Computational Statistics and Data Analysis 54 (2010) 1505–1515

Contents lists available at ScienceDirect

Computational Statistics and Data Analysis

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

Optimized U-type designs on flexible regions

D.K.J. Lin[a], C. Sharpe[b], P. Winker[b,*]

[a] *Pennsylvania State University, USA*

[b] *University of Giessen, Germany*

ARTICLE INFO

Article history:
Received 20 August 2009
Received in revised form 27 January 2010
Accepted 27 January 2010
Available online 2 February 2010

Keywords:
Central composite discrepancy
Experimental design
Flexible regions
Threshold Accepting
U-type design

ABSTRACT

The concept of a flexible region describes an infinite variety of symmetrical shapes to enclose a particular region of interest within a space. In experimental design, the properties of a function on the region of interest are analyzed based on a set of design points. The choice of design points can be made based on some discrepancy criterion. The generation of design points on a flexible region is investigated. A recently proposed discrepancy measure, the central composite discrepancy, is used for this purpose. The optimization heuristic Threshold Accepting is applied to generate low-discrepancy U-type designs. The proposed algorithm is capable of constructing optimal U-type designs under various flexible experimental regions in two or more dimensions. The illustrative results for the two-dimensional case indicate that by using an optimization heuristic in combination with an appropriate discrepancy measure produces high-quality experimental designs on flexible regions.

1. Introduction

The development of a flexible region approach by Draper and Guttman (1986) built on previous research in experimental design. The central issue is to understand the effect of selecting points in the experimental domain (input space) for response surface analysis, where the functional relationship between input and output is unknown. It is assumed that a sub-space of the input space can be localized such that inputs from this sub-space produce an output that meets some set objective. This sub-space selected for experimental examination is called the region of interest and is represented by a set of design points.

In standard settings, simple shapes such as spherical, cuboidal, and triangular ones are preferred for the region of interest (Draper and Lawrence, 1965, 1966). The cuboidal shape, typically a multi-dimensional unit cube, has received a great deal of attention, and many approaches to construct good experimental designs have been proposed. A survey is provided by Fang et al. (2006), and a recent application to spherical regions of interest is in Huang et al. (2010). However, for many applications, a standard shape might not provide a good approximation of the region of interest. The target is to generate low-discrepancy designs for a given region of interest, but it is difficult to derive appropriate designs from simple transformations. Designs constructed for differently shaped regions of interest are viable solutions to overcome this obstacle.

The flexible region approach as proposed by Draper and Guttman (1986) is a combination of the first two standard shapes (spherical and cuboidal). By adjusting a single parameter, it is possible to obtain an infinite variety of intermediate symmetrical shapes. This approach is particularly useful because it offers the possibility for intuitive linguistic mapping to shape types. When applying a flexible region to a real problem, Draper and Guttman (1986) point out that often there is some a priori knowledge as to the structure of the region of interest, which is first expressed in terms of natural language.

* Corresponding address: Department of Economics, University of Giessen, Licher Street 64, 35394 Giessen, Germany. Tel.: +49 641 99 22 640; fax: +49 641 99 22 649.

E-mail address: Peter.Winker@wirtschaft.uni-giessen.de (P. Winker).

0167-9473/$ – see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.csda.2010.01.032

The specification is then mapped onto a specific value of the parameter determining the shape of the flexible region. This process offers some intriguing possibilities to identify a tool to systematically generate suitable flexible regions.

We consider U-type designs as proposed Fang (1980) and Wang and Fang (1981). In a U-type design, all design points are located on a fine, uniform grid over the search space. No specific assumptions are made about the functional relationship between the input factors and output. Thus, the aim is to find a U-type design that covers the input space as uniformly as possible (e.g., by minimizing some appropriate measure of discrepancy). The resulting optimized U-type designs gather the maximum amount of information for a given number of design points without assuming any functional for relating the inputs and outputs.

There are several common measures for uniformity, and at first glance the centered L_2-discrepancy proposed by Hickernell (1996) would appear to be the most reasonable choice given that a flexible region is a symmetrical shape with its origin at the center of the region of interest. However, as with other discrepancy measures, the discrepancy value is defined for a hypercube and cannot be directly adapted for other shapes. The central composite discrepancy (CCD) suggested by Chuang and Hung (2010) is not fixed at a constant point from which to calculate the discrepancy value, which removes the constraint on the shape type and hence makes it possible to apply it also for a flexible region. At present, there is no closed-form expression for the CCD. Therefore, the calculation of the CCD for a given design is computationally intensive, which limits the dimension and size of designs to be considered using this approach.

The process of generating low-discrepancy designs on a flexible region is framed as an optimization problem, with the CCD as the objective function to be minimized. Threshold Accepting (TA) is a heuristic technique used to tackle this problem. It has been applied successfully to many experimental design problems (see, e.g., Fang et al. (2000), Winker (2001, Ch. 11), and Fang et al. (2005)). For some problem instances, lower bounds are derived that could be reached by the TA implementation (Fang et al., 2003).

We describe how to produce designs of low discrepancy for a variety of flexible region shapes and design configurations. The results show that the TA implementation is able to construct U-type designs with low values of the CCD for different regions of interest.

The remainder of this paper is structured as follows: In Section 2 we discuss Draper and Guttman (1986) formulation of a flexible region; in Section 3 we cover U-type design and the CCD as measure of uniformity; in Section 4 we state the TA algorithm in the context of design optimization; in Section 5 we describe the experiment setup and results. In Section 6, finally, we summarize our findings and consider the next steps for research.

2. Flexible region formulation

For any positive value of m, Draper and Guttman (1986) define a flexible region R in dimension s by the following constraint on the potential design points $x = (x_1, \dots, x_s)$:

$$|x_1|^m + |x_2|^m + \cdots + |x_s|^m \leq 1. \tag{1}$$

This definition is for the hypercube $[-1, 1]^s$, while in recent work on experimental design the hypercube $C^s = [0, 1)^s$ is considered. To account for this change, the original definition is modified as follows: For a given shape parameter $m > 0$, the flexible region R_m is defined by

$$R_m = \{x \in [0, 1]^s | (|2(x_1 - 0.5)|^m + \cdots + |2(x_s - 0.5)|^m) \leq 1\}. \tag{2}$$

In both the original and the modified definition, the parameter m determines the shape of the region. Fig. 1 shows some examples for different values of m in the two-dimensional case. For this figure – as in the following optimization approach – only design points located on a grid over the unit cube are considered. Specifically, the grid consists of q^2 points, where $q = 49$ for the current application.

As can be seen, when $m \to \infty$ the entire input space is the region of interest, and when $m = 0.3$ only the very center and the extreme points extending to one edge cover the region of interest. It might be observed that as m gets smaller the flexible region transformation is a deformation and rotation of three types of symmetric shapes, which are, in their most basic form, a square, circle, and astroid.

With regard to the goal of a uniform covering of the input space by only a few design points, Fig. 1 shows that using good designs for hypercubes ($m \to \infty$) is not a feasible approach for smaller values of m. In particular, a simple projection of a good design on C^s to R_m will contain only a few points for small values of m. These points might not be scattered uniformly on R_m. Constructing more refined mappings and keeping the number of design points fixed while minimizing the discrepancy on R_m appears impossible. Therefore, in the following we consider the explicit construction of low-discrepancy designs on flexible regions based on a measure of discrepancy appropriate for each region.

3. *U*-type design and central composite discrepancy

We define a U-type design $\mathcal{P}$ as a set of n points, $\mathcal{P} = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$, sampled from the s-dimensional unit cube C^s on a grid with q^s points. This set of points can also be described by a $n \times s$ design matrix U, where each row corresponds to one run and each column to one factor. Thereby, the factors can take on $l = 1, \dots, q$ different levels. The correspondence

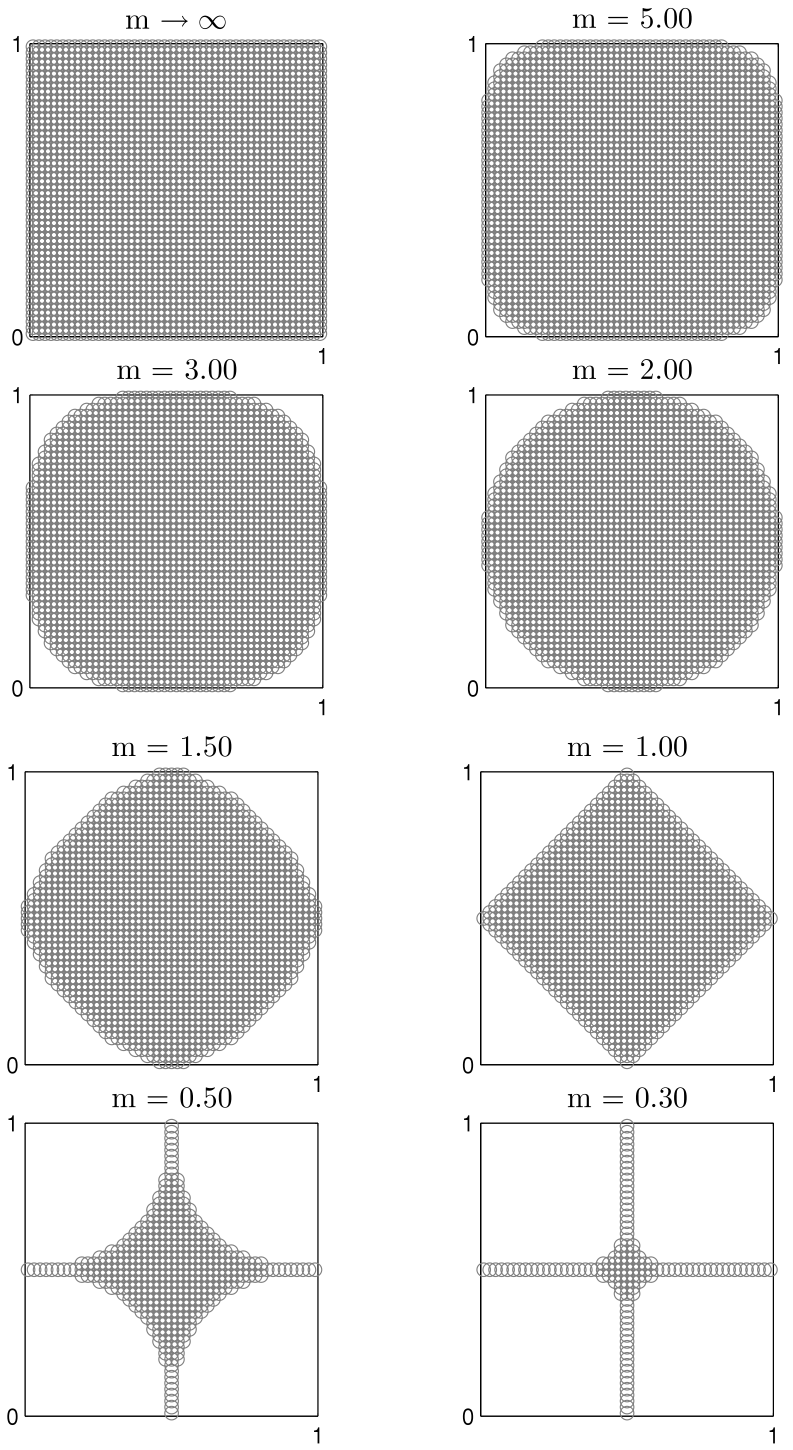

Fig. 1. Examples of flexible regions.

between the set of design points and the set of design matrixes $U(n, q^s)$ is given by the transformation $\frac{2l-1}{2q}$, $l = 1, \ldots, q$. As a result, the number of possible grid points grows at rate q^s. Therefore, each design will only consist of n grid points that typically is much smaller than q^s.

Having to represent the input space by a subset of grid points means that these points have to be selected in an optimal way. To this end, first one has to define properties of 'good' point sets, which is done using measures of design uniformity (Fang et al., 2006, Ch. 3). The lower the design discrepancy value, the more uniform the design. Several discrepancy formulations have been proposed for the case of unit hypercubes (Hickernell, 1996). However, these measures are not appropriate for a flexible region when m is small. In fact, low-discrepancy designs based on these standard measures put almost all design points on the boundary of the flexible region.

The central composite discrepancy (CCD), as proposed by Chuang and Hung (2010), measures uniformity only with regard to the flexible region, i.e., considering subsets falling within it. While the concept is similar to the centered L_2-discrepancy according to Hickernell (1996), it provides a useful discrepancy measure that is suitable for differently shaped regions of interest. The CCD for a set of points $\mathcal{P}$ in a region of interest R is defined by

$$\mathrm{CCD}_p(\mathcal{P}) = \left\{ \frac{1}{v(R)} \int_R \frac{1}{2^s} \sum_{k=1}^{2^s} \left| \frac{N(R_k(x), \mathcal{P})}{n} - \frac{v(R_k(x))}{v(R)} \right|^p \mathrm{d}x \right\}^{1/p}, \tag{3}$$

where $p > 0$ is a parameter defining the underlying norm (for our application, $p = 2$ is used, corresponding to the L_2-norm); $v(R)$ is the volume of the region R and $v(R_k(x))$ the volume of $R_k(x)$, a subregion of R defined below; n is the number of all design points, and $N(R_k(x), \mathcal{P})$ the number of design points in subregion $R_k(x)$. In contrast to the centered L_2-discrepancy, no simple analytical expression is available for CCD_p on general flexible regions. Therefore, we resort to an approximation based on the q^s grid on $[0, 1]^s$ introduced above. The volume of R and $R_k(x)$ is approximated by the number of these grid points falling into these regions.

The principal idea of the CCD, and the reason why it can be applied to flexible regions, is that measurement is not taken from one fixed point: Every point in R is considered a center point. At a given grid point, we divide R into 2^s subregions — the approach can be generalized to m^s subregions for $m \leq 2$ (Chuang and Hung, 2010, p. 221). Dividing hyperplanes pass through the grid point and are parallel to one axis. For example, in the two-dimensional case, R is cut into four subregions by a horizontal and vertical line crossing at the grid point. For each resulting subregion R_k, the share of design points in that subregion from all design points $(N(R_k(x), \mathcal{P})/n)$ is compared with the relative volume of the sub-space, which is approximated by the fraction of grid points falling into $R_k(x)$ versus R. The absolute values of these differences are taken to the pth power before adding them up over all 2^s subregions. Multiplication by $\frac{1}{2^s}$ provides the average over the subregions. This calculation is repeated for all grid points in R. The resulting integral (which is again approximated by a finite sum as only points on the q^s grid are considered) provides the discrepancy measurement after normalization with $\frac{1}{v(R)}$.

An optimal U-type design $\mathcal{P}^*$ on R is obtained if the n design points from $\mathcal{P}$ are distributed in R to minimize $\mathrm{CCD}_p(n, \mathcal{P})$, or more formally,

$$\mathcal{P}^* = \arg\min_{P} \mathrm{CCD}_p(\mathcal{P}). \tag{4}$$

4. Threshold Accepting

The local search algorithm TA, first devised by Dueck and Scheuer (1990), is an optimization technique that has been applied in low-discrepancy designs on unit cubes for different measures of discrepancy (Fang et al., 2000, 2002). Based on theoretically lower bounds for the discrepancy, Fang et al. (2003, 2005) obtain uniform designs (U-type designs with the lowest possible discrepancy) for several problem instances using this methodology. It is highly flexible (non-problem specific) and simple to implement and adapt. The new algorithm for the construction of low-discrepancy designs on flexible regions is given by Algorithm 1. It is adapted from the algorithm suggested by Winker (2001, Ch. 11) for the uniform design problem on the unit cube. The fact that no closed-form analytical expression for CCD_p is available requires additional attention. Some of these issues will be addressed after discussing the general outline of the algorithm.

Algorithm 1 TA Algorithm for Design Optimization.

```
1:  Initialize design $U^c \in \mathcal{U}$
2:  Initialize $n_R$, $n_S$, and threshold sequence $\tau_r$, $r = 1, \ldots, n_R$
3:  for $r = 1$ to $n_R$ do
4:    for $i = 1$ to $n_S$ do
5:      Generate at random a solution $U^n \in \mathcal{N}(U^c)$
6:      if $\mathrm{CCD}(U^n) - \mathrm{CCD}(U^c) < \tau_r$ then
7:        $U^c = U^n$
8:      end if
9:    end for
10: end for
```

First (1:), we initialize the algorithm with a U-type design U^c on R_m whose points are chosen from the q^s grid points falling into R_m. To this end, each column of U^c is filled with n uniform random numbers drawn from $1, \ldots, q$. Obviously, the resulting design points x (rows of U^c after transformation to $[0, 1)$) may not fall in the flexible region R_m as desired. Therefore, the initial design matrix U^c is iterated through row by row. The corresponding design-point x is in the region of interest R_m if $\sum_{j=1}^{s} |x_j|^m \leq 1$. Any point that is found to lie outside the flexible region is moved until it is inside. Once all points are inside the region of interest, the CCD value is calculated for this design.

The second step is to generate the threshold sequence, a critical component of the algorithm and one that controls the criteria for accepting or rejecting the design changes at each iteration i. A data-driven approach as outlined by Fang and Winker (1997) is implemented to perform this task. It takes the initial design and selects a design-point element – a row and column index – at random. A level is drawn at random from $\{1, \ldots, q\}$ to replace the previous point element. The new point is tested as before to check whether it falls into R_m. If it is rejected, then the level selection process is repeated until the point is in the region of interest. The objective function value is calculated for this modified design and subtracted from the original objective function value; the absolute value of the difference is stored in a vector of length n_R. This process is repeated n_R times and the resulting vector is sorted in descending order providing the threshold sequence.

In the main body of the algorithm, from line 3:, we begin optimization of the initial design matrix. At each iteration we select a new design matrix U^n in a neighborhood of the current one ($\mathcal{N}(U^c)$) at random. The neighborhood mimics the idea of a natural ordering (Winker, 2001, Ch. 11) where a neighborhood is defined by a generalization of the Hamming distance. This states that the new design U^n might differ from U^c by entries in $k \leq s$ columns. We set $k = s$, and select each column at random (we might have several changes in one column). A row from that column is selected at random and the point element is replaced. The new point element can take any level as long as the point is inside the flexible region; if it is outside, we repeat the replacement procedure in exactly the same way as done in the threshold sequence generation until a feasible neighbor is found.

After the changes are applied, the objective function $\text{CCD}(U^n)$ is calculated for the new design U^n. Finally (6:), the new objective function value is subtracted from the objective function of the previous design matrix and the result compared against the current threshold value τ_r. The new design is deterministically adopted if this result is lower than the current threshold value.

At each step r the next value in the threshold sequence is taken, which has the effect of tightening by some degree the condition on accepting a design that is worse than the previous design. Thereby, it is expected that the final design after the n_R times n_S iterations of the algorithm should have a small CCD value, and hence design points are distributed uniformly over the flexible region. Although it is not made explicit in the algorithm, the best design discovered throughout the search process is always retained in memory. Whether we are able to find an optimal design will be discussed in the results section.

The algorithm for calculating the CCD along with a simple working example are presented in Appendix A. This is achieved by exploiting the symmetry of the flexible region in concert with the manipulation of simple data structures in the two-dimensional case. Its real worth is in the potential for extension into higher dimensions and also for the calculation of all symmetrical shape types. Useful as any development in this direction may be, research effort ought to be concentrated on finding a general closed-form solution to calculate the CCD in a flexible region, which may even allow to derive lower bounds.

5. Experiment description and results

5.1. Experimental setup

In order to analyze the performance of the TA implementation for generating low-discrepancy U-type designs on flexible regions, a set of experiments is run. In particular, different shapes (corresponding to different values of the parameter m) as well as different numbers of design points n are considered. Although the proposed algorithm is capable of constructing optimized designs in higher dimensions, we use $d = 2$ for illustration. The TA algorithm is run with different values for the total number of iterations, i.e., $n_R \times n_S$. For each problem instance, the best design obtained during the run of the TA algorithm as well as the corresponding CCD value is recorded.

All experiments are implemented using Matlab 2008a, with the objective function code written using Matlab's interface to a C compiler. Using compiled code for calculating the objective function reduces execution time by approximately one third. The algorithm is executed on an Intel 2 Core Quad, clocking at 2.83 GHz, with 8 GB of RAM, and with Windows XP Professional x64 Edition, 2003 operating system. For the most intensive experiment, execution time was approximately 26 h for 20 design configurations, with 10 separate runs for each configuration to take into account the stochastic component of TA. For each run a design was optimized over $n_R \times n_S = 10^6$ iterations. All codes and any other experiment details are available upon request.

For all designs $U(n, q^s)$ in the experiment, we set the dimension $s = 2$. For the unit cube ($m \to \infty$), the number of levels is set to $q = 31$, which corresponds to $q^2 = 961$ grid points. For smaller values of m, the number of levels is increased in order to keep the number of grid points in the region of interest almost constant. A significantly smaller number of levels would make the optimization process an almost trivial task. The grid needs to be of fine enough resolution to allow the optimization process to be effective in lowering the discrepancy once points are moved into the flexible region. Basically, for higher values of q, there are more points to choose from. Having an odd number of levels increases the space available for regions of interest for values $m < 1$.

The number of runs, n, is one of the two varying factors in the experiment. We select four values for n, {5, 7, 9, 11}. The parameter characterizing the shape of the flexible region, m, is the other varying factor. We select five values for m, {9999, 2, 1, 0.5, 0.3}, which represent a general sample of flexible region shape types. Note that due to the constraint on U-type designs on the q^s grid, the first value results in a covering of all grid points for the given design parameters. The number of levels is chosen as a function of m. For $m = 9999$ ($m \to \infty$), $q = 31$; for $m = 2,\ q = 35$; for $m = 1,\ q = 43$; for $m = 0.5,\ q = 75$; and for $m = 0.3,\ q = 159$ is used.

In addition to discussing the stochastic properties of TA applied to design problems, Winker (2006) suggests ways to best concentrate computational resources to speed up the rate of convergence towards a global optimum. The algorithm is run with a total of I iterations ($I = n_r \times n_S$, threshold reductions $\times$ design change iterations) and $\mathcal{R}$ replications (the number of times we restart an individual experiment with different random starting design and different random numbers for the

Table 1
CCD values[a] for optimized designs $U(n, 31^2)$ with $I = 10^5$ iterations, different values of n and m.

n	Best	Worst	Mean	Std. Dev.	Freq. of best
$m \to \infty,\ q = 31$					
5	5.865	6.318	6.028	0.132	1/10
7	3.303	3.580	3.464	0.100	1/10
9	2.278	2.657	2.465	0.118	1/10
11	1.766	1.953	1.865	0.070	1/10
$m = 2,\ q = 35$					
5	7.840	8.397	8.048	0.168	1/10
7	4.585	4.980	4.715	0.127	1/10
9	2.836	3.592	3.194	0.243	1/10
11	2.113	2.631	2.347	0.166	1/10
$m = 1,\ q = 43$					
5	10.457	10.853	10.671	0.145	1/10
7	5.738	6.219	5.973	0.168	1/10
9	3.774	4.297	4.028	0.192	1/10
11	2.557	3.172	2.841	0.202	1/10
$m = 0.5,\ q = 75$					
5	9.552	9.974	9.639	0.144	1/10
7	5.020	5.461	5.152	0.120	1/10
9	3.199	3.556	3.280	0.121	1/10
11	2.203	2.573	2.373	0.120	1/10
$m = 0.3,\ q = 159$					
5	7.772	7.905	7.794	4.730	6/10
7	4.233	4.298	4.265	2.630	2/10
9	2.728	2.840	2.765	4.355	1/10
11	2.162	2.402	2.220	7.914	1/10

[a] All entries in columns 2–5 are scaled by a factor of 10^3.

selection of neighbors). Thus, total computational resources used for one problem instance are approximately proportional to $C = I \times \mathcal{R}$. If the design matrix is small and the number of levels low, then it is possible to find an optimum design for $\mathcal{R} = \frac{C}{I}$, where $I \geq 5000$ might be sufficient, although most of the instances considered here might be classified as medium-sized problems owing to the number of levels. Of course, the specific shape of the flexible region might alter the problem type classification, e.g., for small m, the actual number of feasible grid points becomes rather small. These subtleties are not explored here. In general, it is not clear how to select I and $\mathcal{R}$ for a given C in order to find good or even optimal designs. The suggestion, in light of the resources at our disposal, is to select a large I and $10 \leq \mathcal{R} \leq 20$, as suggested by Winker (2001, p. 129ff). We select $\mathcal{R} = 10$. The computational load for a single evaluation of the objective function CCD directly influences the upper limit of I. We select $I \in \{100\,000,\ 1\,000\,000\}$, so we are able to see if the empirical results conform to convergence theory; i.e., whether the quality of the results improves with increasing I.

5.2. Results

The results in Tables 1 and 2 are for the two values of I. Given that we run 10 replications for each problem instance, the results obtained can be interpreted as an empirical distribution. To provide relevant distributional information along the guidelines suggested by Gilli and Winker (2007), the best and worse objective function values obtained by optimization are reported, along with the empirical mean $\hat{\mu}$ and standard deviation $\hat{\sigma}$, and the frequency at which the best objective function was obtained.

To corroborate the quality of these designs, Fig. 2 shows the distribution of points in the flexible region for the designs corresponding to the best discrepancy values reported in Table 2.

It can be seen that the points are evenly distributed over the design space, exploiting the space available as best possible, most noticeably when the flexible region is contracted. Therefore, it appears that using the CCD as a measure of uniformity is suitable for design optimization on flexible regions.

When running the algorithm repeatedly, it is observed that design patterns are generated that are simple rotations of one another and that exhibit the same discrepancy. This is an expected outcome since the CCD meets a key requirement for discrepancy measures, namely that it is invariant under rotation of the coordinates. However, we might also find designs with the same discrepancy that cannot be obtained from one another by a simple transformation, e.g., rotation. For such problem instances, considering low-dimensional projections as suggested by Chuang and Hung (2010, p. 224) might result in unique solutions though at much higher computational cost.

Table 2
CCD values[a] for optimized designs $U(n, 31^2)$ with $I = 10^6$ iterations, different values of n and m.

n	Best	Worst	Mean	Std. Dev.	Freq. of best
$m \to \infty,\ q = 31$					
5	5.643	5.867	5.741	0.070	1/10
7	3.197	3.428	3.296	0.075	1/10
9	2.052	2.220	2.128	0.049	1/10
11	1.470	1.679	1.596	0.061	1/10
$m = 2,\ q = 35$					
5	7.840	7.995	7.892	0.061	1/10
7	4.256	4.619	4.478	0.110	1/10
9	2.653	3.237	2.848	0.192	1/10
11	1.945	2.212	2.080	0.079	1/10
$m = 1,\ q = 43$					
5	10.457	10.605	10.472	0.047	9/10
7	5.629	5.880	5.742	0.076	1/10
9	3.503	3.992	3.706	0.143	1/10
11	2.439	3.017	2.687	0.195	1/10
$m = 0.5,\ q = 75$					
5	9.552	9.622	9.561	0.022	7/10
7	5.009	5.126	5.066	0.045	2/10
9	3.144	3.294	3.181	0.048	1/10
11	2.187	2.385	2.265	0.056	1/10
$m = 0.3,\ q = 159$					
5	7.772	7.780	7.773	0.002	3/10
7	4.233	4.270	4.246	0.014	2/10
9	2.728	2.789	2.739	0.023	5/10
11	2.162	2.208	2.177	0.019	1/10

[a] All entries in columns 2–5 are scaled by a factor of 10^3.

The statistical distribution of the discrepancy for optimized designs, as discussed by Winker (2006), is assumed to be left truncated. Furthermore, as I increases, $\hat{\mu}$ should converge to this lower truncation point, while $\hat{\sigma^2}$ decreases to zero. The ten-fold increase in I in Table 2, as predicted, produces lower-discrepancy values for $m > 0.5$, where it is less likely to consistently find the global minimum. The mean, $\hat{\mu}$, moves closer to the lowest value found, and the variance $\hat{\sigma^2}$ is reduced. In the case $m = 1$, $U(5, 43^2)$, the best value is found in 9 out of 10 runs, which suggests – but does not prove – that this design reaches the global minimum. Indeed, as $m < 1$, the frequency of finding the best value tends to increase. For larger flexible regions, i.e. $m > 1$, we cannot state how close our CCD values are to the global minimum since no lower bounds are known.

5.3. *Evaluation of results*

So far the results presented could demonstrate that the TA implementation is able to produce designs of low discrepancy on flexible regions. However, given that no lower bounds are available for the CCD, it cannot be proven that these designs are actually optimum. Therefore, we provide some additional evidence for the quality of our results by comparing them with randomly generated U-type designs on the flexible region and with U-type designs generated with the Switching Algorithm suggested by Chuang and Hung (2010).

Note that considering the projections of uniform designs for the unit cube on the flexible region would not be a reasonable benchmark as the actual number of points falling into the flexible region is hard to control for. Thus, we construct $I = 10^6$ random designs on the flexible region. The empirical distribution of the CCD values for these randomly generated designs, in particular the 1% quantile, provides a benchmark for evaluating the optimized designs. The Switching Algorithm proposed by Chuang and Hung (2010) also exhibits a stochastic component by its choice of the starting design. Therefore, we allowed for a number of restarts of the Switching Algorithm so that the total number of function evaluations is the same as for the Threshold Accepting algorithm ($I = 10^6$). Some statistics of the empirical distributions of the CCD values obtained by the two approaches are presented in Table B.3 in Appendix B.

For all problem instances considered, the mean and the variance of the designs obtained by the Threshold Accepting algorithm are substantially smaller than those obtained by the Switching Algorithm. In addition, TA-based designs exhibit a smaller CCD value than the 1% quantile of randomly generated U-type designs on the flexible region. Thus, we provide strong evidence to support the claim that optimization is worth the additional time and effort required to obtain better designs.

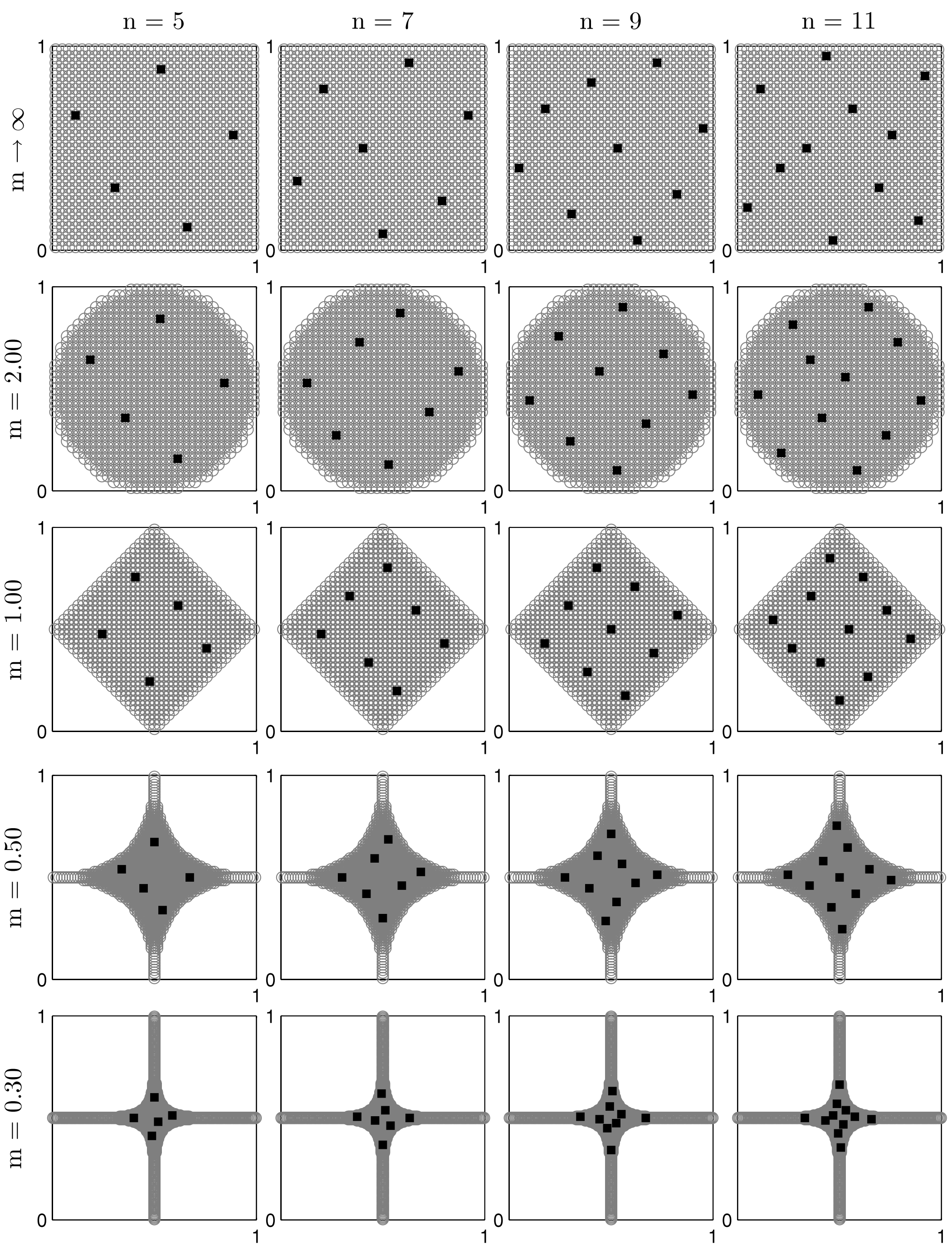

Fig. 2. Optimized designs $U(n, q^2)$ for $I = 10^6$ iterations, different values of n and m.

6. Conclusion

We analyze the construction designs for flexible regions Draper and Guttman (1986), which enclose some region of interest within a larger design space. We select points on a grid from the region of interest, resulting in U-type designs. A TA implementation is used to optimize these U-type designs. The discrepancy measure applied (CCD) is well suited for flexible regions. We consider several design configurations and flexible region shapes. The results indicate that by using the TA heuristic produces optimized, low-discrepancy designs that distribute design points uniformly over the region of interest. In some cases, particularly for small flexible regions, it might be speculated that the resulting designs are in fact already optimal designs. The same results are found for most design configurations considered.

Future research on flexible regions may follow two different approaches. The first may be to run experiments and produce results for higher design dimensions, although the absence of an analytical formula for the CCD has imposed constraints on what can be achieved due to the computational load for calculating CCD for a given design. If an analytical formula for the CCD cannot be obtained, perhaps efficient updating rules could be used when calculating the CCD value for a slightly

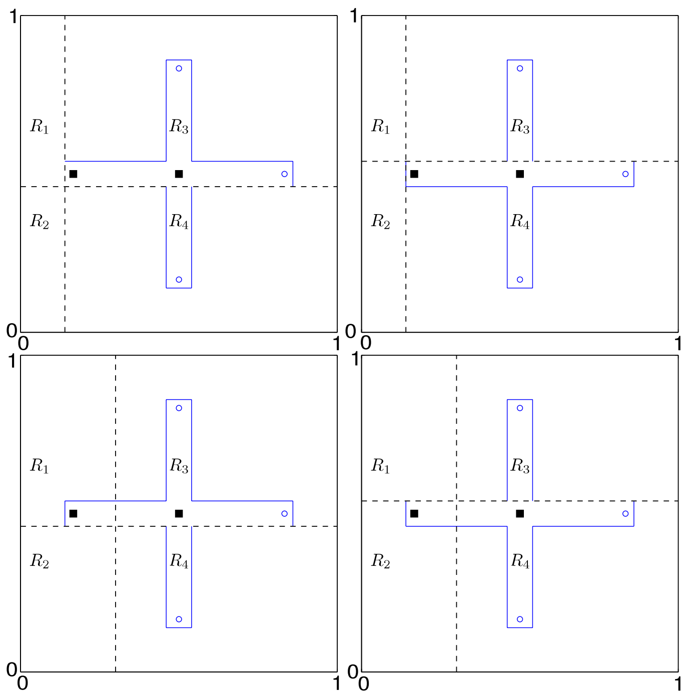

Fig. A.3. CCD calculation example, $U(2, 3^2)$.

modified design in the course of the optimization procedure. The other approach may be to apply the method to a real meta-modeling problem, such as the example described in the original paper by Draper and Guttman (1986), to demonstrate to what extent optimized U-type designs on flexible regions might improve the results.

Acknowledgements

We are thankful to two anonymous referees for their helpful comments and suggestions. Financial support from the EU Commission through MRTN-CT-2006-034270 COMISEF and the DFG through grant No. 62201080 (Mercator Lin) is gratefully acknowledged.

Appendix A. Calculating the CCD

The procedure to calculate the CCD is summarized in Algorithm 2. The following demonstrates using a simple example the workings of the algorithm. The diagrams in Fig. A.3 show a flexible region ($U(2, 3^2)$ and $m = 0.3$), its enclosing border, and the subregions for the first few moves of the region traversal. The dashed horizontal and vertical lines mark the border lines between regions. Design points are represented as black squares. The traversal of the flexible region is made by row and then column of the grid points.

Algorithm 2 CCD Calculation Pseudocode.

```
 1: initialize the grid-point and design-volume matrixes for subregions
 2: while flexible region not traversed do
 3:   if we are not on a slope of the border, advance to the next grid-point row
 4:   for grid points in a row do
 5:     if we are not on the slope of the border, update the grid-point and design-volume matrixes
 6:     calculate the discrepancy contribution at each point in the row
 7:     sum the discrepancy contributions for all points in the row
 8:   end for
 9:   sum the total discrepancy contributions for rows
10: end while
11: calculate and return the CCD
```

There are several data structures required to calculate the CCD contributions made at each grid point (1:). These are now described. A flexible region profile matrix contains the coordinates for the start of a border and the total grid points in the flexible region in that column. For the example, the matrix is {1, 2, 1; 2, 1, 3; 3, 2, 1}. Therefore the information contained in the first row tells us that the border of the flexible region at this point starts at grid level coordinates (1, 2) with one grid point in that column.

A design-point profile vector contains the number of design points in each column, which is {1, 1, 0}.

Table B.3
CCD values[a] of random designs and designs obtained from switching algorithm ($U(n, q^2)$), different values of n and m.

n	Random designs			Switching algorithm	
	1% quantile	Mean	Std. Dev.	Mean	Std. Dev.
$m \to \infty,\ q = 31$					
5	10.492	28.832	14.042	6.965	0.779
7	7.220	20.611	10.304	3.996	0.326
9	5.504	16.018	8.105	2.657	0.324
11	4.449	13.107	6.692	1.910	0.226
$m = 2,\ q = 35$					
5	15.107	44.334	22.895	9.826	1.317
7	10.421	31.746	16.853	5.685	0.750
9	7.959	24.777	13.351	3.749	0.508
11	6.451	20.300	11.023	2.740	0.367
$m = 1,\ q = 43$					
5	20.051	62.212	33.277	13.194	2.049
7	13.903	44.571	24.473	7.537	1.100
9	10.653	34.797	19.370	4.923	0.650
11	8.649	28.595	16.040	3.570	0.475
$m = 0.5,\ q = 75$					
5	17.980	60.593	33.964	11.682	1.673
7	12.546	43.675	25.131	6.562	0.878
9	9.770	34.308	19.917	4.422	0.708
11	8.006	28.299	16.501	3.192	0.491
$m = 0.3,\ q = 159$					
5	16.188	61.477	36.381	9.506	1.715
7	11.368	44.760	26.974	5.619	0.960
9	8.969	35.512	21.439	3.662	0.467
11	7.558	29.579	17.750	2.812	0.328

[a] All entries in columns 2–6 are scaled by a factor of 10^3.

Another vector contains the row coordinate of each individual design point and is grouped by column and sorted into ascending order for each column. It is {1, 1}.

A grid-point volume state matrix contains grid points at each row for all regions and is updated dynamically during the traversal. Its initial state is {0, 0, 0; 1, 3, 1}. The first column in the matrix represents Regions 1 and 2, the other column Regions 3 and 4. Similarly, there is a design-point state matrix. Its initial state is {0, 0, 0; 0, 2, 0}.

Once the data structures are initialized, the traversal of the flexible region starts. The first move (top left diagram in Fig. A.3) starts at the left extreme of the flexible region. No changes are made to the volume state matrixes. The grid-point volumes are $v(R1) = v(R2) = 0,\ v(R3) = 4$ and $v(R4) = 1$. The design-point region volumes are $N(R1) = N(R2) = 0,\ N(R3) = 2$, and $N(R4) = 0$.

The CCD contribution for all regions at a given grid point in a row (6:) is calculated by

$$\frac{1}{2^s} * \sum_{k=1}^{2^s} \left| \frac{\text{grid points in } R_k}{\text{total design points}} - \frac{\text{design points in } R_k}{\text{total grid points}} \right|^2 ,$$

which is summed for all rows in that column and then summed for each column (9:).

The horizontal border line is moved up one row, as shown in the top right diagram of Fig. A.3. Again, there are no changes made to the volume state matrixes. The grid-point volumes are $v(R1) = v(R2) = 0,\ v(R3) = 1$, and $v(R4) = 4$. The design-point volumes are $N(R1) = N(R2) = 0,\ N(R3) = 0$, and $N(R4) = 2$.

The vertical region border is moved past the first grid-point column, and the horizontal border is moved to below the border of the flexible region, as shown in the bottom left diagram of Fig. A.3. Here a slope is detected and indicates that the state of the grid-point volume matrix requires updating, but only after the horizontal border is moved over the first grid point. The grid-point region volumes are $v(R1) = 1,\ v(R2) = 0, v(R3) = 3$, and $v(R4) = 1$. The design-point region volumes are $N(R1) = 1,\ N(R2) = 0,\ N(R3) = 1$, and $N(R4) = 0$.

The horizontal border is moved up a row, as shown in the diagram at the bottom right corner of Fig. A.3. The new state of the grid-point data structure is {0, 1, 0; 1, 2, 1}. The updating process is done dynamically as the horizontal border is moved over the grid points, which means that the volume state data is correct for the two adjacent horizontal regions below the horizontal border. The remaining regions can be calculated by summing volumes in the flexible region profile matrix (where the volumes on either side of the vertical border are stored) and subtracting them from the other horizontal region volumes.

A design point also lies in this column, and as the horizontal border moves over the design point – as identified in the design row vector – the state of the design-volume matrix is updated. It is {0, 1, 0; 0, 1, 0}. As with the grid-point volumes, we have only design volumes for two adjacent regions. The calculations for the other regions are done in the same way as those described for the grid-point volumes.

The grid-point volumes become $v(R1) = 1,\ v(R2) = 0,\ v(R3) = 3$ and $v(R4) = 1$. The design-point region volumes are $N(R1) = 1,\ N(R2) = 0,\ N(R3) = 1$, and $N(R4) = 0$.

The whole process is repeated until the right extreme of the flexible region is reached and its CCD calculation is completed.

Appendix B. Results for random designs

See Table B.3.

References

Chuang, S.C., Hung, Y.C., 2010. Uniform design over general input domains with applications to target region estimation in computer experiments. Computational Statistics and Data Analysis 54, 219–232.

Draper, N.R., Guttman, I., 1986. Response surface designs in flexible regions. Journal of the American Statistical Association 81, 1089–1094.

Draper, N.R., Lawrence, W.E., 1965. Designs which minimise model inadequacies: Cuboidal regions of interest. Biometrika 52, 111–118.

Draper, N.R., Lawrence, W.E., 1966. The use of second-order 'spherical' and 'cuboidal' designs in the wrong regions. Biometrika 53, 496–599.

Dueck, G., Scheuer, T., 1990. Threshold accepting. A general purpose optimization algorithm superior to simulated annealing. Journal of Computational Physics 90, 161–175.

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

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

Fang, K.-T., Li, R., Sudjianto, A., 2006. Design and Modeling for Computer Experiments. Chapman & Hall/CRC, Boca Raton, FL.

Fang, K.-T., Lu, X., Winker, P., 2003. Lower bounds for centered and wrap-around l2-discrepancy and construction of uniform designs by threshold accepting. Journal of Complexity 19, 692–711.

Fang, K.-T., Ma, C.-X., Winker, P., 2002. Centered l_2-discrepancy of random sampling and latin hypercube design, and construction of uniform designs. Mathematics of Computation 71, 275–296.

Fang, K.-T., Maringer, D., Tang, Y., Winker, P., 2005. Lower bounds and stochastic optimization algorithms for uniform designs with three or four levels. Mathematics of Computation 75 (254), 859–878.

Fang, K.-T., Winker, P., 1997. Application of threshold accepting to the evaluation of the discrepancy of a set of points. SIAM Journal on Numerical Analysis 34 (5), 2028–2042.

Gilli, M., Winker, P., 2007. Editorial—2nd special issue on applications of optimization heuristics to estimation and modelling problems. Computational Statistics and Data Analysis 52 (1), 2–3.

Hickernell, F.J., 1996. A generalized discrepancy and quadrature error bound. Mathematics of Computation 67, 299–322.

Huang, M.-N.L., Lee, C.-P., Chen, R.-B., Klein, T., 2010. Exact D-optimal designs for a second-order response surface model on a circle with qualitative factors. Computational Statistics and Data Analysis 54 (2), 516–530.

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

Winker, P., 2001. Optimisation Heuristics in Econometrics: Applications of Threshold Accepting. Wiley, Chichester.

Winker, P., 2006. The stochastics of threshold accepting: Analysis of an application to the uniform design problem. In: Rizzi, A., Vichi, M. (Eds.), COMPSTAT 2006, Proceedings in Computational Statistics. Physica, Heidelberg, pp. 495–503.