Contents lists available at ScienceDirect

Computational Statistics and Data Analysis

journal homepage: www.elsevier.com/locate/csda

Uniform design over general input domains with applications to target region estimation in computer experiments

S.C. Chuang[a], Y.C. Hung[b,*]

[a] *Graduate Institute of Statistics, National Central University, Jhongli 32049, Taiwan*
[b] *Department of Statistics, National Chengchi University, Taipei 11605, Taiwan*

A R T I C L E   I N F O

Article history:
Received 18 November 2008
Received in revised form 3 June 2009
Accepted 14 August 2009
Available online 21 August 2009

A B S T R A C T

The power of uniform design (UD) has received great attention in the area of computer experiments over the last two decades. However, when conducting a typical computer experiment, one finds many non-rectangular types of input domains on which traditional UD methods cannot be adequately applied. In this study, we propose a new UD method that is suitable for any type of design area. For practical implementation, we develop an efficient algorithm to construct a so-called nearly uniform design (NUD) and show that it approximates very well the UD solution for small sizes of experiment. By utilizing the proposed UD method, we also develop a methodology for estimating the target region of computer experiments. The methodology is sequential and aims to (i) provide adaptive models that predict well the output measures related to the experimental target; and (ii) minimize the number of experimental trials. Finally, we illustrate the developed methodology on various examples and show that, given the same experimental budget, it outperforms other approaches in estimating the prespecified target region of computer experiments.

1. Introduction and motivating examples

We first illustrate two examples from stochastic processing networks that are used to motivate this research.

Motivating Example 1: Consider a queueing system comprised of K infinite capacity first-in–first-out (FIFO) queues in parallel, and each queue corresponds to a different class of job traffic. The class k jobs arrive according to some process with mean rate x_k and are queued up for service, $k = 1, \ldots, K$. At any point in time, the system can be in one of S service modes. When service mode s is used, the first job of queue k receives service with mean rate μ_{sk}, $k = 1, \ldots, K$. Therefore, mode s is associated with the service rate vector $U_s = (\mu_{s1}, \ldots, \mu_{sK})$, $s = 1, \ldots, S$. This queueing system is known as the Switched Processing System (SPS), which captures the essence of a fundamental resource allocation problem in many modern systems involving heterogeneous processors and multiple classes of job traffic flows (e.g., parallel computing, wireless networking, call centers, flexible manufacturing, etc.).

It has been shown in the literature (Armony and Bambos, 2003; Hung and Michailidis, 2008) that, in order to achieve system stability (e.g. the long-term input rate for each queue is equivalent to its long-term service rate), the input rate vector $x = (x_1, \ldots, x_K)$ must lie within the following region:

$$D = \left\{ x \in \mathbb{R}_+^K : x_k < \sum_{s=1}^{S} \omega_s \mu_{sk}, \text{ for all } k = 1, \ldots, K \right\},$$

* Corresponding author.
E-mail address: hungy@stat.ncu.edu.tw (Y.C. Hung).

0167-9473/$ – see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.csda.2009.08.008

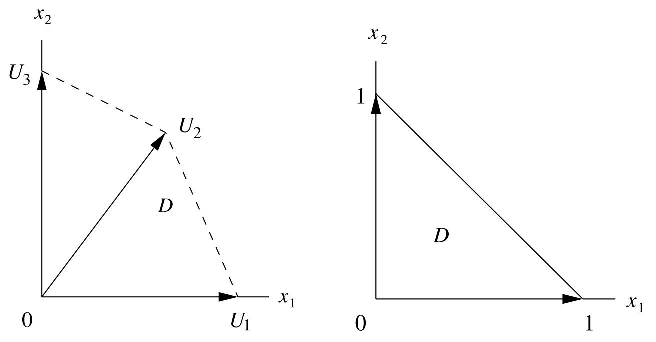

Fig. 1. (Left panel): The domain D of the input rate vectors x for a 2-queue system with three service rate vectors $U_1 = (3, 0), U_2 = (2, 3)$, and $U_3 = (0, 4)$. (Right panel): The domain D of the weight vectors x (in the MaxProduct policy) for a 3-queue system (where $K = 3$ and $x_3 = 1 - x_1 - x_2$).

where $0 \le \omega_s \le 1$ for all $s = 1, \ldots, S$, and $\sum_{s=1}^{S} \omega_s = 1$. Note that the linear constraints in D describe that the long-term input rate of each queue k cannot exceed its long-term service rate. Further, it can be shown that D is a *convex hull* generated by all service rate vectors U_s and their projections on the axes. An example of such D is shown in the left panel of Fig. 1. In addition, a service-mode allocation policy is called a throughput-maximizing policy if it can stabilize the system for all input rate vectors $x \in D$. A natural question followed is, under a particular throughput-maximizing policy π, how the input rate vector x affects the performance measures of interest (such as delay, backlog, etc). In this case, $x_1, \ldots, x_K$ are treated as input factors, while the input domain D is a convex polygon.

Motivating Example 2: Let us consider the same queueing system introduced in Example 1. Suppose now all the input rates are fixed (but still inside the region D), and we consider a throughput-maximizing policy called the "MaxProduct" (Armony and Bambos, 2003; Hung and Chang, 2008; Hung and Michailidis, 2008). This policy employs service mode s^* at time t if

$$s^* = \arg\max_{s=1,\ldots,S} \sum_{k=1}^{K} x_k Y_k(t) \mu_{sk},$$

where $Y_k(t)$ represents the state of queue k (either the number of jobs or the workload) at time t, and x_k is any chosen positive queue weight. Another interesting question is, how the choice of the queue weight vector $x = (x_1, \ldots, x_K)$ affects the performance measures of interest. It has been shown that the weight vector x affects the MaxProduct policy only through its directions in $\mathbb{R}_+^K$ (Hung and Michailidis, 2008). Therefore, it suffices to consider the vectors x satisfying that $x_1 + \cdots + x_K = 1$. This constraint can be further reduced to $x_1 + \cdots + x_{K-1} \le 1$, which clearly represents a *simplex* in $\mathbb{R}_+^{K-1}$. In this case, the queue weights $x_1, \ldots, x_{K-1}$ are treated as input factors, while the input domain D is a $(K - 1)$-simplex. An example of such D is shown in the right panel of Fig. 1.

Note that due to the complex structure of dynamics, it is often hard to obtain the relationship between the input factors and the response measures of interest for such systems. Therefore, this type of problems are often examined through computer simulations (Hung et al., 2003; Hung and Michailidis, 2008). However, simulation of such complex systems is expensive in terms of CPU time and the requirement of simulation resources. An natural question is then how one can obtain a comprehensive understanding of system's performance by performing the minimum possible number of simulation trials. This has been a challenging task in the area of design and analysis of computer experiments (DACE).

The uniform design (UD) was first proposed by Fang and Wang (Fang, 1980; Wang and Fang, 1981) and has been widely used in computer experiments over the last two decades (Fang and Lin, 2003). Its basic idea is to seek input points to be uniformly scattered on the input domain so that the relationships between the response(s) and the input factors can be explored using a reasonable number of experimental trials. Traditional methods have provided solutions to UD for the experiments without restrictions (Fang and Wang, 1994), i.e., the design area is or can be reasonably transformed into a unit hypercube $[0, 1]^K$. However, for non-rectangular types of design areas (such as the input areas shown in the two motivating examples), how to best perform the UD is not fully discussed in literature. The work done by Fang et al. (1999b) is closely related to the examples introduced above, wherein they proposed a "simplex method" to perform a stochastic representation of UD over convex polyhedrons. The shortcomings for this type of Monte Carlo methods are: (i) the represented UD has larger variations; and (ii) they are shown to have relatively low efficiency on approximating the output measures of interest (Fang and Wang, 1994).

In this study, we propose a new UD method that is suitable for any types of design area under the framework of the so-called number-theoretic method (NTM). The proposed UD method has an important feature that the optimal design is invariant under coordinate rotations and can be properly extended so that lower-dimensional uniformity is also considered (see Section 2 for details). For practical applications, we also develop a methodology to estimate the target region of computer experiments by utilizing the proposed UD method. Note that the target region here represents a subset of the input domain in which the experimental output measure(s) of interest is desired to be produced. For example, let us consider the queueing system introduced in Example 1, where one can specify a target region for the input rate vectors so that the average delay (i.e., the average time waiting until service is first provided) of jobs in each queue does not exceed a prespecified quantity. By keeping the input rates in the target region, the system then provides a commitment to a certain level of quality service. Another example can be found in the recent work done by Ranjan et al. (2008), where a sequential design based on the Gaussian stochastic process (GASP) model was proposed to estimate a contour of a complex computer code. Their work can be viewed as a special case of our problem, for which the experiment has merely one output measure and the contour line corresponds to the boundary of the target region. Analogous to the method proposed by Ranjan et al. (2008), our methodology is also sequential and aims to (i) provide adaptive models that predict well the output measures related to the experimental target; and (ii) minimize the number of experimental trials. In addition, it has the following advantages: (i) it can easily handle the experiments with a large number of input factors; (ii) it can handle the target region comprised of multiple output measures.

The rest of the paper is organized as follows. In Section 2, the proposed UD method and its extension called weighted UD are introduced. For practical purposes, an efficient algorithm is developed to construct a so-called nearly uniform design (NUD) and shown to approximate well the UD solution. In Section 3, the methodology for estimating the target region of computer experiments is introduced. Note that the methodology is comprised of two main components: (i) design; and (ii) fitting response models. For component (i), the proposed UD method with sequentially updated weight functions is utilized; while for component (ii), the GASP model and another modelling technique called support vector regression (SVR) are employed. In Section 4, the proposed methodology is illustrated on two examples. The numerical results show that, given the same experimental budget, it outperforms other approaches in estimating the prespecified target regions. Some concluding remarks are drawn in Section 5.

2. Uniform design over general input domains

The uniform design (UD), first proposed by Wang and Fang in 1980, is one of the space filling designs (Box and Draper, 1987; Cheng and Li, 1995; Hickernell, 1999; Wu and Hamada, 2000) that seeks input points to be uniformly scattered on the input region D. Its basic idea is introduced in the following. Suppose we would like to choose a set of n experiment points $\mathcal{P} = \{p_1, \dots, p_n\}$ that are uniformly scattered on an *identifiable* input domain D, $D \subset \mathbb{R}^K$. Let M be a measure of uniformity of $\mathcal{P}$ such that smaller M corresponds to better uniformity. Let $Z(n)$ be the set of all possible sets $\{p_1, \dots, p_n\}$ on D. A set $\mathcal{P}^* \in Z(n)$ is called a uniform design if it has the minimum value of M over $Z(n)$, i.e.,

$$M(\mathcal{P}^*) = \min_{\mathcal{P}\in Z(n)} M(\mathcal{P}). \tag{1}$$

The popular measures of uniformity are discrepancy (with various modified versions), dispersion, mean square error, and sample moments (Fang et al., 2000; Hickernell, 1998; Fang and Wang, 1994). However, most of these methods for UD are developed under the assumption that the experimental domain D can be reasonably transformed into a unit cube (e.g. rectangles). Motivated by the examples introduced in Section 1, in this study we propose a UD method that is suitable for experiments with any types of input domain.

2.1. A new measure of uniformity: Central composite discrepancy

We first introduce some notations that are necessary for constructing a new measure of uniformity called "central composite discrepancy". For any point $x \in \mathbb{R}$, denote the set

$$x^{(i)} = \{r \in \mathbb{R} : x + a_i < r \le x + a_{i+1}\}, \quad i = 0, 1, \dots, m-1, \tag{2}$$

where $a_0 = -\infty$, $a_m = \infty$, $a_1 < a_2 < \cdots < a_{m-1}$, and $a_j = 0$ for some $1 \le j \le m-1$. Thus, the real line is divided into m parts at the point x. With the division on each coordinate of a given point $x = (x_1, \dots, x_K) \in D \subset \mathbb{R}^K$, the input domain D is decomposed into (at most) m^K subregions, where the kth subregion is denoted by $D_k(x) = \{x_1^{(i_1)} \times \cdots \times x_K^{(i_K)}\} \bigcap D$, and $(i_1, \dots, i_K)$ is the base-m display of integer $k-1$. The examples of such a decomposition for a two-dimensional convex polygon D are shown in Fig. 2.

Consider a set of n experiment points $\mathcal{P} = \{p_1, \dots, p_n\}$ on D and let

$$N(D_k(x), \mathcal{P}) = \sum_{i=1}^{n} I\{p_i \in D_k(x)\}, \tag{3}$$

which represents the number of points allocated in the subregion $D_k(x)$ given by the decomposition of D at x, $x \in D$. The *central composite discrepancy* is defined as

$$CCD_p(n, \mathcal{P}) = \left\{ \frac{1}{v(D)} \int_D \frac{1}{m^K} \sum_{k=1}^{m^K} \left| \frac{N(D_k(x), \mathcal{P})}{n} - \frac{v(D_k(x))}{v(D)} \right|^p \mathrm{d}x \right\}^{1/p}, \tag{4}$$

where $p > 0$, $v(D)$ and $v(D_k(x))$ denote the volume of D and $D_k(x)$, respectively. The optimal allocation of the n experiment points is the set that minimizes $CCD_p(n, \mathcal{P})$, that is,

$$\mathcal{P}^* = \arg\min_{\mathcal{P}\in Z(n)} CCD_p(n, \mathcal{P}). \tag{5}$$

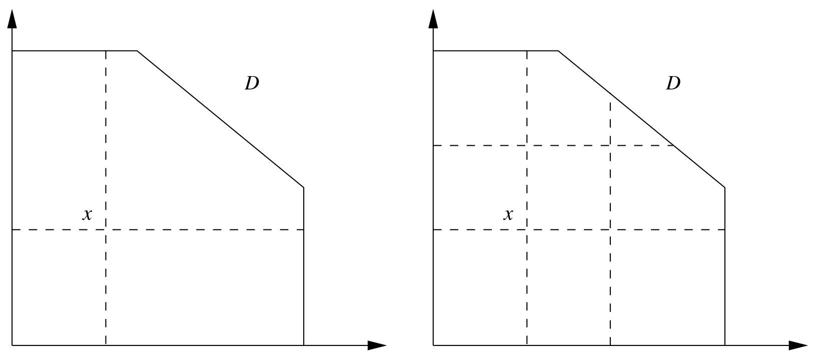

Fig. 2. (Left panel): The decomposition of D at x with $m = 2$ and $a_1 = 0$. (Right panel): The decomposition of D at x with $m = 3$ and $a_1 = 0$.

Note that the goal of placing the quantity $1/v(D)$ in (4) is to rescale the input domain D so that it has volume one. However, this does not affect the optimal solution $\mathcal{P}^*$ for any given D.

The basic idea of the proposed central composite discrepancy is that each point x in D is treated as a "center", and uniformity is measured over all decomposed subregions around it. In the special case when D is a hyper-rectangle, $m = 2$, $a_1 = 0$, and $p = 1$, it is equivalent to the so-called "symmetrical L_1-discrepancy" (Ma, 1997a). Note that the central composite discrepancy and the symmetrical discrepancy both share the same intuition that the optimal design is invariant under coordinate rotation. However, the former can be applied to the entire class of input domains while the latter can merely be applied to hyper-rectangles.

Remark: In addition to the central composite discrepancy defined in (4), one can also consider other measures of uniformity such as

$$M(n, \mathcal{P}) = \sup_{x \in D} \sum_{k=1}^{m^K} \left| \frac{N(D_k(x), \mathcal{P})}{n} - \frac{v(D_k(x))}{v(D)} \right|$$

or

$$M(n, \mathcal{P}) = \sum_{k=1}^{m^K} \sup_{x \in D} \left| \frac{N(D_k(x), \mathcal{P})}{n} - \frac{v(D_k(x))}{v(D)} \right|.$$

2.2. The weighted uniform design

Let $f(x)$ be a continuous function defined on $D, f(x) > 0$ for all $x \in D$ and $\int_D f(x)\mathrm{d}x = 1$. How do we find a set of n points $\mathcal{P} = \{p_1, \ldots, p_n\}$ on D so that they have a "good representation" for $f(x)$? By utilizing the measure of uniformity defined in (4), we next define the *weighted central composite discrepancy* by

$$WCCD_{f,p}(n, \mathcal{P}) = \left\{ \frac{1}{v(D)} \int_D \frac{1}{m^K} \sum_{k=1}^{m^K} \left| \frac{N(D_k(x), \mathcal{P})}{n} - F(D_k(x)) \right|^p \mathrm{d}x \right\}^{1/p}, \tag{6}$$

where $F(D_k(x)) = \int_{D_k(x)} f(x)\mathrm{d}x$ represents the proportion of points expected to be allocated on each subregion $D_k(x)$, $k = 1, \ldots, m^K$. Therefore, a good representation for $f(x)$ will be the set of points $\mathcal{P}^*$ that minimizes $WCCD_{f,p}(n, \mathcal{P})$.

Note that if $f(x)$ corresponds to a probability density function and p is chosen to be 1, then the quantity defined in (6) is a rotation-invariant version of the so-called "F-discrepancy" (Fang and Wang, 1994). However, the interpretation of the function $f(x)$ is not restricted here. In general, it can represent the "weight" (or "importance") of each point x in D—the larger the value of $f(x)$ is, the more important the point x is considered. Some examples of how to choose the function $f(x)$ in correspondence with the prespecified targets of experiment are shown later in Section 3.

2.3. Construction of nearly uniform designs

It is known that solving $\mathcal{P}^*$ is a NP hard problem as the number of allocated design points goes to infinity. In practice, a computationally more efficient way is to construct a so-called nearly uniform design (NUD) with a low measure of uniformity. Traditional techniques for constructing the NUDs are the good lattice point method and its modifications (Wang and Fang, 1981; Fang and Li, 1995; Ma, 1997b), the method based on searching only a subset of U-type designs (Fang and Hickernell, 1995), the construction methods based on Latin squares (Fang et al., 1999a) and orthogonal designs (Fang, 1995), the threshold accepting method based on U-type designs (Winker and Fang, 1998; Fang et al., 2001), the method by collapsing two uniform designs (Fang and Qin, 2003), and the cutting method (Ma and Fang, 2004). In order to deal with general types of input domain, here we utilize an efficient approach (called "switching algorithm") that has been widely used in design literature (Winker & Fang, 1998; Fang et al., 2001) and cluster analysis (e.g. K-means clustering, (Sharma, 1996)). The steps of the switching algorithm are summarized in the following.

The Switching Algorithm

Step 1: Superimpose N candidate grids $g_1, \ldots, g_N$ on the primary input domain and denote the new input domain by $D = \{g_1, \ldots, g_N\}$. Arbitrarily choose an initial design $\mathcal{P}^{(0)} = \{g_1, \ldots, g_n\}$ from D, set $i = 0$.
Step 2: Set $j = 1$ and $\mathcal{P}^{(i+1)} = \mathcal{P}^{(i)}$.
Step 3: Let $g^* = \arg\min_{g \in D \setminus \mathcal{P}^{(i+1)}} CCD_p(n, \{g\} \bigcup \mathcal{P}^{(i+1)} \setminus \{g_j\})$.
If $CCD_p(n, \{g^*\} \bigcup \mathcal{P}^{(i+1)} \setminus \{g_j\}) < CCD_p(n, \mathcal{P}^{(i+1)})$,
set $\mathcal{P}^{(i+1)} = \{g^*\} \bigcup \mathcal{P}^{(i+1)} \setminus \{g_j\}$.
Step 4: Set $j = j + 1$. If $j \le n$, go to Step 3; otherwise go to Step 5.
Step 5: If $\mathcal{P}^{(i+1)} \ne \mathcal{P}^{(i)}$, set $i = i + 1$ and go to Step 2; otherwise return $\mathcal{P}^{(i)}$.

Note that in Step 1, an initial design $\mathcal{P}^{(0)}$ of size n is arbitrarily placed. For example, one can choose $\mathcal{P}^{(0)}$ by utilizing the technique of *simple random sampling*. After that, the design $\mathcal{P}^{(i)}$ is iteratively updated by consecutively switching its design points with other candidate grids in D so as to reduce the proposed measure of uniformity (Step 2–Step 4). In Step 5, the design points are extracted when the measure of uniformity cannot be further reduced (i.e., no more switchings are needed to improve uniformity). Denote the resulting design by $\mathcal{P}^{(i*)}$, we next show that: (i) solving $\mathcal{P}^{(i*)}$ requires at most $O(N^{2+p})$ computations of $CCD_p(n, \mathcal{P})$; and (ii) the resulting $\mathcal{P}^{(i*)}$ approximates very well the optimal design $\mathcal{P}^*$.

Fact 1. *For any given* $\mathcal{P} \subset D = \{g_1, \ldots, g_N\}$, $0 \le [CCD_p(n, \mathcal{P})]^p \le 1$.

Fact 2. $CCD_p(n, \mathcal{P}^{(i)})$ *is a non-increasing function of* i.

Note that the results of Facts 1 and 2 are straightforward, so the proofs are omitted.

Fact 3. *If* p *is a positive integer and* $\mathcal{P}^{(i+1)} \ne \mathcal{P}^{(i)}$ *in Step 5, then*

$$[CCD_p(n, \mathcal{P}^{(i)})]^p - [CCD_p(n, \mathcal{P}^{(i+1)})]^p \ge \frac{1}{n^p N^{1+p} m^K}.$$

Proof. Define

$$W(n, \mathcal{P}) = \sum_{g \in D} \sum_{k=1}^{m^K} |N \cdot N(D_k(g), \mathcal{P}) - n \cdot N(D_k(g), D)|^p,$$

it is clear that $W(n, \mathcal{P})$ is a positive integer and by definition

$$[CCD_p(n, \mathcal{P}^{(i)})]^p - [CCD_p(n, \mathcal{P}^{(i+1)})]^p = \frac{W(n, \mathcal{P}^{(i)}) - W(n, \mathcal{P}^{(i+1)})}{n^p N^{1+p} m^K}.$$

The result then follows since by Fact 2 we know that $W(n, \mathcal{P}^{(i)}) - W(n, \mathcal{P}^{(i+1)}) \ge 1$ when $\mathcal{P}^{(i+1)} \ne \mathcal{P}^{(i)}$. □

Theorem 1. *For any positive integer* p*, the computation time of* $CCD_p(n, \mathcal{P}^{(i)})$ *in the switching algorithm is at most* $O(N^{2+p})$.

Proof. Note that to finish the update of each design $\mathcal{P}^{(i)}$, the required computation time of $CCD_p(n, \mathcal{P})$ is $n(N - n)$ (since there are n switchings needed to be checked and each switching requires $N - n$ computations of $CCD_p(n, \mathcal{P})$). In addition, from Facts 1–3 we know that $CCD_p(n, \mathcal{P}^{(i)})$ is a non-increasing function of i and $CCD_p(n, \mathcal{P}^{(0)})$ can be reduced at most $n^p N^{1+p} m^K$ times. These together imply the total computation time of $CCD_p(n, \mathcal{P})$ is bounded above by $n^p N^{1+p} m^K \cdot n(N - n)$, which can be represented as $O(N^{2+p})$. □

The result of Theorem 1 is quite essential from the perspective of computational efficiency. To see this, note that the computation time of $CCD_p(n, \mathcal{P})$ for finding the optimal design (based on exhaustive search) is clearly $\binom{N}{n} = O(N^n)$. However, since p is often chosen to be 1 or 2, the computation time can be dramatically reduced to $O(N^{2+p})$ by using the switching algorithm. To investigate how the resulting NUD approximates the true optimal design $\mathcal{P}^*$, we consider the unit square input domain on which 30 grids are superimposed (i.e., $N = 30$). The switching algorithm is then carried out for 100 times (with each initial design chosen by simple random sampling) and the "average" central composite discrepancy (with $p = 2$) of all resulting NUDs is computed. The results for different sizes of experiment (say $n = 1, \ldots, 15$) are shown in Fig. 3.

From Fig. 3, we see that the NUDs obtained from the switching algorithm approximate very well (in average sense) the optimal design for various sizes of experiment. In addition, the numerical results show that the switching algorithm is quite stable since the standard deviation of the resulting $CCD_2(n, \mathcal{P}^{(i*)})$ (based on 100 NUDs) is less than 2×10^{-3} for all n. For practical purposes, the CPU time for finding the optimal design (based on exhaustive search) and the average CPU time for

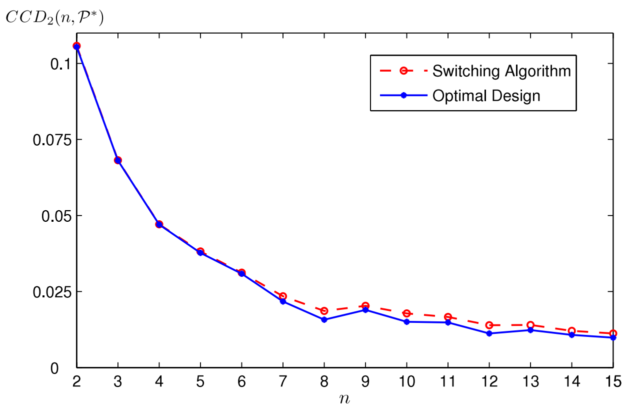

Fig. 3. The comparison of the optimal design and the NUD based on the switching algorithm for the input domain of unit square.

Table 1
The CPU time (in seconds) for finding the optimal design (based on exhaustive search) and the average CPU time (in seconds) for finding the NUD.

n	Optimal design	NUD	n	Optimal design	NUD	n	Optimal design	NUD
1	<0.001	–	6	34	0.289	11	4525	0.816
2	0.015	0.047	7	128	0.377	12	7592	0.878
3	0.172	0.100	8	384	0.446	13	11 183	1.112
4	1.235	0.145	9	1048	0.586	14	14 286	1.207
5	7	0.230	10	2342	0.686	15	19 627	1.356

finding the NUDs (based on 100 iterations of the switching algorithm) are attached in Table 1. As can be seen from Table 1, as n becomes larger, the CPU time for finding the NUD based on the switching algorithm becomes significantly smaller than that of finding the optimal design.

Remark: To improve the performance of nearly uniform design, one can superimpose coarse grids over the input domain and then utilize the optimal design as the initial design in the switching algorithm. However, such a setup will require some extra computational cost.

Remark: The optimal design and the NUDs were obtained based on the simulation trials that were executed on 2GHz Pentium 4 processors with 1GB of cache memory. Computer programs were all written in Fortran.

2.4. Discussion

In this section we discuss some other computational issues of the proposed UD method.

Low-dimensional Uniformity. It is noted that the proposed measure of uniformity can be modified so that uniformity over low-dimensional spaces is also taken into account. Let ψ be a non-empty subset of $\{1, \ldots, K\}$ and $|\psi|$ be the number of elements in ψ. For any given weight function $f(x)$, define $f_\psi = \int f(x)\mathrm{d}x_{-\psi}$, where $\mathrm{d}x_{-\psi} = \prod_{i\in\{1,\ldots,K\}\setminus\psi} \mathrm{d}x_i$. Thus, a more general measure of uniformity, called *projected central composite discrepancy*, can be defined as

$$PCCD_{\Psi,f,p}(n, \mathcal{P}) = \left\{ \frac{1}{|\Psi|} \sum_{\psi\in\Psi} [WCCD_{f_\psi,p}(n, \mathcal{P}_\psi)]^p \right\}^{1/p}, \tag{7}$$

where Ψ is the collection of all non-empty subsets of $\{1, \ldots, K\}$, and $\mathcal{P}_\psi$ is the projection of $\mathcal{P}$ to the subspace ψ. It is noted that similar ideas are used to construct the so-called "centered L_p-discrepancy" (Hickernell, 1998). However, the consideration of projections to low-dimensional spaces will require a large amount of computation, as can be seen from (7). Therefore, here we restrict our attention on the uniformity in the K-dimensional space, i.e., we choose $\Psi = \{\{1, \ldots, K\}\}$.

Choice of N. It is noted that if N is too small (i.e., the number of candidate grids is too small), then the resulting NUD may not be good enough. On the other hand, if N is large, then the switching algorithm may not be efficient. Therefore, to choose the value of N, one must take into account the tradeoff between design optimality and computational cost. Here we provide a guideline for choosing N: if K is small, then choose $N = n^K$; if K is large, then choose $N = nK$. Based on the guideline, an n^K factorial design is used for choosing the candidate grids in a small space, while the number of candidate grids is chosen to be proportional to the size of design in a large space.

When K **is Large.** Another computational issue is how to evaluate the summations in (4) and (6) when K is large. In practice, this can be done by considering the summation over a random subset of all decomposed subregions. For example, Eq. (4)

can be replaced by

$$\left\{ \frac{1}{v(D)} \int_D \frac{1}{L} \sum_{l=1}^{L} \left| \frac{N(D_{k_l}(x), \mathcal{P})}{n} - \frac{v(D_{k_l}(x))}{v(D)} \right|^p \mathrm{d}x \right\}^{1/p},$$

where $\{k_1, \dots, k_L\}$ is a random subset of $\{1, \dots, m^K\}$, and similarly for Eq. (6).

3. Methodology for target region estimation

Consider a typical computer experiment that can simultaneously produce outputs (or responses) $y_1(x), \dots, y_m(x)$, where $x = (x_1, \dots, x_K) \in D$ represents the vector of K input factors, and D is a general convex region. Suppose there are I output measures of interest, with each denoted by $z_i(x) = h_i(y_1(x), \dots, y_m(x))$, $i = 1, \dots, I$, and we would like to know how to control the input factors x so that these output measures satisfy

$$a_i \le z_i(x) \le b_i \quad \text{for all } i = 1, \dots, I, \tag{8}$$

where a_i and b_i are prespecified threshold values. Let us denote the "target region" for each output measure $z_i(x)$ by $T_i(x)$, where

$$T_i(x) = \{x \in D : a_i \le z_i(x) \le b_i\}, \quad i = 1, \dots, I. \tag{9}$$

Therefore, the "overall" target region of the experiment is defined as

$$T(x) = \bigcap_{i=1}^{I} T_i(x), \tag{10}$$

which is the set of input factors so that all desired outputs of interest can be produced.

Note that $T(x)$ represents a general form of target regions involving multiple responses of interest. However, the analytical solution for $T(x)$ is often hard to obtain due to the complexity of experiment. A naive approach is to perform a large number of experimental trials over the entire input domain D so that $T(x)$ can be estimated through an empirical study. However, such a procedure reveals to be infeasible due to a high cost of each trial (an important feature of computer experiments). Therefore, we propose an efficient methodology for estimating $T(x)$. The proposed methodology is sequential and comprised of two main components: (i) design and (ii) fitting response surfaces. For the part of design, the UD method introduced in Section 2 is utilized; while for fitting response models, the techniques based on Gaussian stochastic process (GASP) and support vector regression (SVR) are considered. Before we proceed, the following preliminaries are shown necessary.

3.1. Choosing the weight function $f(x)$ for UD

As we know, every input point x in D may provide different information for estimating the target region $T(x)$. For example, the input points near the boundary of the target region (denoted by $\bar{T}(x)$) intuitively can provide more useful information for estimating $T(x)$. This means that the local model fitting near $\bar{T}(x)$ is more attractive than the empirical model fitting over the entire input domain D. Therefore, the experimental design (allocation of input points) has to place much more weight on the input points near $\bar{T}(x)$. Our first step is to construct a function $f(x)$ which represents well the "importance" of each input point $x \in D$ with respect to $\bar{T}(x)$. After normalizing $f(x)$, we can treat it as a weight function so that the proposed measure of uniformity (6) can be used to select the input points for analysis.

We start by considering the experiment with merely a single output measure $z_i(x)$. For each input point $x \in D$, consider

$$f_i(x) = \max\left(\mathrm{e}^{-\beta_i(k)\frac{|z_i(x)-a_i|}{\sigma_{z_i}}}, \mathrm{e}^{-\beta_i(k)\frac{|z_i(x)-b_i|}{\sigma_{z_i}}} \right) \tag{11}$$

or

$$f_i(x) = \max\left(0, 1 - \beta_i(k)\frac{|z_i(x)-a_i|}{\sigma_{z_i}}, 1 - \beta_i(k)\frac{|z_i(x)-b_i|}{\sigma_{z_i}} \right), \tag{12}$$

where $\beta_i(k)$ is a positive control process chosen by the designer, and σ_{z_i} is the standard deviation of z_i. With the definitions in (11) and (12), it is clear that $0 < f_i(x) \le 1$ for all $x \in D$, and the maximum value happens at the input point(s) having the output measure equivalent to one of the threshold values (i.e., a_i or b_i). As the output measure $z_i(x)$ gets further away from the threshold values, $f_i(x)$ becomes exponentially or linearly smaller (i.e., away from 1). It should be noted that, the goal of placing the divisor σ_{z_i} is to make the weight function scale-free. In practice, σ_{z_i} can be estimated by all available data. Fig. 4 shows two examples of the weight function $f_i(x)$ defined in (11) and (12).

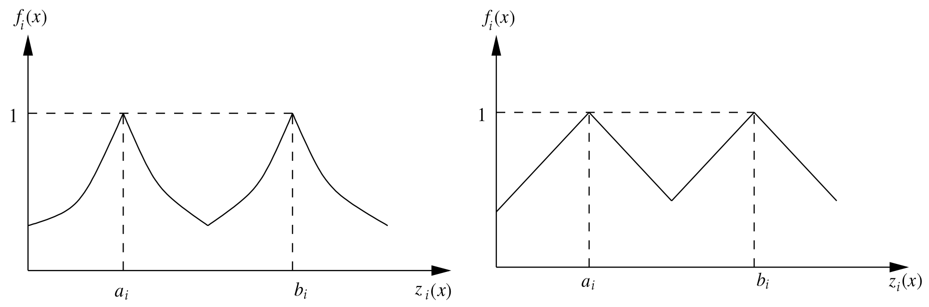

Fig. 4. (Left panel): The exponential weight function defined in (11). (Right panel): The linear weight function defined in (12). Note that $\beta_i(k)$ and σ_{z_i} are chosen to be one for both cases.

To account for all output measures of interest, we define another function

$$f_i'(x) = \begin{cases} 1 & \text{if } x \in T_i(x), \\ f_i(x) & \text{if } x \notin T_i(x). \end{cases} \tag{13}$$

The overall weight function is then defined as

$$f(x) = \begin{cases} \max\limits_{i=1,\ldots,I} f_i(x) & \text{if } x \in T(x), \\ \prod\limits_{i=1}^{I} f_i'(x) & \text{if } x \notin T(x). \end{cases} \tag{14}$$

From (11)–(14), it can be shown that $0 < f(x) \le 1$ for all $x \in D$. Further, $f(x)$ has the maximum value on the boundary of the target region $T(x)$ and becomes smaller as the input point gets further away from the boundary. To normalize the overall weight function $f(x)$, we simply define

$$f^*(x) = \frac{f(x)}{\int_{x\in D} f(x)\mathrm{d}x}, \quad x \in D. \tag{15}$$

It is noted that the value of $f^*(x)$ is determined by all the output measures $y_1(x), \ldots, y_m(x)$ (or $z_1(x), \ldots, z_I(x)$). In practice, each output measure $y_i(x)$ can be estimated by fitting an adequate model over the entire input region D.

3.2. Fitting response models

In this section, we introduce two techniques that are used to model the responses in computer experiments. The first technique is the Gaussian stochastic process (GASP) model, while the second one is support vector regression (SVR).

Gaussian Stochastic Process (GASP) Model. The GASP model was introduced by Sacks et al. (1989) and has been widely used for modelling the output from complex computer codes (Jones et al., 1998). Suppose we are given n observed data $\{(x_1, y_1), \ldots, (x_n, y_n)\}$, where the ith input is a K-dimensional vector $x_i = (x_{i1}, \ldots, x_{iK})$ and $y_i = y(x_i)$ is the corresponding output. Consider the model

$$y(x_i) = \mu + \epsilon(x_i), \quad i = 1, \ldots, n, \tag{16}$$

where μ is the overall mean, $\epsilon(x_i)$ is a spatial Gaussian process with $E[\epsilon(x_i)] = 0$, $\mathrm{Var}(\epsilon(x_i)) = \sigma_\epsilon^2$, $\mathrm{Cov}(\epsilon(x_i), \epsilon(x_j)) = \sigma_\epsilon^2 R_{ij}$, and

$$R_{ij} = \mathrm{Corr}(\epsilon(x_i), \epsilon(x_j)) = \prod_{k=1}^{K} \exp\{-\theta_k (x_{ik} - x_{jk})^{p_k}\}. \tag{17}$$

Thus, the output vector $y = (y_1, \ldots, y_n)'$ has a multivariate normal distribution $N_n(\mathbf{1}_n\mu, \Sigma)$, where $\Sigma = \sigma_\epsilon^2 R$, and $R = [R_{ij}]$.

An important feature of the GASP model is that, the correlation between the output measures is modelled as a stochastic process, while p_k corresponds to the "smoothness" of the response surface in the direction of the kth input factor. Since the likelihood for this model is straightforward, the likelihood (or profile likelihood) estimates of all parameters can be computed. To estimate the output at any untried input point x^*, one can utilize the best linear unbiased predictor (BLUP) for $y(x^*)$, of which the closed-form solution can be clearly presented (Henderson, 1975; Ranjan et al., 2008). Although the GASP model has many nice properties from a theoretical point of view, finding the likelihood estimates of all parameters becomes a hard task when the number of input factors becomes large (i.e., when K is large). We next introduce another modelling tool called support vector regression (SVR) that can easily handle the system with a large number of input factors.

Support Vector Regression (SVR). The SVR originates from the framework of statistical learning theory (Boser et al., 1992; Cortes and Vapnik, 1995; Guyon et al., 1993; Schölkopf, 1997; Vapnik, 1998). The ideas of SVR are summarized as follows. Suppose we are given n observed data $\{(x_1, y_1), \ldots, (x_n, y_n)\}$, where the ith input is a K-dimensional vector $x_i = (x_{i1}, \ldots, x_{iK})$ and $y_i = y(x_i)$ is the corresponding output. In ε-SVR, the goal is to find a function $g(x)$ that has at most ε deviation from the actually obtained outputs y_i for all the training data, and at the same time, is as flat as possible. For simple linear functions $g(x) = \langle \omega, x\rangle + b$, this corresponds to finding the solution of the following optimization problem:

$$\text{minimize } \frac{1}{2}\|\omega\|^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*)$$

$$\text{subject to } \begin{cases} y_i - \langle \omega, x_i\rangle - b \le \varepsilon + \xi_i \\ \langle \omega, x_i\rangle + b - y_i \le \varepsilon + \xi_i^* \\ \xi_i, \xi_i^* \ge 0. \end{cases} \tag{18}$$

Note that the l_2-norm $\|\omega\|^2$ takes into account the flatness of function $g(x)$, $\sum_{i=1}^{n}(\xi_i + \xi_i^*)$ is the amount up to which deviations lager than ε are tolerated, and $C > 0$ is the trade off between both (Vapnik, 1995). To achieve nonlinearity, the SVR algorithm finds the optimal solution of g in a high-dimensional feature space (or Hilbert space) $\mathcal{H}$ using a mapping $\Phi : D \to \mathcal{H}$. With this mapping, it is shown that the optimization solution depends on the data merely through inner products in $\mathcal{H}$, that is, on functions of the form $\langle \Phi(x_i), \Phi(x_j)\rangle$. Hence, a computationally cheaper way is to use a kernel function $k(x_i, x_j) = \langle \Phi(x_i), \Phi(x_j)\rangle$ instead of $\Phi(\cdot)$ explicitly.

A popular choice of the kernel function is the Gaussian kernel (or radial basis function), which has the form that

$$k(x_i, x_j) = \exp\left\{\frac{-\|x_i - x_j\|^2}{2\sigma^2}\right\}. \tag{19}$$

From the viewpoint of implementation, the Gaussian kernel has the following two advantages: (i) it can easily handle nonlinear models by mapping data into infinite-dimensional spaces; and (ii) it has relatively low complexity for model selection (since the model has only two unknown parameters C and σ^2). In practice, (C, σ^2) can be chosen by performing a grid search in R_+^2 and utilizing the idea of cross-validation so as to minimize the "mean square prediction error" (Hsu et al., 2003). For other approaches of choosing the best combination of (C, σ^2), the readers can refer to the work done by Keerthi and Lin (2003) and Huang et al. (2007).

3.3. Sequential weighted uniform design for target region estimation

We now propose a sequential approach for estimating the target region $T(x)$ of computer experiments, as described in the previous section. The idea is to first estimate the "importance" of each point x in the input domain D by fitting a response model for each output measure y_i based on an initial uniform design, and then sequentially update its "importance" by refitting each response model with one new trial added. When some stopping criterion has been achieved, the resulting estimation of $T(x)$ is then extracted. We summarize all the steps in the following.

Step 1: Superimpose N candidate grids in D and perform an initial uniform design of size n_0 based on (4). Perform experimental trials at the selected input grids and obtain the corresponding output measures.
Step 2: Fit a model $g_i(x)$ for each of the output measures y_i based on the existing data and obtain the estimated output measures $\hat{z}_1(x), \ldots, \hat{z}_I(x)$ for each grid point $x \in D$. Calculate the normalized weight function $f^*(x)$ using Eqs. (11)–(15).
Step 3: Augment the existing design points and select an additional one so that (6) is minimized (i.e. consider $f(x) = f^*(x)$ in (6)). Perform an experimental trial at the selected input point and obtain the corresponding output measures.
Step 4: Repeat Step 2–Step 3 until some prespecified criterion has been achieved.
Step 5: Obtain the estimated $\hat{z}_1(x), \ldots, \hat{z}_I(x)$ based on the resulting response models for each grid point $x \in D$ and extract the ones satisfying (8).

In the first step of this procedure, an initial allocation of design points is used. The purpose of this step is to have an initial set of trials uniformly scattered in the input region so that a response model can be fit to obtain a fairly good estimation for each output measure. Note that with a prespecified budget of available experimental trials, an initial allocation of about 25%–35% of the total budget is a suggested rule of thumb (Ranjan et al., 2008). However, here we suggest an initial allocation of about 25%–50% of the total experimental budget. The reason is that, due to the feature of the weight function defined in (11) and (12), the input points away from the "initially estimated boundary $\bar{T}(x)$" might only have a little chance to be selected in the following iterations. Therefore, increasing the size of initial design is sometimes necessary in order to reduce the overall predictive uncertainty.

In Step 2, a response model is fit for each output measure y_i based on the existing data and the estimated values $\hat{y}_i(x)$ for all $x \in D$ are obtained. All the functions $f_i(x)$ and $f_i'(x)$ are calculated using (11)–(13), wherein $\beta_i(k)$ is chosen to be a nondecreasing function of k (k corresponds to the stage of the sequential design). The goal of choosing $\beta_i(k)$ to be a nondecreasing function of k is that, as the number of allocated design points increases we gain more and more information about the output models $y_i(x)$, therefore, adding more design points near the boundary of the target region intuitively can improve the estimation of $\bar{T}(x)$. To obtain the normalized overall weight function $f^*(x)$ defined in (15), we first calculate the overall weight function $f(x)$ by (14) and then estimate $\int_{x \in D} f(x)\mathrm{d}x$ by its Riemann sums (using all candidate grids in D). As mentioned before, the resulting $f^*(x)$ represents the (estimated) "importance" of each input x in D for identifying the target region $T(x)$.

In Step 3, an additional input point is selected by the proposed uniform design based on the weight function $f^*(x)$. It is worth noting that by augmenting the existing data, this step allows for efficient use of available resources. In addition, the allocation of the new design point aims to (i) maximize the information for improving the model fitting near the boundary of $T(x)$, and (ii) reduce the overall predictive uncertainty.

Lastly, in Steps 4 and 5, a set of grids is extracted to be the final estimation of $T(x)$ when the prespecified criterion has been achieved. In this study, we consider two possible criteria: (i) the prediction accuracy of the response models; and (ii) the experimental budget (i.e., the number of experimental trials allowed). We now explain how to use criterion (i) as a stopping rule. Denote the estimated mean square prediction error of the output measure y_i after n iterations by $\widehat{\mathrm{MSE}}_i(n)$, where

$$\widehat{\mathrm{MSE}}_i(n) = \frac{1}{(n_0+n)} \sum_x [\hat{y}_i(x) - y_i(x)]^2, \quad i = 1, \ldots, m. \tag{20}$$

The proposed procedure stops after n iterations if

$$\max_{i=1,\ldots,m} \widehat{\mathrm{MSE}}_i(n) < \delta \tag{21}$$

or

$$\max_{i=1,\ldots,m} \frac{|\widehat{\mathrm{MSE}}_i(n) - \widehat{\mathrm{MSE}}_i(n-1)|}{\widehat{\mathrm{MSE}}_i(n-1)} < \delta' \tag{22}$$

for some positive δ and δ'. As can be seen from (21) and (22), both criteria provide a certain level of prediction accuracy for all output measures. However, it should be noted that the criterion defined in (21) is scale-dependent, while the criterion defined in (22) is scale-free. Criterion (ii) is straightforward, since trials cannot be undertaken when the experimental budget is out.

4. Performance assessment

To evaluate the performance of our proposed methodology, we consider the following two examples. The first example is the so-called Goldprice function that has been studied by Andre et al. (2000) and Ranjan et al. (2008). The second example comes from the queueing system introduced in Section 1, where the target region is comprised of multiple responses. To implement our proposed UD method, we choose $m = 2$, $a_1 = 0$, and $p = 2$ so that the computation of Eq. (4) is simplified (see Section 2.1 for details). Further, the exponential weight function defined in (11) is chosen to construct the weighted UD.

Example 4.1 (*The Goldprice Function*). The Goldprice function is given by

$$\begin{aligned} y(x_1, x_2) &= [1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)] \\ &\quad \times [30 + (2x_1 - 2x_2)^2(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)], \end{aligned}$$

where two input factors x_1 and x_2 are defined on the domain $D = (-2, 2) \times (-2, 2)$. For this example we consider the target region $T(x) = \{(x_1, x_2) \in D : y(x_1, x_2) \geq 1.5 \times 10^5\}$, of which the boundary $\bar{T}(x)$ corresponds to the contour line at $y(x_1, x_2) = 1.5 \times 10^5$. To implement our proposed methodology, we first superimpose $40^2 = 1600$ fine grids in the input domain D. The (estimated) true target region based on the simulation at all these 1600 grid points is shown in Fig. 5.

Suppose now the total number of simulation trials is limited to be $n = 40$, and we first consider an initial design of size $n_0 = 15$. The control process is chosen to be $\beta(k) = 4k/(n - n_0)$, where k represents the number of input points added after the initial design, $k = 1, \ldots, 25$. The result of our proposed method based on fitting SVR models is shown in Fig. 6.

Remark: Note that choosing $\beta(k) = 4k/(n - n_0)$ results in the fact that $0 < \beta(k) \leq 4$ for $k = 1, \ldots, n - n_0$, which is a rule of thumb we obtained from a large number of simulation trials.

The upper left panel of Fig. 6 shows the estimated boundary of the target region (or contour line) based on an initial uniform design of size $n_0 = 15$. As can be seen, the estimation totally misses the true target region in the bottom right of input domain D. The upper right panel of Fig. 6 shows the estimated target region after 8 additional design points are added. As can be seen, due to the feature of the proposed weighted uniform design, some of the newly added points are allocated near the estimated boundary, while others are allocated in the sparsely sampled regions. In particular, the true target region in the bottom right of input domain is detected when the 8th point (numbered 23) is added. The lower left panel of Fig. 6 shows that the true target region is very well estimated after 15 additional design points are added. Further, by sequentially increasing the weight of the control process $\beta(k)$, we see that more and more design points are allocated

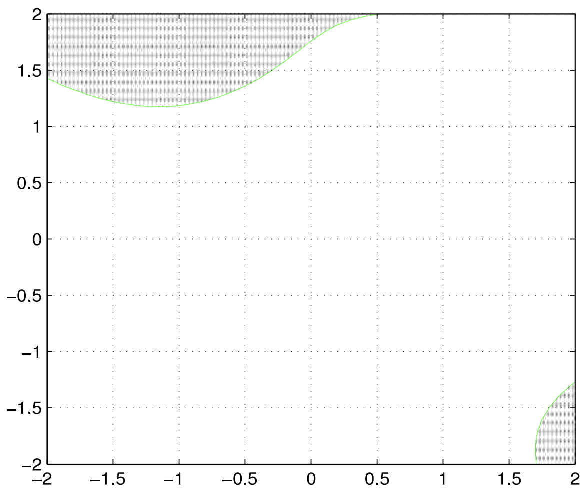

Fig. 5. The true target region $T(x) = \{(x_1, x_2) \in D : y(x_1, x_2) \geq 1.5 \times 10^5\}$ (the gray part) for the Goldprice function based on 1600 simulation trials.

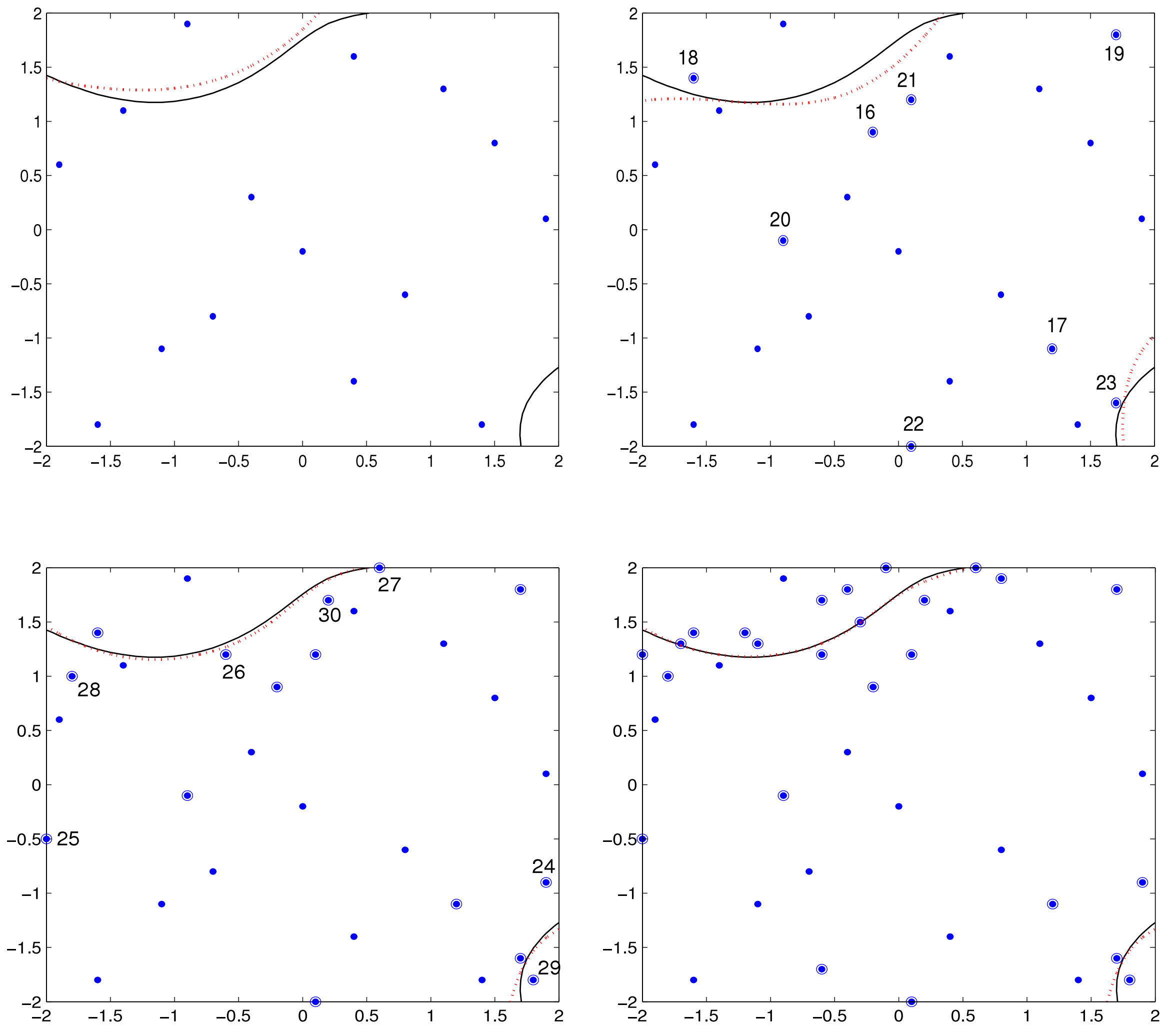

Fig. 6. The illustration of the proposed sequential UD and the estimated target region based on fitting SVR models. Note that the small dotted points refer to the initial design, the solid curve refers to the true boundary of the target region, while the dotted curve refers to the estimated boundary.

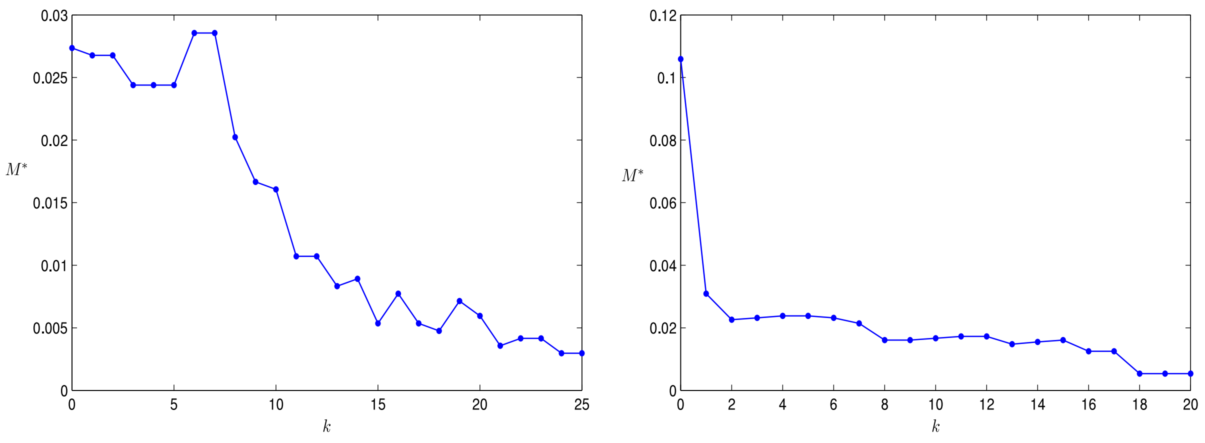

Fig. 7. (Left panel): The discrepancy measure (M^*) versus the number of additional points (k) added after the initial design of size $n_0 = 15$. (Right panel): The discrepancy measure (M^*) versus the number of additional points (k) added after the initial design of size $n_0 = 20$.

Table 2
The comparison of discrepancy measure M^* for five different methods. Note that for the methods based on sequential design, M^* are calculated with an initial design of size $n_0 = 10$, 15, and 20, while for the methods based on pure UD, M^* are calculated with all design points allocated at one time.

Methods	Discrepancy measure $M^*(n = 40)$		
	$n_0 = 10$	$n_0 = 15$	$n_0 = 20$
GASP + Sequential design	0.00654	0.00535	0.00773
SVR + Sequential UD	0.00535	0.00297	0.00535
GASP + Sequential UD	0.00000	0.00119	0.02261
GASP + Pure UD		0.00654	
SVR + Pure UD		0.00654	

near the boundary of the target region, thus improving the accuracy of estimation. The final estimated target region with 25 added design points is shown in the lower right panel of Fig. 6.

Ranjan et al. (2008) considered three discrepancy measures (lack in correlation, average L_2 distance, and maximum L_2 distance) to evaluate the closeness of the estimated contour to the true one. The shortcoming of these discrepancy measures is that, for small target regions that are not detected (such as the target region in the bottom right of Fig. 5), the false estimation will not be penalized (see Ranjan et al. (2008)). To overcome this problem, we propose another discrepancy measure based on the N superimposed candidate grids in D:

$$M^* = \left|\left(T(x) \setminus \tilde{T}(x)\right) \bigcup \left(\tilde{T}(x) \setminus T(x)\right)\right| / N \tag{23}$$

where $T(x) \subset D$ is the true target set and $\tilde{T}(x) \subset D$ is the resulting estimated target set. This new discrepancy measure accounts for the proportion of candidate grids located in the difference of two sets $T(x)$ and $\tilde{T}(x)$—the smaller the measure is, the closer the estimated target set is to the true one (thus having higher prediction accuracy).

The diagnostic plots of the discrepancy measure M^* for the initial design of size $n_0 = 15$ and $n_0 = 20$ are shown in Fig. 7. As can be seen from Fig. 7, the value of M^* has a decreasing trend (to zero) as the number of added points gets larger. Theoretically, our estimated target set will converge to the true target set as the number of design points becomes large.

The numerical results of M^* are shown in Table 2, wherein the following methods are compared: (i) the GASP model based on a sequential design proposed by Ranjan et al. (2008); (ii) the SVR model based on our proposed sequential UD; (iii) the GASP model based on our proposed sequential UD; (iv) the GASP model based on pure UD (i.e., with 40 design points allocated at one time); and (v) the SVR model based on pure UD.

As can be seen from Table 2, the proposed sequential UD outperforms other design methods for almost all given sizes of initial design. An exception is found in the case when $n_0 = 20$, where the GASP model has a relatively larger prediction error compared with other methods. This may be due to the fact that the GASP model is rather sensitive to the observed data, thus missing the target region in the bottom right. Therefore, reducing the size of initial design seems necessary when the GASP model is employed. On the other hand, the SVR model reveals to be rather stable in terms of prediction for the given sizes of initial design. It is also noted that both the GASP and SVR models based on pure UD perform "fairly well" in this example. The reason is that the input domain D is now a small two-dimensional rectangle, while relatively sufficient simulation budget (say $n = 40$) is provided.

Example 4.2 (*The Queueing Example with Multiple Responses*). Recall the queueing system introduced in Example 1 of Section 1, where we assume there are only two queues and jobs arrive in accordance with independent Poisson processes

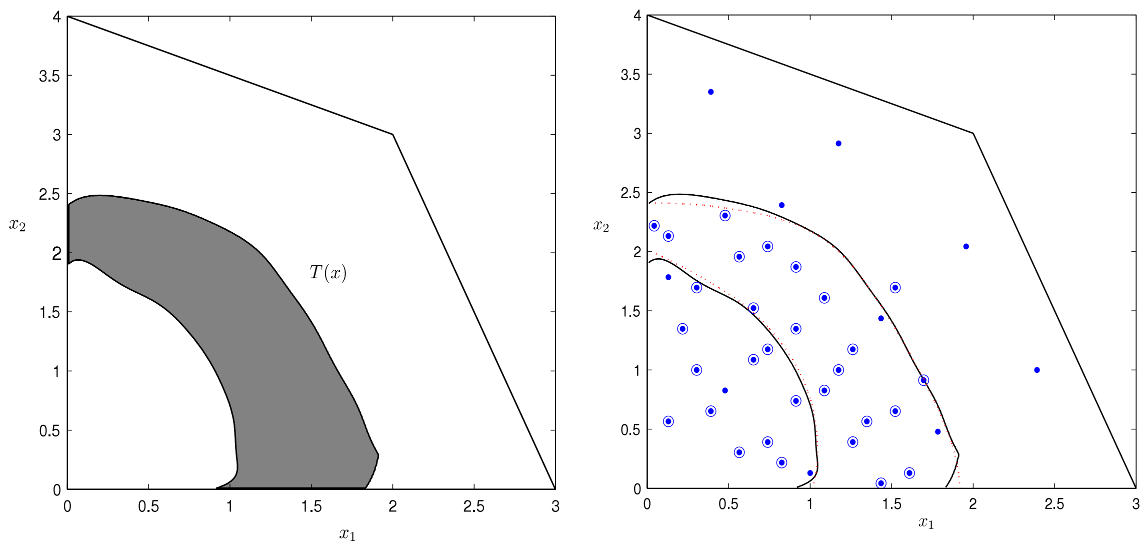

Fig. 8. (Left panel): The true target region $T(x) = \{x \in D : y_1(x) < 1,\ y_2(x) < 1, y_3(x) > 0.5\}$ (the gray part) based on 1124 simulation trials. (Right panel): The illustration of the proposed sequential UD ($n = 40$) with an initial design of size $n_0 = 10$ (small dotted points) and the estimated target region (dotted line) based on fitting SVR models.

Table 3
The comparison of discrepancy measure M^* for four different methods. Note that for the methods based on sequential UD, M^* are calculated with an initial design of size $n_0 = 10$, 15, and 20, while for the methods based on pure UD, M^* are calculated with all design points allocated at one time.

Methods	Discrepancy measure $M^*(n = 40)$		
	$n_0 = 10$	$n_0 = 15$	$n_0 = 20$
GASP + Sequential UD	0.03203	0.02313	0.03292
SVR + Sequential UD	0.00712	0.02580	0.03826
GASP + Pure UD		0.21174	
SVR + Pure UD		0.10587	

with rates x_1 and x_2, respectively. Suppose there are three service rate vectors $U_1 = (3, 0)$, $U_2 = (2, 3)$, and $U_3 = (0, 4)$. Thus, the input rate vector $x = (x_1, x_2)$ belongs to a convex polygon D, as shown in the left panel of Fig. 1. For this example, we are interested in how the input rate vector x affects the following three performance measures regarding "quality of service" under the so-called MaxProduct policy. The first performance measure $y_1(x)$ is the average delay (i.e., the average time waiting until service is first provided) of jobs in queue 1, the second performance measure $y_2(x)$ is the average delay of jobs in queue 2, while the third performance measure $y_3(x)$ is the average delay of jobs in both queues. Suppose now the desired target region is $T(x) = \{x \in D : y_1(x) < 1,\ y_2(x) < 1,\ y_3(x) > 0.5\}$ and 1124 grids are superimposed in the domain D. The estimated target region based on simulating the system at all candidate grid points is shown in the left panel of Fig. 8. Assume the total number of simulation trials is limited to be $n = 40$, the allocated design points and the estimated target region based on the proposed sequential UD (with initial design of size $n_0 = 10$) and fitting SVR models are shown in the right panel of Fig. 8.

Note that since the sequential design proposed by Ranjan et al. (2008) cannot handle the target region with multiple responses, it is excluded from comparison. The numerical results of the discrepancy measure M^* for other methods are shown in Table 3. As can be seen from Table 3, the proposed sequential UD significantly outperforms pure UD for fitting both the GASP and SVR models with various sizes of initial design (say $n_0 = 10$, 15, and 20).

5. Concluding remarks

This research is mainly divided into two parts. In the first part, we propose a new UD method that is suitable for any types of design area. The proposed UD method has an important feature that the optimal design is invariant under coordinate rotations and can be properly extended so that lower-dimensional uniformity is also considered. In order to reduce the computational cost of finding the optimal design, we propose an efficient algorithm to construct a so-called nearly uniform design (NUD). The numerical results also show that the constructed NUD approximates very well the optimal design for small sizes of experiment. In the second part, we develop an efficient methodology for estimating the target region of computer experiments. The methodology is sequential and comprised of two main components: (i) design; and (ii) fitting response models. For component (i), the proposed UD method with sequentially updated weight functions is utilized; while for component (ii), the GASP and SVR models with sequentially updated parameters are employed. It is noted that the proposed methodology can be viewed as an extension of the work done by Ranjan et al. (2008). However, it has a more general scope from the following viewpoints: (i) the employment of SVR models allows us to easily handle the experiments with a large number of input factors; (ii) the proposed methodology can handle the target region comprised of multiple output measures (remember the work done by Ranjan et al. (2008) can handle merely one output measure). The numerical results also show that our proposed methodology outperforms other approaches in estimating the target region of various computer experiments. We are currently investigating the computational issues that arise with high-dimensional input spaces and also how to best compare the performance of different approaches.

References

Andre, J., Siarry, P., Dognon, T., 2000. An improvement of the standard genetic algorithm fighting premature convergence. Advances in Engineering Software 32 (1), 49–60.

Armony, M., Bambos, N., 2003. Queueing dynamics and maximal throughput scheduling in switched processing systems. Queueing Systems: Theory and Applications 44, 209–252.

Boser, B.E., Guyon, I.M., Vapnik, V.N., 1992. A training algorithm for optimal margin classifiers. In: The 5th Annual ACM Workshop on COLT. pp. 144–152.

Box, G.E.P., Drapper, D.R., 1987. Empirical Model Building and Response Surfaces. John Wiley & Sons, New York.

Cheng, C.S., Li, K.C., 1995. A study of the method of principal Hessian direction for analysis of data from design experiments. Statistica Sinica 5, 617–639.

Cortes, C., Vapnik, V.N., 1995. Support vector networks. Machine Learning 20, 273–297.

Fang, K.T., 1980. The uniform design: Application of number-theoretic methods in experimental design. Acta Mathematical Application Sinica 3, 363–372.

Fang, K.T., Hickernell, F.J., 1995. The uniform design and its applications. Bulletin of Institute of International Statistics 333–349. 50th Session, Book 1.

Fang, K.T., Li, J.K., 1995. Some new results on uniform design. Chinese Science Bulletin 40, 68–72.

Fang, K.T., Lin, D.K.J., 2003. Uniform experimental designs and their applications in industry. In: Handbook of Statistics, vol. 22. pp. 131–170.

Fang, K.T., Lin, D.K.J., Winker, P., Zhang, Y., 2000. Unifrom design: Theory and applications. Technometrics 42, 237–248.

Fang, K.T., Ma, C.X., Winker, P., 2001. Centered L_2-discrepancy of random sampling and Latin hypercube design, and construction of uniform design. Mathematical Computation 71, 275–296.

Fang, K.T., Qin, H., 2003. A note on construction of nearly uniform designs with large number of runs. Statistics & Probability Letters 61, 215–224.

Fang, K.T., Shiu, W.C., Pan, J.X., 1999a. Uniform designs based on Latin squares. Statistica Sinica 9, 905–912.

Fang, K.T., Tian, G.L., Xie, M.Y., 1999b. Uniform design over a convex polyhedron. Chinese Science Bulletin 44, 112–114.

Fang, K.T., Wang, Y., 1994. Number-theoretic Methods in Statistics. Chapman and Hall, London.

Fang, Y., 1995. Relationships between uniform design and orthogonal design. In: The 3rd International Chinese Statistical Association Conference, Beijing.

Guyon, I.M., Boser, B.E., Vapnik, V.N., 1993. Automatic capacity tuning of very large VC-dimension classifiers. Advances in Neural Information Processing Systems 5, 147–155.

Henderson, C.R., 1975. Best linear unbiased estimation and prediction under a selection model. Biometrics 31, 423–447.

Hickernell, F.J., 1998. A generalized discrepancy and quadrature error bound. Math. Comput. 67, 299–322.

Hickernell, F.J., 1999. Goodness-of-fit statistics, discrepancies and robust dessigns. Statistics & Probability Letters 44, 73–78.

Hsu, C.W., Chang, C.C., Lin, C.J., 2003. A practical guide to support vector classification. Technical Report CWH03a, Department of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan.

Huang, C.M., Lee, Y.J., Lin, D.K.J., Huang, S.Y., 2007. Model selection for support vector machines via uniform design. Computational Statistics & Data Analysis 52, 335–346.

Hung, Y.C., Chang, C.C., 2008. Dynamic scheduling for switched processing systems with substantial service-mode switching times. Queueing Systems: Theory and Applications 60, 87–109.

Hung, Y.C., Michailidis, G., 2008. Modeling, scheduling, and simulation of switched processing systems. ACM Transactions on Modeling and Computer Simulation 18, Article 12.

Hung, Y.C., Michailidis, G., Bingham, D.R., 2003. Developing efficient simulation methodology for complex queueing networks. In: Proceedings of the Winter Simulation Conference, New Orleans. pp. 152–159.

Jones, D., Schonlau, M., Welch, W., 1998. Efficient global optimization of expensive Black–Box functions. Journal of Global Optimization 13, 455–492.

Keerthi, S.S., Lin, C.-J., 2003. Asymptotic behaviors of support vector machines with Gaussian kernel. Neural Computation 15, 1667–1689.

Ma, C.X., 1997a. A new criterion of uniformity — Symmetrical discrepancy. Journal of Nankai University 30, 30–37.

Ma, C.X., 1997b. Construction of uniform designs using symmetrical discrepancy. Application of Statistics and Manegement 166–169.

Ma, C.X., Fang, K.T., 2004. A new approach to construction of nearly uniform designs. International Journal of Materials and Product Technology 20, 115–126.

Ranjan, R., Bingham, D., Michailidis, G., 2008. Sequential experiment design for contour estimation from complex computer codes. Technometrics 50, 527–541.

Sacks, J., Welch, W.J., Mitchell, T.J., Wynn, H.P., 1989. Design and analysis of computer experiments. Statistical Science 4, 409–423.

Schölkopf, B., 1997. Support Vector Learning. R. Oldenbourg Verlag, Munich.

Sharma, S., 1996. Applied Multivariate Techniques. Wiley.

Vapnik, V.N., 1998. Statistical Learning Theory. Wiley, New York.

Vapnik, V.N., 1995. The Nature of Statistical Learning Theory. Springer, New York.

Wang, Y., Fang, K.T., 1981. A note on uniform distribution and experimental design. KeXue TongBao 26, 485–489.

Winker, P., Fang, K.T., 1998. In: Niederreiter, H., Zinterhof, P., Hellekalek, P. (Eds.), Optimal U-type Design. Monte Carlo and Quasi-Monte Carlo Methods 1996. Springer, 436–448.

Wu, C.F.J., Hamada, M., 2000. Experiments: Planning, Analysis, and Parameter Design. Wiley, New York.