Online ISSN: 2367-5357
DOI: 10.7546/itc-2019-0019

Scanning of Elongated Constrained Areas in Mixture Spaces

K. Stoyanov

Key Words: *Scanning algorithm; grid search optimization algorithm; experiments with mixtures; constrained mixture spaces.*

Abstract. *The scanning of factor spaces (both process and mixture), usually is referred to as a technique for creation of a set of points. This set can be applied as the domain, where the values of some response function are calculated. Alternatively this can be the set of candidate points, used for searching of optimal experimental designs. In many practical applications, where some constraints over the components are imposed, the width of the subregion along the components, can differs significantly. In such cases the regions of interest obtain complicated, elongated shape. A scanning algorithm, which provides options to choose different steps over the components, will save both computer resources for multicomponent (multifactor) tasks as well will provide regular (or at least controlled) distribution of the points across the scanned region. While the scanning, with different step lengths in an Euclidean coordinate space is trivial, the scanning of constrained mixture regions rises some algorithmic difficulties. The article presents an approach for scanning of* $q \geq 2$ *– component constrained mixture factor spaces. The steps length or number over each of the components can be chosen independently. The only condition, imposed by the requirement* $\sum_{i=1}^{q} x_i = 1$*, is the necessity the scanning along two of the components, to be performed with equal steps. The implementation of the algorithm is illustrated with practical examples and a C++ program listing.*

1. Introduction

In the mixture experiments the following linear restrictions take place:

(1) $0 \leq x_i \leq 1, i = 1, 2, \ldots, q,$ and $\sum_{i=1}^{q} x_i = 1,$

where x_i represents the proportion of the i^{th} component, q is the number of the components. Often some additional constraints, imposed by theoretical, technological or economical considerations present:

(2) $0 \leq l_i \leq x_i \leq u_i \leq 1, i = 1, 2, \ldots, q,$

where l_i and u_i are constants.

The space defined by (1) defines a mixture space. The region of the mixture space defined by (2) is referred to as constrained mixture region.

The consistent (see [1]) constraints define a convex polyhedron, where all of the points satisfy the constraints (1) and (2).

The figure, defined by consistent linear constraints (1) and (2) has a shape of a convex $q - 1$ dimensional multiplane, constrained by $q - 2$ dimensional hyperplanes, which in turn are constrained by $q - 3$ dimensional hyperplanes, etc., all the way down to the edges, which are one dimensional and the vertices, which following this logic, can be considered as zero dimensional hyperplanes.

Good references concerning the geometry of the constrained mixture spaces one can find in [4, 5]. A complete reference of the present methods applied in the modelling of mixtures is presented in [1, 8].

Additionally to Eqs. (1) and (2) there exist also widespread situations where:

(3) $u_i - l_i \gg u_j - l_j, i \neq j.$

Some of the constraints are significantly wider that the others.

The inequality (3) constitutes constrained regions having an elongated shape. This type of constraints are related with a wide range of chemical, pharmaceutical, metallurgical etc. experimental problems. They take place especially in the cases where small quantities of high reactive components are under consideration – inhibitors, catalysts, etc. Apparently despite the narrow ranges along some of the axes, the investigator is interested to explore all sub-areas defined by Eqs. (2) and (3) with equally spread density of the experimental points.

According to the Response Surface Modelling (RSM) practice, during the investigation of some mixture, a statistical model, intended to describe the dependence of the responses from the proportions of the mixture components is derived [8, 10-12]. Using this model, a detailed numerical (or *in silico*) exploration of the response surface over the constrained region is possible. One can find a large number of appropriate optimization techniques. Generally a proper adaptation of the commonly used techniques is necessary to make them applicable for the mixture set up (1)-(2). An inclusive description of such techniques one can find in [2]. Exhaustive review of the mixture related publications, spanning 50 years, is provided by Piepel [12].

It is well known that most of the optimization methods assure only the local optima discovery. In fact only the application of some of the variations of the scanning techniques (also known as “grid search”), assures that the global extremum, after a moderate number of calculations, can be found. The common insufficiency of the scanning techniques is the number of calculations, necessary to be performed. It grows rapidly with the number of the dimensionality of the task. This is the main reason why one chooses another less effective but more economical technique.

The quantity of the calculations mainly depends on the number of steps, necessary for the scanning process, which needs to be performed along the factors. Decrease of the number of the necessary steps would avoid the main hurdle for using these methods. The advantage of the minimizing the number of the steps becomes even more apparent in the multiresponse problems, especially in the case where several numerically heavy functions (e.g., for compromise optimization) have to be calculated at each node of the scanning lattice.

The scanning is closely related also with another class of problems – the search for optimal experimental designs in discrete spaces. Widely spread approach is the construction of optimal experimental designs (exact or sequential) by choosing the points of the design, from a previously created set of candidate points. See for instance the seminal works of T. J. Mitchell [9] and V. V. Fedorov [14], see also [13]. In the cases where constrained mixture spaces are under consideration, the size of the candidate set and the position of the points within the region influence significantly the success of the search of the optimal design. It is crucial for the efficiency of the algorithm, the candidate points to be distributed as evenly as possible across the region. Any clustering or aggregating of the points will bias the algorithm towards some particular areas or will cause singularity of the information matrix.

Choosing of arbitrary steps along the axes in scanning of Euclidean (non-mixture) factor spaces is a trivial task, see *figure 4,* where both factors are scanned with different step lengths. This makes possible, three steps to be performed along each factor.

Imposing the constraints (1) make the same task (scanning with arbitrary step lengths) along the factors, algorithmically complicated. Without appropriate algorithm, one is restricted to perform the scanning with equal steps along the factors. This creates crowding of the points along some axes, at the expense of other axes (*figure 5*). This can be compared with *figure 2,* where the points are evenly distributed along the factors.

The aim of the presented work is to present an algorithm and C++ program, for scanning of a q-dimensional mixture space. The method allows the investigator independently to choose the step length or the number of steps along each of the axes. The only limitation, provoked by the nature of the coordinate system, defined by Eq. (1), is the necessity that two of the factors should have equal step length.

Section 2 presents the idea of scanning and its application in mixture problems, where the requirement for different step lengths is imposed. A step by step description of the algorithm is provided. **Section 3** presents two illustrative examples – scanning of three and four components mixtures. While the former uses artificial data, the latter example uses the data of McLean and Anderson [3].

The application of the algorithm is illustrated with a C++ program, provided in the **Appendix**.

2. Scanning of constrained mixture regions using adaptive steps over the components

2.1 Equal vs different steps

According to the on-line Merriam-Webster dictionary [15], the meaning of “scan” is “to examine by point-by-point observation or checking” and also to examine systematically (as by passing a beam of radiation over or through) in order to obtain data especially for display or storage. For the purpose of the present work we will refer to “scanning” as a process of creating of set of points, by calculation of their coordinates, where (1)-(2) hold. These points should be distributed across the region of interest as even as possibly. One approach to scan a specific region, is to generate randomly as much as possible evenly distributed points. Another more systematic approach, which is considered here, is to start with a point $\vec{x}$, having coordinates which satisfies the constraints (1, 2) and the requirements for consistency [6]. After that by systematic increasing of the coordinates to create the members of R.

We will illustrate the idea for by starting with Euclidean (non-mixture) space.

Suppose we have to investigate a situation with

Euclidean space, and we are not able to choose the length of each step independently from the other. In this case one have to choose one of two options – (1) both step lengths should equal the length of the first one or (2) both step lengths should equal the length of the second one, see *figure 1* and *figure 2*.

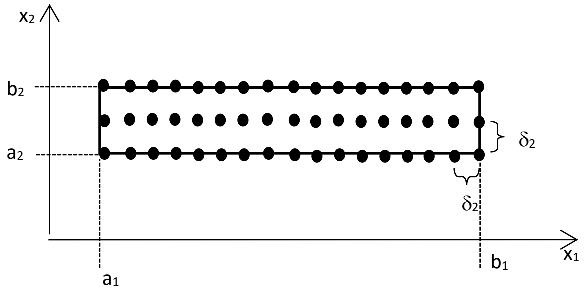

Figure 1. Euclidean (non-mixture) space, for two dimensions, where both lengths of the scanning steps equal to the smaller one (*δ*2)

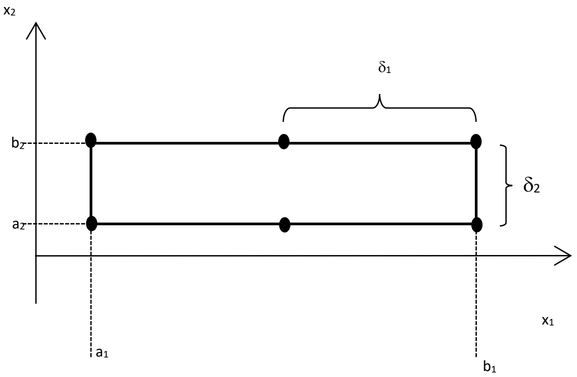

Figure 2. Euclidean (non-mixture) space, for two dimensions, where both lengths of the scanning steps equal to the <u>bigger</u> one (δ_1)

To control the number of the points, during the scanning process, one needs to be able to use different and mutually independent steps along the axes. It is trivial to implement this requirement for non-mixture factor spaces, but this is not quite the case in mixture environments.

Let's assume that we have a two factors, "non-mixture" problem and we are restricted with the number of the points to be generated and we want to perform exactly three steps per factor. This means to determine the coordinates of three points along each axis, totally making nine points (see *figure 3*).

By forcing the condition of equal steps, one will have troubles to perform the desired number of steps. Too many in the case where the smaller step is accepted, or too few, in the ca case with the bigger one.

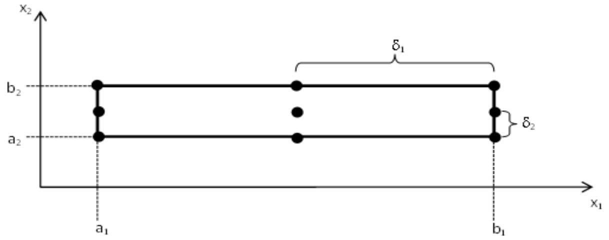

Figure 3. Scanning of Euclidean (non-mixture) factor space with different steps along the factors

Considering the case with equal steps, one will perform much more (probably unnecessary) calculations along x_1, then along x_2, (*figure 1*). If the investigator decides to decrease the number of the unnecessary steps, then one can choose both step lengths to become equal to the biggest one (δ_1). In this case along x_1, there will be three steps, but along x_2, there will be only two – remember, the condition is to use only steps equal to δ_1. As long $\delta_1 > \delta_2$, a step, equal to δ_1 will create a point along x_2 above b_2, one assign the value of the upper bound – b_2. The result is that along x_2 one will perform only two calculations and one step, see *figure 3*.

Apparently the flexibility of different length of the steps is important, but while to accomplish this is a trivial task in *non-mixture*, situation, the imposing of the constraints (1), combined with Eq. (3), namely when we have to investigate mixtures creates algorithmic problems, which will be addressed here.

To scan of a mixture subregion, with different steps, even if it has elongated shape, in a way similar to the one, as shown on *figure 4*, will result to the situation as shown on *figure 2*. In this case the same region will be scanned by using only two steps (three calculations) along each component, despite of the difference in the width of the ranges.

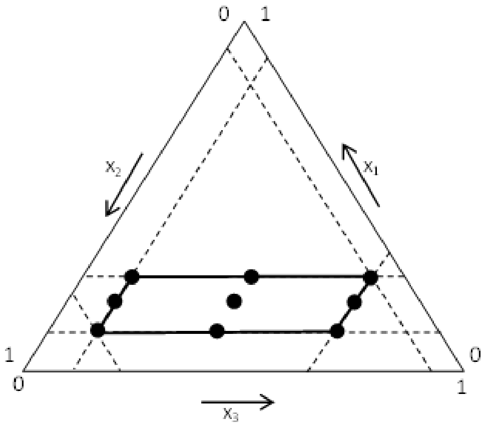

Figure 4. Scanning mixture constrained region with <u>different</u> steps along the variables

The mixture analog of the non-mixture situation, shown on *figure 1*, can be observed on *figure 5*. The requirement to perform “at least three steps” along x_1, forces us to perform eleven calculations (ten steps) along x_2 and eight calculations (nine steps) along x_3.

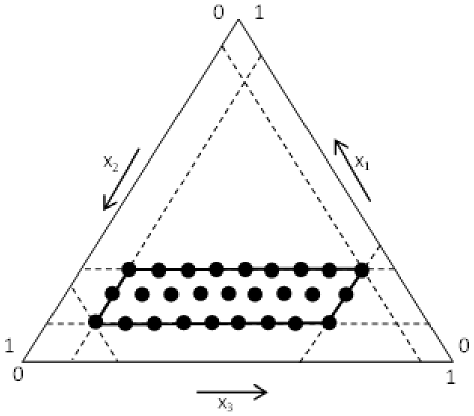

Figure 5. Scanning mixture constrained region with <u>equal</u> steps along the variables

2.2 The algorithm outline

The idea of the algorithm will be described by considering of a hypothetical four component mixture, where the following constraints hold:

$$\begin{aligned} l_1 &\le x_1 \le u_1 \\ l_2 &\le x_2 \le u_2 \\ l_3 &\le x_3 \le u_3 \\ l_4 &\le x_4 \le u_4 \end{aligned} \tag{4}$$

Firstly one will scan a two-dimensional plane, determined by the components x_1, x_2 and x_3, assuming that $x_4 = l_4$. See the plane ***A, B, C, D***. This means that a set of points, which lies over this surface (the plane ***A-D***) will be created. The process of scanning of the plane x_1, x_2 and x_3 (the plane ***A-D***) is illustrated on *figure 6*.

To scan this it, one need to start with scanning a line segment, coinciding with one of its edges, than to translate geometrically the line segment towards the opposite edge, to scan the line segment again etc. until the opposite edge of the surface of the two dimensional hyperplane is reached.

Let us consider a line segment $a - d$ parallel to the x_2 axis (see *figure 6*). For clarity on *figure 6* the line segment $a - d$ is drawn in the general case parallel to ***A-B***. To scan the line segment $a - d$ one proceeds in the following manner:

- choose a step with length δ_{12} where $\delta_{12} > 0$;

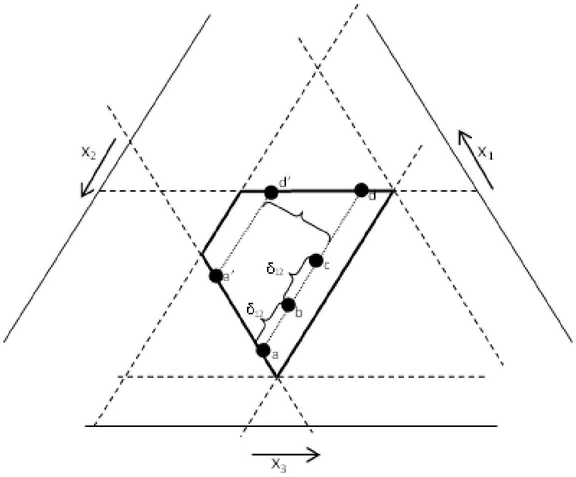

Figure 6. Scanning of mixture region with <u>different</u> steps

- choose an initial point, In this case the point is $a[x_1^a, x_2^a, x_3^a, x_4^a \equiv l_4]$. We assume that the coordinates of the initial point holds the constraints (1) and (2). Apparently in the beginning $a \equiv \boldsymbol{A}$;

- perform a step along the line $a - d$ creating point b, having coordinates $b[x_1^b, x_2^b, x_3^b, x_4^b \equiv l_4]$, calculated by:

$$\begin{aligned} x_1^b &= x_1^a + \delta_{12} \\ x_2^b &= x_2^a - \delta_{12} \\ x_3^b &= x_3^a \end{aligned} \tag{5}$$

- perform the next step along the line $a - d$ by calculating the coordinates $c[x_1^c, x_2^c, x_3^c, x_4^c \equiv l_4]$ of the next point c:

$$\begin{aligned} x_1^c &= x_1^b + \delta_{12} \\ x_2^c &= x_2^b - \delta_{12} \\ x_3^c &= x_3^b \end{aligned} \tag{6}$$

- the steps will continue in a similar way, until some of the coordinates outsteps some of the constraints (2).

In this case the scanning of the line segment will finish at point d (see *figure 6*), where the upper limit u_2 will be reached. One can conclude that a set of points $a, b, c, \dots, d$ over the line segment $a - d$ has been generated.

Next stage is to start scanning along the next dimension of the surface. This will mean to move the line segment $a - d$, parallel to itself, towards ***DC***, along x_2 and to scan it in the same manner as discussed here.

We will do this, by creating of a new line segment $a' - d'$, parallel to $a - d$. This can be done by performing a step with length δ_3, along the axis x_3 of a, creating in this way the point a'. After that, a similar to the above described way, a scanning along $a' - d'$ can be performed. The performance of additional steps δ_3 by constructing of additional lines parallel to $a - d$ and $a' - d'$ will continue until the upper limit u_3 along the x_3 axis is reached.

Once the line segment reaches the upper limit of the plane, then the scanning over the plane will be finished. The parallel move of the line $a - d$, combined with the scanning over it, in fact is equivalent to a scanning along the third mixture component, namely x_3.

Assuming that, at the beginning, this scanning started at the lower limits of x_1, x_2 and x_3 (the edge ***AB***) and finished at the respective upper limits (the edge ***DC***), one can conclude that a scanning of the two-dimensional hyperplane defined by:

$$(7)\quad \begin{aligned} &l'_1 \leq x_1 \leq u'_1 \\ &l'_2 \leq x_2 \leq u'_2 \\ &l'_3 \leq x_3 \leq u'_3 \\ &l'_4 = x_4, \end{aligned}$$

has been performed.

Further, in the same manner, it is possible, by performing steps δ_4 along the axis x_4, to translate geometrically the hyperplane, defined by (7). All of the newly generated points need to respect the constraints [2].

There are not any special requirements for the values of δ_{12}, δ_3 and δ_4, hence it is possible that $\delta_{12} \neq \delta_3 \neq \delta_4$.

It is important to consider the irregular shape of the constrained region. During its move parallel to itself, the line segment $a - d$ must change its length in order to keep the boundaries. In other words, considering the shape of the region the length of $a' - d'$ should be different than the length of $a - d$.

In multidimensional problems where $q > 3$, the shape of the constrained region becomes very complicated, in some subregions new additional constraints, different to the initial (e.g., Eq. (2)) ones will appear. For these subregions the initial constraints will become inconsistent (see [6]) and have to be replaced by new constraints, designated by l' and u', where following inequalities will take place:

$$(8)\quad \begin{aligned} &l' \geq l \\ &u' \leq u. \end{aligned}$$

Because of this, some difficulties in the choice of the step lengths, can be expected. One can handle this by choosing the following approaches:

1. *Number approach.* Initially to choose some appropriate <u>*number*</u> of the steps along the components and to calculate the step <u>*lengths*</u> according to the particular values of the l' and u';

2. *Length approach.* Initially to choose an appropriate, constant step <u>*lengths*</u>, along the each of the components than to calculate the respective <u>*number*</u> of steps.

In the cases where the shape of the region is not too elongated, these two approaches give similar results.

If the <u>*number*</u> approach is solely considered, before to proceed with the i^{th} component x_i, one needs to calculate the respective step length δ_i:

$$(9)\qquad \delta_i = (u'_i - l'_i / s_i)$$

where s_i designates the chosen number of steps along the i^{th} component.

If the <u>*length*</u> approach is chosen, then before to proceed with each of the steps δ_j, one has to check for possible outstep of the respective upper limit u_j. Hence to check whether the following condition holds:

$$(10)\quad x_j + \delta_j \leq u'_i.$$

It is worth to mention that if one accepts only the δ_i <u>*length of the steps*</u> approach, there is a great possibility to loose (not to create points in) the regions where the condition $u'_i - l_i \ll \delta_i$ holds. In the case where only the <u>*number of steps*</u> approach is chosen, one will be forced to perform the same number of steps within regions having in any size. For instance the number of steps for the case where $u'_i - l_i < 0.05$ will be equal to the case where $u'_i - l_i > 0.5$.

To avoid this, we have combined these approaches. One starts with <u>*constant length*</u> and continues, until the consistent upper bound u' is reached. Always before to perform the first step along some component, one checks the possibility to perform some given number of (i.e., three) steps In the cases where this is impossible, the step is adapted and the length of the step is recalculated by using $\delta_i = (u'_i - l_i)/2$.

The scanning and the recalculation of the step lengths continues until the condition

$$(11)\quad \delta_i \leq \epsilon$$

for the $i^{th}, i = 1, \dots . q$ component is fulfilled. Here by ϵ we represent the desired precision of the scanning. The scanning precision value could be assigned based on practical reasons (e.g., required accuracy to locate the extremum) or the computer zero [7] can be used.

Once the condition (11) is fulfilled, on can conclude that the scanning over the i^{th} component is finished.

The algorithm, logically can be divided on two main parts:

- initialization of the scanning,
- performing of the scanning.

During the *initialization* stage, all of the components take their initial values. At the beginning of the scanning along the component i_s, all components having numbers i, where $3 \geq i > i_s$ receives values equal to their lower constraints.

Then the respective lower bound of the component, referred to as *number 1* is assigned. Once all components except x_2 have values, the value of the component *number 2* is calculated by using:

$$x_2 = \sum_{\substack{i=1\\ i\neq 2}}^{q} x_i. \qquad (12)$$

For each of these components, the consistent upper constraint is calculated:

$$\{u_i' = min[u_i; (1 - \textstyle\sum_{j=i+1}^{q} l_j)], i = 3, i_s - 1\}, i_s = 3, q. \qquad (13)$$

Before to proceed with the performing of the scanning, when the consistent upper and lower bounds are calculated, one needs to check whether the scanning over the region just defined will be sensible. The scanning will not be sensible if the points to be generated, appears to be too close to each other (similar to points 1-3 on *figure 8*). The sensibility depends on considerations related with the accuracy of the extremum localization (for optimization tasks) or the distance between the points (for generation of candidate sets to be used for search of optimal designs). Also before to proceed with some step, one checks for passing over the respective upper or lower constraint.

In the case where it is insensible, some new steps to be performed or the next step would overstep the respective bound, then:

- if $u_i' - x_i \leq \epsilon$ the respective component receives the value of the i^{th} *upper* constraint;
- if $x_i - l_i \leq \epsilon$ the respective component receives the value of the i^{th} *lower* constraint.

The algorithm continues with increasing the values of the respective coordinates with the values of the respective steps. The scanning over the first and second component proceeds in a reciprocal way. At each step one increases the value of the first component with δ_{12} and decreases the value of the second component with the same value.

$$\begin{aligned} x_1' &= x_1 + \delta_{12} \\ x_2' &= x_2 - \delta_{12} \qquad (14) \\ x_i' &= x_i + \delta_i; i = 3, \dots, i_s; i_s = 3, \dots, q. \end{aligned}$$

Because the embedded adjustment of the bounds, the algorithm is applicable also for scanning of subregions having inconsistent bounds (see [6]). The algorithm automatically adjusts the bounds of the convex hyper polyhedron.

3. Illustrative examples

3.1 Three components mixture

In *figure 7* the constrained region of three components mixture is presented.

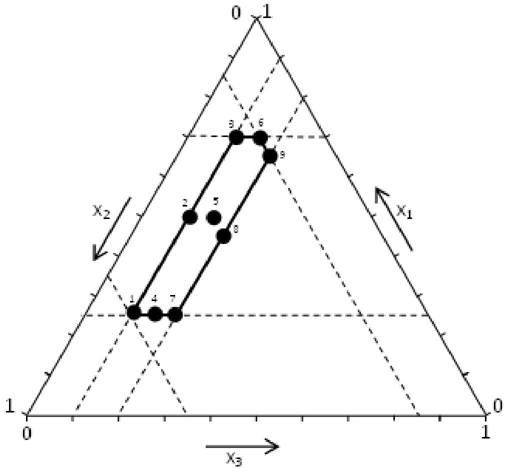

Figure 7. Scanning of a two-dimensional plane. Initially the line segment $a - d$ will be scanned, by generating the points b and c, by increasing one of the coordinates by δ_{12}. Then the line segment is translated geometrically to $a' - d'$, where the scanning proceeds similarly.

The following constraints take place over the variables:

$$\begin{aligned} 0.25 \leq x_1 \leq 0.70 \\ 0.15 \leq x_2 \leq 0.65 \qquad (15) \\ 0.10 \leq x_3 \leq 0.20 \end{aligned}$$

One has decided to scan this region by using three steps over each of the components. Considering the constraints, the step for the first and second components, namely x_1 and x_2 is $\delta = 0.25$. The step chosen for the third component x_3 is $\delta_3 = 0.05$. In the same figure one can see the points, calculated during the scanning process. The numbers next to the points, correspond to the numbers in *table 1*.

Table 1. Three components example

No.	x_1	x_2	x_3	No.	x_1	x_2	x_3
1	0.250	0.650	0.100	6	0.700	0.150	0.150
2	0.475	0.425	0.100	7	0.250	0.550	0.200
3	0.700	0.200	0.100	8	0.450	0.350	0.200
4	0.250	0.600	0.150	9	0.650	0.150	0.200
5	0.475	0.375	0.150				

The steps of the algorithm for the scanning process are as follows:

1. The initial data set is assigned (number of variables, constraints, steps length or steps number);

2. Variables x_1 and x_3 receive the values of their lower constraints;

3. By using (12), the value of x_2 is calculated;

4. The point obtained in such way is designated as point No. **1**;

5. One checks the possibility to perform at least three steps, which will satisfy:

$$\begin{aligned} &\frac{u_1 - x_1}{\delta_{12}} + 1 \geq 3 \\ &\frac{x_2 - l_2}{\delta_{12}} + 1 \geq 3 \end{aligned} \tag{16}$$

6. The result is as follows:

$$\frac{u_1 - x_1}{\delta_{12}} + 1 = \frac{0.65 - 0.15}{0.5} + 1 = 1.8 < 3 \tag{17}$$

7. The result of condition (16) appears to be FALSE, hence it is not possible to perform three steps as required. The value of δ_{12} must be recalculated:

$$\begin{aligned} \delta_{12} &= \frac{min[(u_1 - x_1),(x_2 - l_2)]}{2} = \\ &= \frac{min[(0.7 - 0.25),(0.65 - 0.15)]}{2} = \\ &= \frac{min[0.45{,}0.50]}{2} = \frac{0.45}{2} = 0.225 \end{aligned} \tag{18}$$

Hence the new value of δ_{12} will be 0.225.

8. One performs two steps along x_1 and x_2. The value of the x_1 increases, the value of x_2 decreases, while the value of x_3 remains the same. In this manner one calculates the points designated as No. **2** and No. **3**.

9. In point **3**, the component x_1 reached its upper constraint, so the scanning along this line is completed;

10. Next task is to perform some steps along x_3. So, one needs to examine the possibility to perform at least three steps at length δ_3 along x_3. The trueness of the condition $\frac{u_3 - x_3}{\delta_3} + 1 \geq 3$ should be checked. So

$$\frac{u_3 - x_3}{\delta_3} + 1 = \frac{0.2 - 0.1}{0.05} + 1 = 3$$

11. The condition is TRUE, therefore it is possible to continue with the steps along x_3.

12. The lower constraint is assigned to x_1 and the value of x_2 is calculated by using (12).

13. The newly generated point satisfies conditions (1) and (15), this can be the next point of the scanning, designated as No. **4**.

14. One reapplies stages 5 ÷ 8 so the points No. **5** and No. **6** are generated.

15. The procedure continues from Stage 12 and the points No. **7**, No. **8** and No. **9** are determined in a similar manner.

Actually, at point No. **9** the lower constraint of x_2 is reached and at the same time no further steps along x_3 are possible. As long there is no fourth component, the scanning appears to be finished.

The region has complicated shape, which is not a paralleloid. To illustrate the influence of the non-pralleloidal shape, we will apply the procedure to the same problem, but with alternative ordering of the components. The set of the generated points is given on *table 2* and *figure 8*.

Table 2. Three components example
with alternative ordering of the components

No.	x_1	x_2	x_3	No.	x_1	x_2	x_3
1	0.650	0.200	0.150	5	0.450	0.150	0.400
2	0.675	0.175	0.150	6	0.500	0.100	0.400
3	0.700	0.150	0.150	7	0.250	0.100	0.650
4	0.400	0.200	0.400				

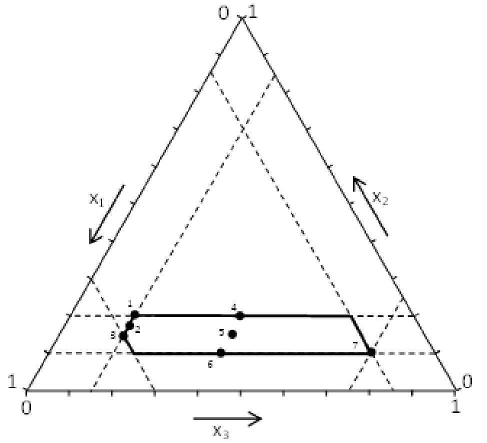

Figure 1. The same three component example, but with different order of the components

The reordered set of constraints is as follows:

$$\begin{aligned} &0.25 \;\le\; x_1 \;\le\; 0.70 \\ (19)\quad &0.10 \;\le\; x_2 \;\le\; 0.20 \\ &0.15 \;\le\; x_3 \;\le\; 0.65 \end{aligned}$$

The steps, chosen for scanning are $\delta_{12} = 0.05$ and $\delta = 0.25$, respectively. Considering the respective constraints, this particular choice of the value for δ_{12} assures at least three steps over x_1 and moderate number of steps over x_2.

One can see that in this case, that only seven point are enough for the scanning of the same region. This is because of the fact that in point **7**, the *consistent* upper constraint of x_2 coincide with the respective lower constraint, so the line degenerates to a vertex. In addition one can see from *figure 8*, that upon the edge at $x_3 = 0.15$, there are three points which appeared quite closer to each other.

3.2. Four components mixture based on the McLean and Anderson data

The application of the algorithm for problems with more than three components is presented by using the data used in the seminal paper of McLean and Anderson [3]. In this paper they present a method for constructing of experimental design for four component mixture region, having the following constraints:

$$\begin{aligned} &0.40 \le x_1 \le 0.60 \\ &0.10 \le x_2 \le 0.50 \\ (20)\quad &0.10 \le x_3 \le 0.50 \\ &0.03 \le x_4 \le 0.08 \end{aligned}$$

As it is reported in [6] the constraints (20) are inconsistent. For instance no points exists, having coordinates $x_2 = 0.5$ or $x_3 = 0.5$. Also one can see that the region has very narrow interval over x_4, namely $x_4 = 0.08 - 0.03 = 0.05$, compared with the others – 0.2 over axis x_1 and 0.4 over axes x_2 and x_3.

We choose the following steps:

$$(21)\quad \delta_{12} = 0.2,\ \delta_3 = 0.1,\ \delta_4 = 0.02.$$

As the step over the first component is too large to provide at last three steps, the algorithm decreases the value of δ_{12}. The points earned after the scanning are shown in *table 3.*

To emphasize the efficiency of this method, the same region can be scanned with equal steps. To get enough coverage of the component x_4 one needs to perform at least three steps over it by using a step length $\delta = 0.02$. The scanning over all components with such step would provide 629 points. This will provide the same quantity of information as the one provided by 48 points.

Conclusion

An algorithm and a C++ program for generating of a set of points by scanning of elongated mixture regions is presented. The term “elongated” refers to the problems, where some of the components has very narrow range, compared with the other. The possibility to choose the step over each of the components provides a flexibility and makes it possible to control the number of the generated points. The alternative approach of using equal steps over all components, which algorithmically is much simpler, provides ether too large number of points along the components with large range or too few along the components with narrow range. For instance, it is demonstrated with the example data of McLean and Anderson [3] that a similar quantity of information (evenly coverage of the constrained region) can be reached with 692 points, by using the classical approach and 49 points, using the approach, presented here.

The proposed algorithm can be applied for two classes of problems:

- Optimization problems in regression context, related with the search of the extremum of some function $\hat{y} = f(\vec{\beta}, \vec{x})$ over the constrained region.

- Problems related with the search of optimal designs (e.g., exchange or Wynn-Fedorov [13] algorithms in discrete spaces), where a candidate set of points has to be generated.

The algorithm is illustrated by using a step-by-step explanation and is supported with practical examples.

Table 3. Four components example

No.	x_1	x_2	x_3	x_4	No.	x_1	x_2	x_3	x_4	No.	x_1	x_2	x_3	x_4
1	0.400	0.470	0.100	0.030	17	0.500	0.250	0.200	0.050	33	0.530	0.100	0.300	0.070
2	0.500	0.370	0.100	0.030	18	0.600	0.150	0.200	0.050	34	0.400	0.130	0.400	0.080
3	0.600	0.270	0.100	0.030	19	0.400	0.250	0.300	0.050	35	0.415	0.115	0.400	0.080
4	0.400	0.370	0.200	0.030	20	0.475	0.175	0.300	0.050	36	0.430	0.100	0.400	0.080
5	0.500	0.270	0.200	0.030	21	0.550	0.100	0.300	0.050	37	0.400	0.420	0.100	0.080
6	0.600	0.170	0.200	0.030	22	0.400	0.150	0.400	0.050	38	0.500	0.320	0.100	0.080
7	0.400	0.270	0.300	0.030	23	0.425	0.125	0.400	0.070	39	0.600	0.220	0.100	0.080
8	0.485	0.185	0.300	0.030	24	0.450	0.100	0.400	0.070	40	0.400	0.320	0.200	0.080
9	0.570	0.100	0.300	0.030	25	0.400	0.430	0.100	0.070	41	0.500	0.220	0.200	0.080
10	0.400	0.170	0.400	0.030	26	0.500	0.330	0.100	0.070	42	0.600	0.120	0.200	0.080
11	0.435	0.135	0.400	0.030	27	0.600	0.230	0.100	0.070	43	0.400	0.220	0.300	0.080
12	0.470	0.100	0.400	0.030	28	0.400	0.330	0.200	0.070	44	0.460	0.160	0.300	0.080
13	0.400	0.450	0.100	0.050	29	0.500	0.230	0.200	0.070	45	0.520	0.100	0.300	0.080
14	0.500	0.350	0.100	0.050	30	0.600	0.130	0.200	0.070	46	0.400	0.120	0.400	0.080
15	0.600	0.250	0.100	0.050	31	0.400	0.230	0.300	0.070	47	0.410	0.110	0.400	0.080
16	0.400	0.350	0.200	0.050	32	0.465	0.165	0.300	0.070	48	0.420	0.100	0.400	0.080

Acknowledgements

This paper is partially supported by the National Scientific Program “Information and Communication Technologies for a Single Digital Market in Science, Education and Security (ICTinSES)”, financed by the Ministry of Education and Science.

References

1. Cornell, J. A. Experiments with Mixtures: Designs, Models and Analysis of Mixture Data. New York, John Willey and Sons, 2011.
2. Ionchev, H. A. Optimization of Multicomponent Systems. Sofia, Technica, 1982 (in Bulgarian).
3. McLean, R. A., Anderson, V. L. Extreme Vertices Design of Mixture Experiments. – *Technometrics*, 8, 1966, 447–454.
4. Crosier, R. B. Mixture Experiments: Geometry and Pseudocomponents. – *Technometrics*, 26, 1984, No. 3, 209–216.
5. Crosier, R. B. The Geometry of Constrained Mixture Experiemtns. – *Technometrics*, 28, 1986, No. 2, 95–102.
6. Piepel, G. F. Defining Consistent Constraint Regions in Mixture Experiments. – *Technometrics*, 25, 1983, 97–101.
7. Malcolm, M. A., Moller, C. B., Forsyth, G. E. Computer Methods for Mathematical Computation. Prentice-Hall, Inc, 1977.
8. Vuchkov, I. N., Ionchev, H. A. Design and Analysis of the Experiment in the Investigation of the Properties of Mixtures and Alloys. Sofia, Technica, 1979 (in Bulgarian).
9. Mitchell, T. J. An Algorithm for the Construction of D-optimal Experimental Designs. – *Technometrics*, 42, 2000.
10. Montgomery, D. C. Design and Analysis of Experiments. John Wiley & Sons, 2017.
11. Khuri, A. I., Cornell, J. A. Response Surfaces: Designs and Analyses. Routledge, 2018.
12. Khuri, A. I. Response Surface Methodology and Related Topics. World Scientific, 2006.
13. Mandal, A., Wong, W. K., Yu, Y. Algorithmic Searches for Optimal Designs. Handbook of Design and Analysis of Experiments, 2015, 755–783.
14. Fedorov, V. V. Theory of Optimal Design. New York, Academic, 1972.
15. https://www.merriam-webster.com/dictionary/scan

Manuscript received on 02.11.2018

Appendix

Source code of the program

```
// Scanning of q-component mixture spaces, using adaptable independent steps
// The lengths of the steps over the components can be different
// The only requirement is the length of the first and second components to be equal
// datain.sks - the imput data file, having the following structure
// "number of components"
// "lower bound of component 1" "upper bound of component 1" "step for component 1"
// "lower bound of component 2" "upper bound of component 2" "step for component 2"
// "lower bound of component 3" "upper bound of component 3" "step for component 3"
// etc.
// "lower bound of component q" "upper bound of component q" "step for component q"
// dataout.sks - the output data file, having coordinates of the generated points
// Variables in use
// c12 - number of the steps over the first and second component
// c3q - number of the steps over the component being scanned
// eps - machine zero
// i, j, is, del, qsr, sm, sum1 - temporary variables
// lb [] - array with the lower bounds
// ub[] - array with the upper bounds
// minst - minimal acceptable number of steps
// np - number of the generated points
// q - number of the components
// st1 [] - array with current length of the consistent steps over components Nos 3 to q
// st12 - initial length of the steps over the first and second component
// st [] - array with the initial length over the components Nos 3 to q
// str - current length of the consistent steps over the fist and second component
// ubr[] - array with the consistent upper bounds
// x [] - array with the current point
// x1[][] - array with the generated points

#include <iostream>
#include <fstream>
#include <iomanip>
using namespace std;

int main(void)
{
        int q,np,c12,c3q;
        int i,j,is;
        double sm,st12,qst,sum1;
        double del,str;
        double x[20],lb[20],ub[20],st[20],st1[20],ubr[20];
        double x1[10000][20];

                double eps=1.0e-7;
        const int minst=3;
        ifstream fin;
        ofstream fout;
        fin.open("datain.sks");

        fin >> q;
        i=0;
        while (fin.eof()||i<=q-1)
        {
                fin >> lb[i] >> ub[i] >> st[i];
            i=i+1;
        }
        fin.close();
        fout.open("dataout.sks");
        if (st[0]!=st[1])
        {
        cout<<"The steps of X1 and X2 are not EQUAL"<<endl<<endl;
        cout<<"Please make them equal or rearange the factors accordingly"<<endl<<endl;
        goto out;
        }
        else
                st12=st[0];
        np=-1;
        is=q;
// intializaton of the scanning for factors Nos 3 to q
one:for (j=2;j<=is-1;j++)
                x[j]=lb[j];
        i=2;
        while (i<is)
        {
                sum1=0;
                for (j=i;j<=q-1;j++)
                        sum1=sum1+x[j];
// calculates the consistent upper bound
                ubr[i]=min(ub[i],(1-sum1));
// calculates the number of the steps to be performed for the ith factor
                c3q=(ubr[i]-lb[i])/st[i]+1;
// recalculate the step to 1/(minst-1) of the consistent range
```

```
        if (c3q<minst)
            st1[i]=(ubr[i]-lb[i])/(minst-1);
        else
            st1[i]=st[i];
        i++;
    }
    is=2;
// intialization of the scanning for the first and second factors
    x[0]=lb[0];
    sm=0;
    for(j=2;j<=q-1;j++)
        sm=sm+x[j];
    x[1]=1-x[0]-sm;
    if (x[1]<lb[1]-eps) goto three;
    if (x[1]>ub[1]+eps)
        {
            del=x[1]-ub[1];
            x[1]=ub[1];
            x[0]=x[0]+del;
            if (x[0]>ub[0]+eps) goto three;
        }
// calculates the number of the steps over the first and second factors
    c12=(min(ub[0]-x[0],x[1]-lb[1])/st12)+1;
// decreases the step length to 1/(minst-1)
    if(c12<minst)
        str=min((ub[0]-x[0]),(x[1]-lb[1]))/(minst-1);
    else
        str=st12;
two:np=np+1;
// stores the generated point
    for(i=0;i<=q-1;i++)
        x1[np][i]=x[i];
     for (j=0;j<=q-1;j++)
        {
            fout << setiosflags(ios::left) << setw(8) <<x1[np][j];
            cout << setiosflags(ios::left) << setw(8) <<x1[np][j];
      }
    fout << endl;
    cout << endl;
// checks whether the bounds of the first and second factor are kept
    if(x[0]>ub[0]-eps||x[1]<lb[1]+eps) goto three;
// steps over the first and second factor
    if (x[0]+str>ub[0]||x[1]-str<lb[1])
        {
            qst=min((x[1]-lb[1]),(ub[0]-x[0]));
            x[1]=x[1]-qst;
            x[0]=x[0]+qst;
        }
    else
        {
            x[1]=x[1]-str;
            x[0]=x[0]+str;
        }
    goto two;
// making steps over factors 3 to q
three:  qst=x[is]+st1[is];
// checks whether the step will break the upper bound
    if (x[is] >=ubr[is])
         {
            is++;
            if(is>q-1) goto out;
            goto three;
        }
    else if (qst<(ubr[is]-eps))
        {
            x[is]=qst;
            goto one;
        }
    else
//if the step will bring us too close to the upper bound
// the one accepts the upper bound, instead
        {
            x[is]=ubr[is];
            goto one;
        }
out:
fout.close();
cout<<"Press ENTER to STOP....."<<endl<<endl;
cin.ignore(1);
```

Kalin Stoyanov, *received his Ph.D. in 1996, is currently working as a Sr. Assistant Professor in the Department of Automation at University of Chemical Technology and Metallurgy. He received his M.Sc. degree in Technology of silicates and Ph.D. degree in Automation of Industrial Processes, both from the University of Chemical Technology and Metallurgy, Sofia, Bulgaria. His main research interests include applied statistics; neural networks; artificial intelligence; design of experiment.*

Contacts:
Department of Automation
University of Chemical Technology and Metallurgy
8, St. Kliment Ohridski Blvd, Sofia, Bulgaria
e-mail: kalin.stoyanov@uctm.edu