, the matrix in (4) of Section 2 is a square matrix of order 2. Therefore, its largest eigenvalue and hence $\rho_{ij}(L)$ can easily be calculated. A sensible criterion for assessing L is

$$\rho(L) = \sum_{1 \le i < j \le m} \rho_{ij}(L). \tag{5}$$

More conservatively, we can also use the criterion $\phi(L) = \max_{i<j} \rho_{ij}(L)$. We are aiming at selecting a Latin hypercube with small ρ or ϕ value. But in the reminder of the paper, we only use $\rho(L)$ in (5) to examine the performance of the algorithm.

4.1. Convergence of the algorithm

Theoretical investigation of the convergence properties of the algorithm appears difficult and perhaps unrealistic. Instead, we examine the algorithm empirically here. We have experimented extensively with the algorithm and the following empirical behaviours have been observed. For convenience, the totality of a forward step and its immediately following backward step is referred to as one iteration. For an initial Latin hypercube, consider the sequence of designs generated by the algorithm. After some iterations, we have found that one of the following three situations must occur: (i) the same design always shows up, which is the case especially when m is small relative to n, (ii) the ρ values of the sequence of designs stabilize and fluctuate around a certain value, and (iii) two designs show up alternatingly for each forward and backward step. However, situation (iii) rarely occurs. We will use the word "stabilize" to describe all the three situations. In any case, the ρ values drop rapidly in the first several iterations and almost always stabilize within ten iterations. Once the ρ values have stabilized, we can then stop the algorithm, and pick the design with the smallest ρ value. Typical behaviours of the sequence are graphically represented in Figures 1 and 2, which correspond to situations (i) and (ii), respectively.

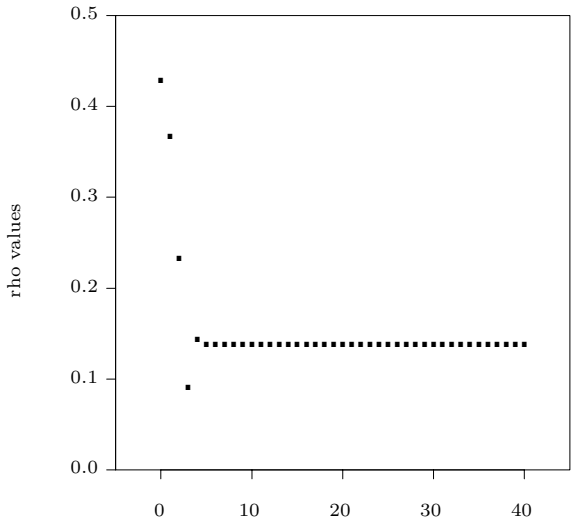


Figure 1. The ρ values of the sequence of designs generated by the algorithm for $n = 20$ and $m = 3$. In the plot, $\rho(0)$ is the ρ value of the initial design, chosen at random and $\rho(2k-1)$ and $\rho(2k)$ are the ρ values of the designs given by the forward and backward steps in the kth iteration, respectively.

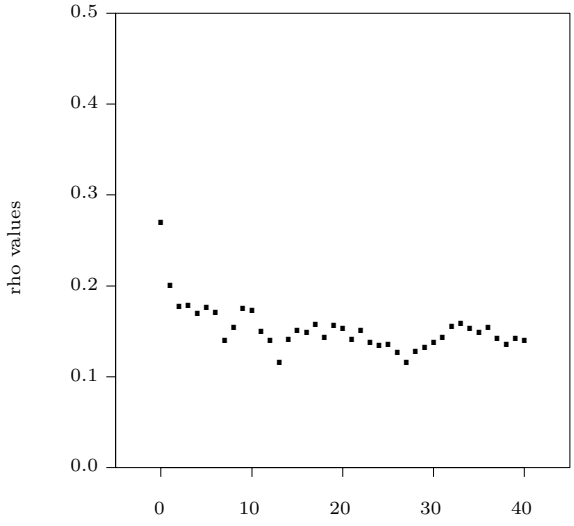


Figure 2. The ρ values of the sequence of designs generated by the algorithm for $n = 40$ and $m = 10$. In the plot, the sequence $\rho(k)$, $k = 0, 1, \ldots, 40$ is similarly defined as in Figure 1.

We note from Figure 1 that even for situation (i) the design with the smallest ρ value is not necessarily the limiting design.

4.2. Effect of initial designs

We have also studied empirically the effect of initial designs on the results of the algorithm. To this end, several random initial Latin hypercubes are chosen for fixed n and m, and the algorithm is then run. We have found that for small n and m, initial designs have a large effect on the results. As n and m are getting larger, the effect of using different initial designs tends to diminish. Figures 3 and 4 illustrate this phenomenon.

Thus it is advisable that several random initial designs be chosen for a small problem, and then take the design with the smallest ρ value from the pool of good designs given by different initial designs. This discovery is encouraging rather than disturbing because we can afford to run the algorithm several times for small values of n and m. For problems of small size, an extra loop will enable the algorithm to cope with several random initial designs, provided that for each initial design we run the algorithm for some specified number I of iterations. Denote the number of initial designs that are to be used by P. Then the algorithm can be modified to

1. for $p = 1, \ldots, P$, do steps 2 through 4,
2. obtain an initial Latin hypercube randomly,
3. run the algorithm I iterations,
4. obtain the design L_p with the smallest ρ value for pth initial design,
5. finally, select the design L with the smallest ρ value from the ρ designs $L_1, \ldots, L_p$.

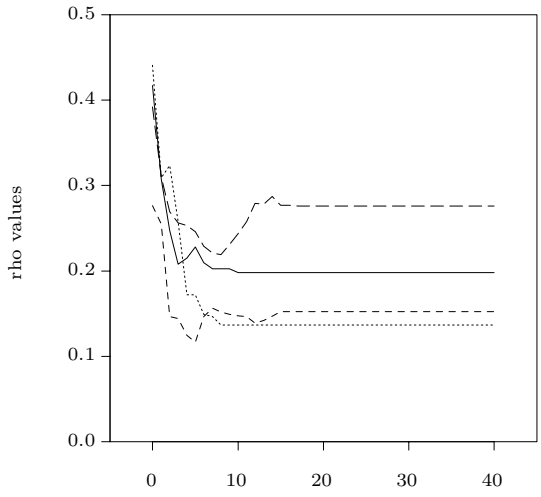


Figure 3. The ρ values of the sequence of designs generated by the algorithm for $n = 25$ and $m = 4$. Four initial Latin hypercubes are used here, which are all randomly chosen. The large effect of using different initial designs is evident.

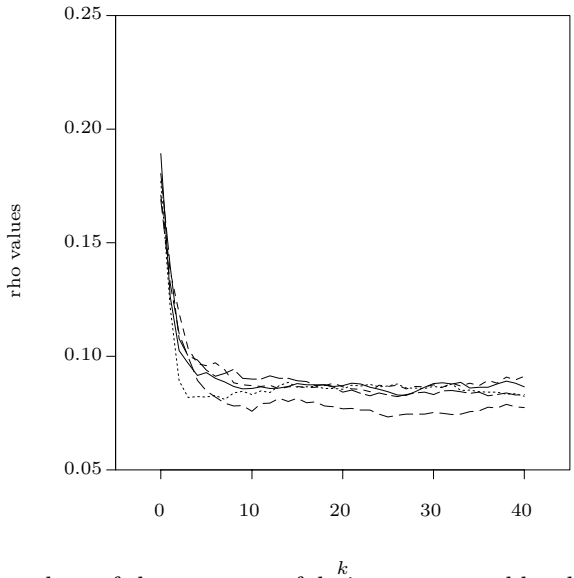


Figure 4. The ρ values of the sequence of designs generated by the algorithm for $n = 100$ and $m = 15$. Five initial Latin hypercubes are used here, which are all randomly chosen. The effect of using different initial designs is only marginal.

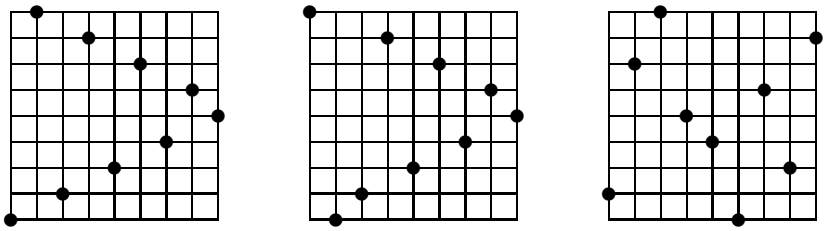
Here choosing the number of iterations to be 10 is sufficient since the ρ values of the sequence of designs almost always stablize within 10 iterations, as mentioned in Section 4.1.

5. Concluding Remarks

Although the rationale for reducing the quadratic canonical correlations of a Latin hypercube is motivated by considering the main effect polynomial model in (1) of Section 2, using a Latin hypercube with small quadratic correlations for its pairs of columns in numerical integration is also advantageous. This can be demonstrated along the same line as Owen (1994) did for the case of linear canonical correlation. Some details are given in Appendix A.

As briefly commented in Section 3, our algorithm is able to control the linear canonical correlations of a Latin hypercube while reducing its quadratic canonical correlations. However, Owen's algorithm does not have any control on the quadratic canonical correlations. Figure 5 illustrates this point. For the given initial design in Figure 5(a), Owen's algorithm produces a Latin hypercube in Figure 5(b), which has linear and quadratic canonical correlations 0.033 and 0.977, respectively. This can be very disturbing when quadratic canonical correlation is important for a problem. However, just in terms of reducing linear canonical correlations, our experience indicates that Owen's algorithm generally,

though not always, does better than our algorithm. This is intuitively understandable because our algorithm is designed to achieve more. So, if small linear canonical correlation is the only concern for a problem, using Owen's algorithm is recommended.



(a) (b) (c)

Figure 5. The same initial Latin hypercube, given in (a), is used for both Owen's and our algorithm. Owen's algorithm produces a Latin hypercube in (b), which has linear and quadratic canonical correlations 0.033 and 0.97, respectively. Our algorithm produces a Latin hypercube in (c) having linear and quadratic canonical correlations 0.033 and 0.196, respectively.

Table 1. The ρ values of some Latin hypercubes constructed using the algorithm. Those entries with $2m + 1 > n$ have been left blank. This is because for $q = 2$ in model (1), the condition $2m + 1 \leq n$ is necessary in order that the parameters in the model can be estimated.

	$n = 10$	$n = 20$	$n = 50$	$n = 100$	$n = 200$	$n = 500$
$m = 2$	0.015	0.010	0.005	0.004	0.003	0.002
$m = 3$	0.066	0.039	0.022	0.012	0.008	0.006
$m = 4$	0.154	0.068	0.035	0.020	0.018	0.016
$m = 5$		0.109	0.055	0.035	0.021	0.019
$m = 7$		0.172	0.066	0.049	0.033	0.024
$m = 9$		0.235	0.089	0.060	0.042	0.032
$m = 12$			0.108	0.068	0.049	0.033
$m = 15$			0.133	0.069	0.050	0.034
$m = 19$			0.157	0.076	0.054	0.035
$m = 23$			0.170	0.086	0.055	0.036
$m = 28$				0.099	0.057	0.037
$m = 35$				0.112	0.061	0.037
$m = 50$					0.069	0.037
$m = 70$					0.080	0.038
$m = 100$						0.040
$m = 150$						0.048

We have constructed a list of Latin hypercubes using our algorithm, with n ranging from 10 to 500, and m from 2 to 150. Table 1 gives some typical

values of n and m for which Latin hypercubes have been constructed. In the table, for each entry we also provide the corresponding ρ value, which is the average of the quadratic canonical correlations of all pairs of columns of a design. Several initial designs have been used for small n and m in the table. Table 1 shows the capability of the algorithm. Finally, we remark that on an IBM RISC System/6000 workstation, one iteration of the algorithm runs approximately five seconds for $n = 100$ and $m = 20$, and 20 minutes for $n = 500$ and $m = 150$.

Acknowledgement

The paper is part of the author's Ph.D dissertation at the University of Waterloo under the supervision of Professor Jeff Wu. The author thanks Professor Jeff Wu for a suggestion on looking into the idea of nonparametric correlation that led to this paper. The author also thanks an anonymous referee whose comments have led to the improvement of the paper. The research is supported by the Natural Sciences and Engineering Council of Canada.

Appendix A: Numerical Integration Using Latin Hypercubes with Small Quadratic Correlations

Let $y = f(\mathbf{x})$, where $y \in R$ and $\mathbf{x} = (x_1, \ldots, x_m) \in R^m$. The objective here is to evaluate $\mu = \int_{[0,1]^m} f(\mathbf{x})d\mathbf{x}$. Suppose that $\mathbf{x}_1, \ldots, \mathbf{x}_n$ are drawn IID from Unif[0,1]^m. Then the estimate of μ given by Monte Carlo is $\bar{Y} = n^{-1}\sum_{i=1}^n f(\mathbf{x}_i)$. Here we look at the situation where the n points $\mathbf{x}_1, \ldots, \mathbf{x}_n$ are derived from the n rows of a Latin hypercube sample $L = (l_{ij})$ via

$$\mathbf{x}_i = (x_{i1}, \ldots, x_{im}) \text{ with } x_{ij} = (l_{ij} - 0.5)/n. \tag{A.1}$$

Let $u(t) = t - 1/2$ and $w(t) = t^2 - 1/3$. Note that $\int_0^1 u(t)dt = \int_0^1 w(t)dt = 0$. Now let $v(t) = w(t) - cu(t)$ with $c = \int_0^1 w(t)u(t)dt / \int_0^1 u^2(t)dt$. We see that $\int_0^1 v(t)dt = 0$ and that $\int_0^1 u(t)v(t)dt = 0$. Let $u_1(t) = u(t)/[\int_0^1 u^2(t)dt]^{1/2}$ and $v_1(t) = v(t)/[\int_0^1 v^2(t)dt]^{1/2}$. Then we have $\int_0^1 u_1^2(t)dt = \int_0^1 v_1^2(t)dt = 1$. For notational brevity, we again use, in what follows, $u(t)$ to denote $u_1(t)$ and $v(t)$ to denote $v_1(t)$. Define the residual function $r(\mathbf{x})$ via

$$f(\mathbf{x}) = \mu + \sum_{j=1}^m \alpha_j(x_j) + \sum_{j<k} \beta_{jk}(x_j, x_k) + r(\mathbf{x}), \tag{A.2}$$

where

$$\alpha_j(x_j) = \int [f(\mathbf{x}) - \mu] \prod_{k \neq j} dx_k$$

$$\beta_{jk}(x_j, x_k) = a_{jk}u(x_j)u(x_k) + b_{jk}u(x_j)v(x_k) + c_{jk}v(x_j)u(x_k) + d_{jk}v(x_j)v(x_k) \tag{A.3}$$

with

$$a_{jk} = \int [f(\mathbf{x}) - \mu] u(x_j) u(x_k) d\mathbf{x}, \qquad b_{jk} = \int [f(\mathbf{x}) - \mu] u(x_j) v(x_k) d\mathbf{x}$$
$$c_{jk} = \int [f(\mathbf{x}) - \mu] v(x_j) u(x_k) d\mathbf{x}, \qquad d_{jk} = \int [f(\mathbf{x}) - \mu] v(x_j) v(x_k) d\mathbf{x}.$$

We note that all the terms on the right hand side of equation (A.2) are orthogonal. For each $\beta_{jk}(x_j, x_k)$, the four terms on the right hand side of equation (A.3) are also orthogonal. Combining (A.1), (A.2), and (A.3), we obtain

$$\bar{Y} = \mu + \sum_{j=1}^{m} n^{-1} \sum_{j=1}^{n} \alpha_j(x_{ij}) + \sum_{j<k} a_{jk} A_{jk} + \sum_{j<k} (b_{jk} B_{jk} + c_{jk} C_{jk} + d_{jk} D_{jk}) + n^{-1} \sum_{j=1}^{n} r(\mathbf{x}_i), \tag{A.4}$$

where

$$A_{jk} = n^{-1} \sum_{j=1}^{n} u(x_{ij}) u(x_{ik}) \quad B_{jk} = n^{-1} \sum_{i=1}^{n} u(x_{ij}) v(x_{ik})$$
$$C_{jk} = n^{-1} \sum_{j=1}^{n} v(x_{ij}) u(x_{ik}) \quad D_{jk} = n^{-1} \sum_{i=1}^{n} v(x_{ij}) v(x_{ik}).$$

Note that the absolute value $|A_{jk}|$ of A_{jk} is essentially the linear correlation between column j and column k of the Latin hypercube $L = (l_{ij})$. As argued in Owen (1994), if A_{jk} is $o_p(n^{-1/2})$, then $\sum_{j<k} a_{jk} A_{jk}$ in (A.4) is filtered out. For the same reason, if the quadratic correlations among the columns of L are $o_p(n^{-1/2})$, then $\sum_{j<k} (a_{jk} A_{jk} + b_{jk} B_{jk} + c_{jk} C_{jk} + d_{jk} D_{jk})$ in (A.4) is filtered out. This is because $|A_{jk}|$, and similarly $|B_{jk}|$, $|C_{jk}|$ and $|D_{jk}|$ cannot exceed the quadratic correlation between column j and column k of the Latin hypercube $L = (l_{ij})$.

References

Iman, R. L. and Conover, W. J. (1982). A distribution-free approach to inducing rank correlation among input variables. *Comm. Statist. Part B—Simulation Comput.* **11**, 311-334.

Johnson, M., Moore, L. and Ylvisaker, D. (1990). Minimax and maximin distance designs. *J. Statist. Plann. Inference* **26**, 131-148.

McKay, M. D., Conover, W. J. and Beckman, R. J. (1979). A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics* **21**, 239-245.

Morris, M. D. and Mitchell, T. J. (1992). Exploratory designs for computational experiments. Research Report, Oak Ridge National Laboratory.

Owen, A. (1994). Controlling correlations in Latin hypercube samples. *J. Amer. Statist. Assoc.* **89**, 1517-1522.

Patterson, H. D. (1954). The errors of lattice sampling. *J. Roy. Statist. Soc. Ser. B* **16**, 140-149.

Tang, B. (1993). OA-based Latin hypercubes. *J. Amer. Statist. Assoc.* **88**, 1392-1397.

Tang, B. (1994). A theorem for selecting OA-based Latin hypercubes using a distance criterion. *Comm. Statist. Theory Methods* **23**, 2047-2058.

Welch, W. J., Buck, R. J., Sacks, J., Wynn, H. P., Mitchell, T. J. and Morris, M. D. (1992). Screening, predicting, and computer experiments. *Technometrics* **34**, 15-25.

Department of Mathematical Sciences, The University of Memphis, Campus Box 526429, Memphis, TN 38152-6429, U.S.A.

E-mail: tangb@msci.memphis.edu

(Received October 1996; accepted August 1997)