Journal of Science Education and Technology manuscript No.
(will be inserted by the editor)

Algorithm to generate multi-factorial experiments to teach experimental design

A.C. Delgado-Chavez · N. Balagurusamy · R. Narayanasamy · S. K. Gadi

the date of receipt and acceptance should be inserted later

Abstract One of the challenges in teaching the subject, Design of Experiments, is to come up with a proper numerical example. In this article, authors present a methodology to generate a numerical example for multifactorial experiments. Also, it presents a simple algorithm, which can be implemented in any programming language to generate unique examples.

Keywords Experimental design; educational tool; generating examples

1 Introduction

Experimental design is applied in almost all the fields involving experimentation [**?**,**?**,**?**,**?**]. It is part of various undergraduate and graduate curriculum, ranging from the engineering to the biological sciences. In general, the objective of experimental design is to minimize cost and time of the experiments and maximize the yield. As an example, in some cases with a lot of data to process, use of experimental designs help to find the optimal conditions for a process or in order to obtain the maximum yield of a product in a minimum number of experiments [**?**]. On the other hand, an improper design of experiment may lead to inaccurate or false conclusions, as well as a loss of money, material and time [**?**].

Learning statistics or mathematics in general is effective by solving a number of numerical examples [**?**]. It helps the students to develop insight in the topics [**?**]. It is well documented that students show good learning experience using visual examples and perform better with the examples of experiments which they can relate [**?**]. Teachers may involve students in finding experiments to teach the topic [**?**,**?**,**?**]. However, it is teacher's task to generate examples for the classroom and for the practice [**?**].

Solving optimization problems and finding the most accurate mathematical model for a process/system in experimental design involves performing various experiments with different combinations of the factors. Conducting experiments on a real system for the classroom purpose is not always feasible due to any of the following limitations.

1. The cost of conducting experiments on a real system is not always negligible.
2. A considerable amount of time may take for each experiment.
3. The combination of factors associated for optimum response is constant for a physical system. Therefore, teachers may not provide a fresh problem.

Hence, a computer program generating responses for the given input factors is a good alternative to mimic the physical systems. In this article a methodology is presented to generate numerical examples which simulate experiments. The objective is to generate unique process for the limits selected by the user, which outputs experimental data for the given combinations of the factors. Teachers may adopt this methodology in generating numerical examples, which highlight all the characteristics they want to present to the classroom, give as practice exercise and conduct exams.

The proposed algorithm is described in Section 4. Readers interested only in the implementation of algorithm may skip the mathematical construction presented in Section 2 and 3.

A numerical example for an experimental design is a mathematical model representing a physical process. This model is a set of static functions (i.e. it does not

Address(es) of author(s) should be given

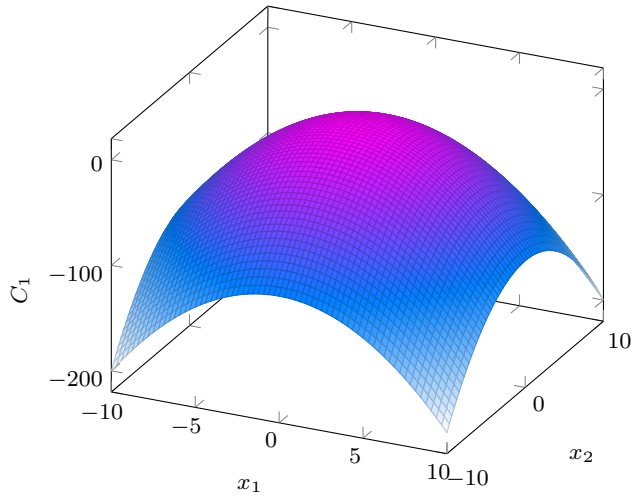

Fig. 1 Quadratic concave function $C_1(x_1, x_2)$.

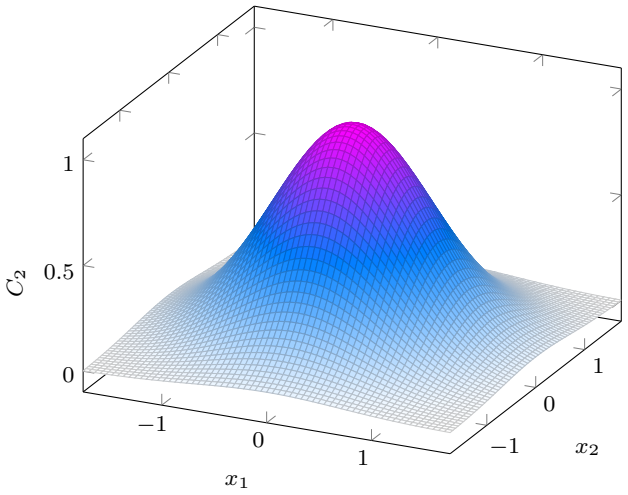

Fig. 2 Two variable Gaussian function $C_2(x_1, x_2)$.

have derivative or integral terms) which maps the factors to the responses. A real life system may present more than one peaks. However, most of the experimental design methods find the local maximum based on the initial base value. Hence, the proposed algorithm is designed to present only one peak. A multi-response system can be represented as

$$y_j = f_j(x_1, x_2, x_3, \ldots, x_n) + \xi_j \tag{1}$$

where y_j, $j \in \{1, 2, 3, \ldots, m\}$ are the responses, x_i, $i \in \{1, 2, 3, \ldots, n\}$ are the factors, f_j, $j \in \{1, 2, 3, \ldots, m\}$ are the nonlinear functions mapping the n factors to the m responses and ξ_i, $i \in \{1, 2, 3, \ldots, m\}$ are the noise.

All the factors, x_i, are constrained by upper and lower limits. The numerical examples should produce a unique optimal responses, y_j^M, for a set of factors within its limits. Construction of a one such mathematical function is presented in the next section.

The proposed algorithm presents the case of single response, which can be adopted to multi-response.

2 Construction of a mathematical function to suit our requirements

In this section possible candidates for the function f_i are discussed. The selected candidate function is then adapted to meet our further requirements in the next section.

2.1 Quadratic concave function

A second order polynomial function, such as

$$C_1(x_1, x_2, x_3, \ldots, x_n) := -\sum_{i=1}^{n} x_i^2 \tag{2}$$

is a concave function, which serve the purpose of providing a unique optimal point. Fig. 1 depicts (2) for the two variables case. However, it doesn't meet the requirements of a good example because to the following limitations.

1. Response surface methodology uses a second order fit algorithm. Hence, the process of reaching optimal solution becomes trivial.
2. A quadratic function is having a property that its slope increases as it moves far from the optimal point. This property trivializes the process of selecting a new base value.

2.2 Multivariable Gaussian function

The multivariable Gussian function

$$C_2(x_1, x_2, x_3, \ldots, x_n) := \prod_{i=1}^{n} \mathrm{e}^{-x_i^2} \tag{3}$$

is a concave function, hence, it has a unique maximum value. The slope of this function is not linearly related with the distance from its optimal point. The concave functions have property that the response of all the points between any two arbitrary points always grater than the responses at these arbitrary points [**?**]. A non-concave function gives additional challenge in solving the optimization problem.

2.3 Modified version of Gaussian function

Keeping above limitations in mind, a modified version of Gussian function is proposed as

$$C_3(x_1, x_2, x_3, ..., x_n) := \sum_{i=1}^{n} \mathrm{e}^{-x_i^2}. \tag{4}$$

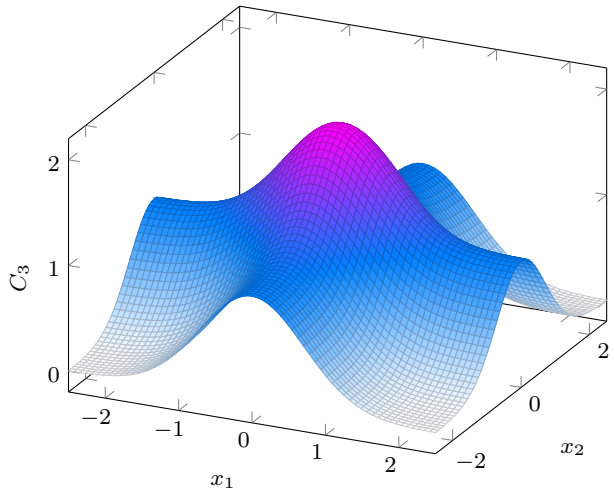

Fig. 3 Modified version of Gaussian function $C_3(x_1, x_2)$.

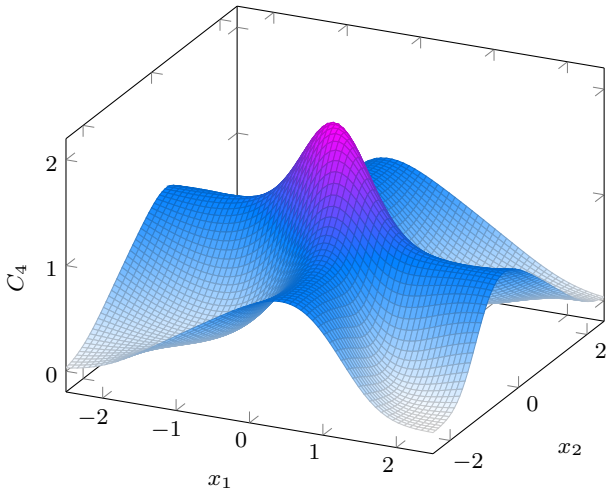

Fig. 4 The proposed function $C_4(x_1, x_2)$ for $a = 0.49$.

Fig. 3 depicts (4) for the case of two variables. A symmetric matrix is called negative definite when all its eigenvalues are negative. A function can be said concave, if Hessian matrix associated with it is negative definite [**?**]. Hessian matrix for (4) is

$$H_{i,j} = \frac{\partial^2 C_3}{\partial {x_i}^2} = \begin{cases} 2\mathrm{e}^{-x_i^2}(2x_i^2 - 1), & \text{if } i = j \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

where $i, j \in \{1, 2, 3, \ldots, n\}$.

The above equation shows that the Hessian matrix, H, is a diagonal matrix. In a diagonal matrix each element on the principal diagonal is an eigenvalue. So, it can be said that this matrix is not a negative definite because there exist positive elements for $|x_i| > \sqrt{\frac{1}{2}}$.

In a function gradient is zero at the peaks, dips and saddle points. The gradient vector of (4) is

$$\Delta C_{3_i} = \frac{\partial C_3}{\partial x_i} = -2x_i\mathrm{e}^{-x_i^2} \tag{6}$$

where $i \in \{1, 2, 3, \ldots, n\}$. $\Delta C_{3_i} = 0$ implies $x_i = 0$ or $x_i = \pm\infty$. Hence, it is guaranteed that there exists only one peak at $x_1 = x_2 = x_3 = \cdots = x_n = 0$.

In this function, the optimum value for any i^{th} factor is unaffected with the other factors. This is not recommended because it trivializes the multi-factorial problems.

2.4 A novel mathematical function

The following mathematical function is proposed by introducing a new nonlinear term to the above model.

$$C_4(.) := \sum_{i=1}^{n} \exp[-\left(x_i + a\sin\left(\sum_{i=1}^{n} x_i\right)\right)^2] \tag{7}$$

Fig. 4 depicts C_4 for a two variable case with $a = 0.49$. The gradient of C_4 is

$$\begin{aligned} \Delta C_{4_i} &= \frac{\partial C_4}{\partial x_i} \\ &= -2\exp[-\left(x_i + a\sin\left(\sum_{i=1}^{n} x_i\right)\right)^2] \times \\ &\quad \left[x_i + a\sin\left(\sum_{i=1}^{n} x_i\right)\right]\left[1 + a\cos\left(\sum_{i=1}^{n} x_i\right)\right] \end{aligned} \tag{8}$$

where $i \in \{1, 2, 3, \ldots, n\}$. Peaks, dips or saddle points form at

$$\Delta C_{4_i} = 0 \tag{9}$$

Only one peak is required, other peaks, dips and saddle points should be suppressed. Equation (9) can be solved by solving the following three equations.

$$\exp[-\left(x_i + a\sin\left(\sum_{i=1}^{n} x_i\right)\right)^2] = 0 \tag{10}$$

$$x_i + a\sin\left(\sum_{i=1}^{n} x_i\right) = 0 \tag{11}$$

$$1 + a\cos\left(\sum_{i=1}^{n} x_i\right) = 0 \tag{12}$$

The solution for (10) is $|x_i| = \infty$, which can be ignored. The solution for (12) is

$$\sum_{i=1}^{n} x_i = \cos[-1](\frac{-1}{a}) \tag{13}$$

Selecting a value $|a| < 1$, all the real solutions of (13) can be suppressed. Equation (11) can be rewritten as

$$x_i = -a\sin\left(\sum_{i=1}^{n} x_i\right) \tag{14}$$

which implies that the solution lies at $x_1 = x_2 = x_3 = \cdots = x_n$. Hence (14) can be rewritten it as

$$nx_i + na\sin(nx_i) = 0 \tag{15}$$

Since $|\sin(p)| < p$ for all the values of p except for $p = 0$, the solution can be limited to only one point $x_1 = x_2 = x_3 = \cdots = x_n = 0$ provided that $|na| < 1$. Considering a positive value for a, only one peak at origin for $a < \frac{1}{n}$ is guaranteed.

In the next section a method is presented to adapt the function C_4 defined at (7) to generate random experiments.

3 Adapting the proposed function

Noise is added. A scaled version of the function C_4 proposed in (7) is

$$\begin{aligned} f\ \ &(x_1, x_2, x_3, \ldots, x_n) = \\ F\ \ &(z_1, z_2, z_3, \ldots, z_n) := F_1 + \\ &\frac{F_2}{n}\left[\sum_{i=1}^{n} \exp[-\left[z_i + a\sin\left(\Sigma_z\right)\right]^2] + K_R\xi\right], \end{aligned} \tag{16}$$

$$F_1 := F_L + \alpha F_I \Xi, \tag{17}$$

$$F_2 := (1 - 2\alpha)F_I\Xi, \tag{18}$$

$$F_I := (F_U - F_L), \tag{19}$$

$$\Xi := 0.5(\xi + 1), \tag{20}$$

$$\Sigma_z := \sum_{i=1}^{n} z_i, \tag{21}$$

$$z_i := \frac{K_D(x_i - x_i^M)}{x_i^I}, \tag{22}$$

$$x_i^I := x_i^U - x_i^L, \tag{23}$$

$$x_i^M := x_i^L + \beta x_i^I + (1 - 2\beta)x_i^I\Xi, \tag{24}$$

$$0 < \alpha < 0.5, \tag{25}$$

$$0 < \beta < 0.5, \tag{26}$$

where $i \in \{1, 2, 3, \ldots, n\}$, ξ is a random variable with the properties $E(\xi) = 0$ and $|\xi| \leq 1$, Ξ is a random variable with the properties $E(\Xi) = 0.5$ and $0 \leq \xi \leq 1$, $E(.)$ is the expected value. $[F_L, F_U]$ is the function range, K_R is a noise factor, K_D is difficulty factor, x_i^M are the optimal combination of factors where the function g reaches its maximum value, x_i^L and x_i^U are lower and upper limit of the i^{th} factor, α and β are padding constants for limiting the maximum value of the function f and limiting the optimal combination within the desired region respectively.

The function $F(z_1, z_2, z_3, \ldots, z_n)$ of (16) preserves all the following mathematical properties of the function C_4 proposed in (7).

1. Gradient is not proportional to the distance from its optimum combination.
2. Has unique maximum value at $z_1 = z_2 = z_3 = \cdots = z_n = 0$ i.e. at $x_i = x_i^M, \forall i \in \{1, 2, 3, \ldots, n\}$, provided that $a < \frac{1}{n}$
3. It is not a concave function.
4. The optimal value of an arbitrary factor is not constant throughout the factorial space.

In the next section, an algorithm is presented to show implementation procedure.

4 Algorithm

Fig. 5 shows the flowchart of the proposed algorithm. The values of α and β should be less than 0.5. It is recommended to use $\alpha = \beta = 0.2$. F_L and F_U are lower and upper limits of the values generated in the experiments. Hence, user should select the values such that $F_L < F_U$. K_D is the difficulty factor, the bigger the value is assigned, the harder it is to reach the optimum value. It is recommended to use a value $K_D = 5$. The noise factor K_R introduces noise into the system.

Using the equations and inequalities given from (16) to (26), the algorithm generates responses for the given x_i inputs.

5 Application

The proposed algorithm shown in Fig. 5 is implemented in the programing language JavaScript, which is complemented with a graphical user interface (GUI) designed in HTML and CSS. This application is used in the classroom for teaching Response Surface Methodology (RSM) to the master students of Biological Sciences Faculty, Universidad Autónoma de Coahuila, Torreón.

This application is executed with $n = 2$, $\alpha = \beta = 0.2$, $F_L = 100$, $F_U = 600$, $K_D = 5$, $K_R = 0.1$, $x_1^L = 0$, $x_1^U = 100$, $x_2^L = -1$, and $x_2^U = 1$. The application calculated $a = 0.475$, $F_1 = 193.043$, $F_2 = 275.866$, $x_1^I = 100$, $x_2^I = 2$, $x_1^M = 25.291$, and $x_2^M = 0.356$. Fig. 6 shows the contour plot of the function $f(x_1, x_2)$ generated with these constants. The RSM is applied to find the optimal values of x_1 and x_2, results of each iteration are also superimposed over the contours.

6 Conclusion

The Construction of a single response, unique peak multivariable mathematical function for is presented. Later it is adapted to generate experimental data for a selected range of factors. An algorithm is proposed,

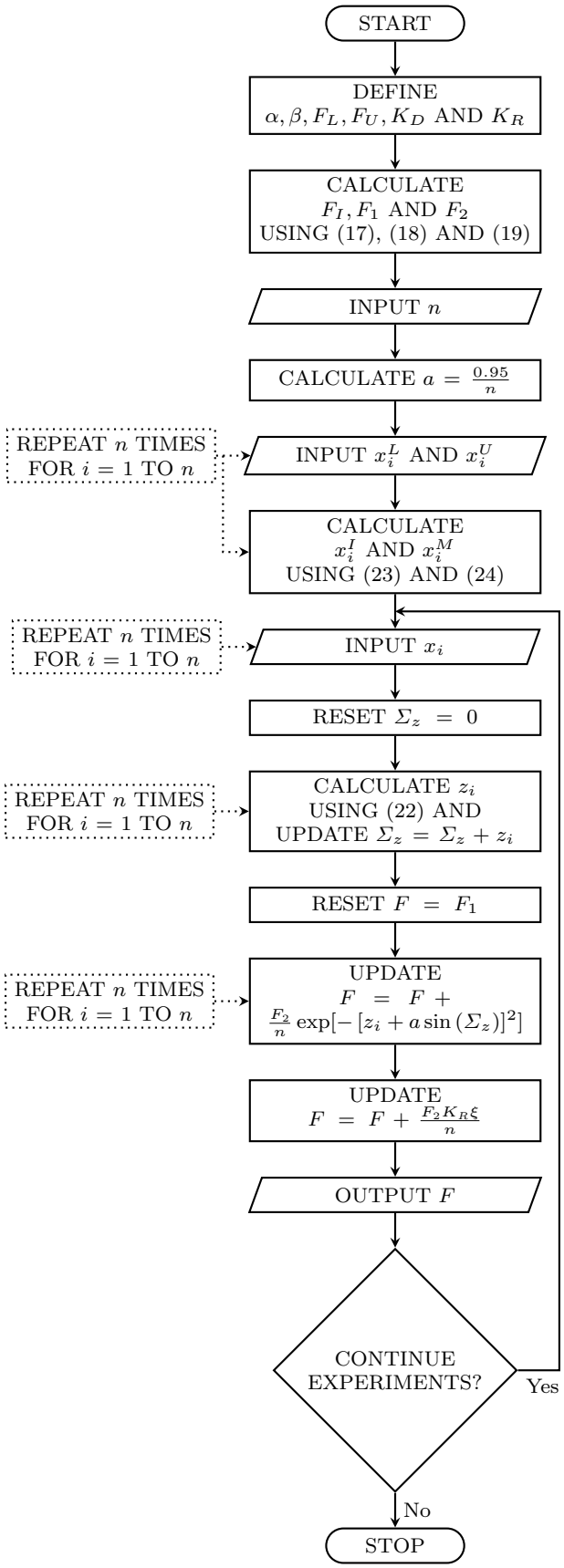

Fig. 5 Flowchart of the proposed algorithm

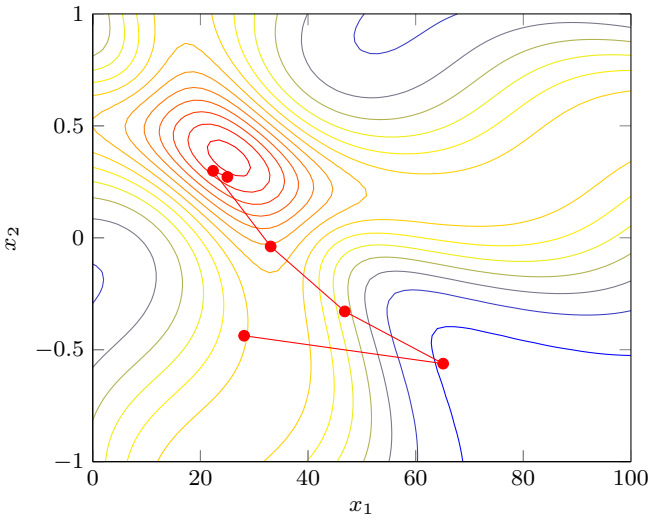

Fig. 6 Contour plot of F with the constants given in Section 5 superimposed with the RSM results.

which can be realized in any programming language. Based on this algorithm an application is designed in HTML, CSS and JavaScript. It is used in the classroom to teach the topic of Response Surface Methodology (RSM).

It is developed for maximum values, but can be adapted to the minimum by putting negative to the function and scaling accordingly. That is a unique dip (opposite to a peak) can be obtained by selecting a negative values for F_2. It requires to modify (17) and (18).

This work can further be extended to a multiple response case by generating m number of functions F_j where $j \in 1, 2, 3, \ldots, m$, which requires to generate $m \times n$ number of values for x^M, i.e. the values x_i^M are replaced by $x_{i,j}^M$ where $j \in 1, 2, 3, \ldots, m$ and $i \in 1, 2, 3, \ldots, n$.