Analusis, **1998**, *26*, 304-310

Neural networks applied to the choice of an optimal experimental design

S. Courtois* and R. Phan-Tan-Luu

L.M.R.E., Faculté des Sciences, Centre de St Jérôme, 13397 Marseille Cedex 20, France

Abstract. In the Methodology of Experimental Research (M.E.R.), the quality of the results depends on the choice of the experimental design. A lot of experimental designs exist. The goal of the presentation is to optimize the choice of an experimental design by neural networks. We present in detail the elaboration of networks which facilitates the choice of an experimental design as a second-degree model which studies six factors in the spherical experimental domain.

Key words. Methodology of Experimental Research (M.E.R.) – experimental design – neural networks with non-chaotic dynamics – learning: gradient back propagation.

Introduction

The object of the Methodology of Experimental Research (M. E. R.) [1–3] is to search an optimal strategy which will allow to obtain the largest number of good quality information (i.e. precise and unbiased) concerning a studied phenomenon, while carrying a limited number of experiments.

For each problem formulated, the first question which must be studied is the choice of the *factors*. We must remind that the factors are the parameters that we can control. We must choose the variation limit of these factors, which determines the *experimental domain*. These variations may have very different orders of magnitude, so that, to be able to compare the factor effects, it is necessary to work with the coded levels of variation for each factor. Each experiment represents a particular point of the experimental domain, and provides a measurement with one or several *responses* of the phenomenon in this point. We are led to build a matrix each line of which will correspond to an experiment, and each column of which will represent a factor. It is the *experimental design*. If we try to forecast the studied phenomenon in all the experimental domain, we then choose a *mathematic*

* Correspondence and reprints.
Received February 27, 1998; revised July 28, 1998; accepted August 04, 1998.

model which is supposed to represent properly this phenomenon in all the points of this domain.

If B represents the vector of the estimations of the model coefficients, including the experimental error,

X the matrix of the model, obtained from the experimental design and from the postulated model,

Y the response vector,

σ_2 the variance of the error made during the experimentation,

we have:

$$Y = X \,.\, B$$

$$B = (X'X)^{-1} \,.\, X' \,.\, Y$$

$$\mathrm{Var}(B) = (X'X)^{-1} \,.\, \sigma_2$$

σ_2 does not depend on the model, the environment imposes it. In order to reduce the variance of the model coefficients, and since we cannot influence the value of σ_2, it is necessary to act on the matrix $(X'X)^{-1}$. So X is very important for the experimental design.

The experimental results, Y, are only used for the calculation of the estimations of the model coefficients. We can see that they don't play a role in the search for information presence or information quality, i.e. in the strategy. It is the choice of the matrix X which will influence the quality of the strategy. But a great number of experimental designs can be chosen [3], and some of them are only known by specialists, so it is important to create a software to help for decision. This software will automate the choice of the strategy through the matrix of experiments.

Why using neural networks rather than a deterministic software?

If it is not ambiguous, a deterministic solution is more appropriate than a connectionist solution. But, if we want to introduce a flexibility possibility a neural network is necessary.

For example, if, for a quantitative problem of six factors, a user wants to find a design that has four different levels for the fifth factor, a connectionist software presents some designs that have four levels for the fifth factor. But it also can propose a design which economizes on the experiments' number and is as efficient, but which has only three levels for the fifth factor.

Then it is possible for the user to maintain the first choice, (four levels for the fifth factor), or to prefer the more performant design i.e. less expensive and quicker (three levels for the fifth factor).

The neural network presents the advantage of being able to select a design, close to the wanted choice, more adapted in certain aspects. On the contrary, a determinist software can only present matrices that satisfy the first choices.

We have chosen for this study, the multilayer networks with learning by error gradient backpropagation (G.B.P.) [4,5].

The structure of a multilayer network is given by figure 1. The networks is made by neurons organised in layers. A neuron receives inputs $t_j = \sum_{i=1}^{n} w_{ij} \times E_i$. After sommation, these input values t_j are modified by the transfer function which is often a sigmoid function (see Tab. VII.1). It produces an output value T_j of the neuron, $-a < T_j < +a$. Between the input and output layers is located at least one hidden layer. In our example, the network is completely connected by layers: each neuron of a layer is connected with all the neurons of the following layer. When we impose to the network an input vector, i.e. when we give a value to each neuron of the input layer, we can calculate the output vector by the equations, table VII.1. If the weights and the biaises are correctly chosen, the output vector will give the expected response. The choice of the weights and the biaises is made by a long learning by alterning propagation and retropropagation. Afterward we impose again to the network the input vector (see equations Tab. VII.1.2.3.). The learning allows the network to adapt gradually its matrices of weight and bias and so to memorize the informations contained in the knowledge base. This base is described by some input vectors named patterns. All the patterns constitute the learning set.

Choice of the network structure

Choice of the input values number of the network

The input values of a network are representative of the characteristics of the problem. Their number is the minimum number of necessary characteristics to describe the patterns of the learning set, which are the examples to memorize. A neural network distinguishes the importance of an input, in comparison with another, only by its value. Thus, to teach the network that two pattern groups are separate, it is necessary to differentiate one of the inputs by attributing to it a sizable weight, or by using two different networks, each of which having an input number lower than the big network, and a smaller learning set.

For instance, if, for a study, it is necessary to study six factors, the chosen design will be with six factors. If, in a same big network, all the expected designs for the studies of two to six factors are processed, it is an input neuron which will take care of the "factors number" information. It is to be feared that, in the case of a six factors design, where

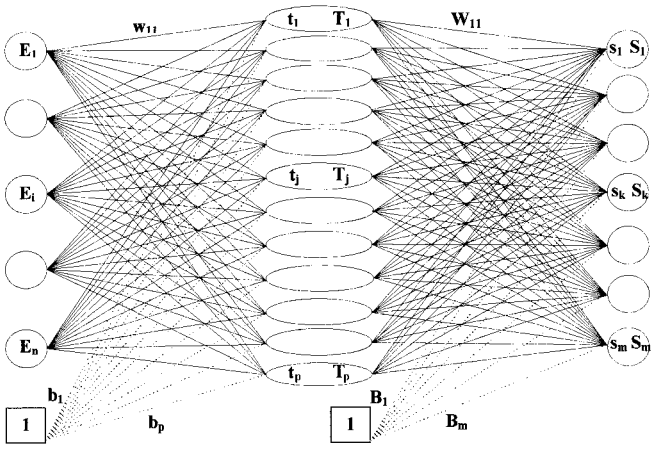

Fig. 1. Neural network 5-12-7.

Table I. Properties of designs with six factors.

Type of design	*1* *number of experiments*	*2* *sequentially exp. domain*	*3* *add variable*	*4* *number of levels per factor*
1. Doelhert	43	yes	yes	1 factor with 3 levels 1 factor with 5 levels 4 factors with 7 levels
2. Doelhert max	43	yes	no	6 factors with 7 levels
3. Doelhert min	43	yes	no*	1 factor with 3 levels 4 factors with 5 levels 1 factor with 7 levels
4. composite $\alpha \neq 1$	45	no**	no*	6 factors with 5 levels
5. Box-Behnken	49	no**	no	6 factors with 3 levels
6. hybrid 628 A	28	no	no	5 factors with 5 levels 1 factor with 4 levels
7. simplex-sum	57	no	no	1 factor with 3 levels 1 factor with 5 levels 1 factor with 7 levels 3 factors with 9 levels

(*) It is not impossible to add a variable in the case of a Doelhert design with level minimization. It is the same, although it is not as easy, in the case of a composite design.
(**) It is possible to extend the domain, providing certain precautions are taken, in the case of a composite matrix $\alpha \neq 1$, and in the case of a Box-Behnken design.

all the inputs are not exactly satisfied, the network presents a five factors design, better responding to the other input constraints. Because a network cannot distinguish the inputs with imperative choice from the inputs with hoped choice. Of course, it is always possible to increase significantly the inputs value with imperative choice. But, if there is too much disparity between the different input values, the corresponding matrices are "not properly configured", and they bring about an instability in the calculations: the input of a pattern a little different from the preceding leads to a weight matrix with a totally different weight. The learning is longer and then converges with difficulty because the dynamics of the system is less stable.

After having carried out a design of experiments to study the influence of the network structure on the response quality [6,7], it seems that it is better to use several small networks rather than only one big network. When the responses were some separate set, we used some separated networks. That is why we chose, for each factors number, a separate network. The learning sets are smaller, the learning times are shorter, and the training dynamics are more stable.

Choice of the number of patterns of the learning set

If a network can modulate its response between the various possibilities that it has learnt, it responds wrongly in extension, i.e. in extrapolation. It does suitable choices only concerning the information contained in the set of patterns that it has memorized. If an information is not contained, explicitly or implicitly, in the patterns which the network has learnt, it will not know it. And the choices that it will make concerning that, will be wrong or incomplete. So, the network cannot answer correctly a question that it has not learnt. This is obvious, and is a matter of mere good sense, but it is not always as easy to verify that the learning set covers effectively all the experimental domain and that it represents effectively all the information to memorize, contained explicitly or implicitly.

Table II. Learning set for six factor designs.

n	*inputs*					*outputs*						
	1	2	3	4	5	1	2	3	4	5	6	7
1	1	0	0	0	0	0	0	0	0	0	1	0
2	0.75	0	0	0	0	1	1	1	0	0	1	0
3	0.5	0	0	0	0	1	1	1	1	0	1	0
4	0.25	0	0	0	0	1	1	1	1	1	1	0
5	0	0	0	0	0	1	1	1	1	1	1	1
6	0	1	0	0	0	1	1	1	0	0	0	0
7	0	0.5	0	0	0	1	1	1	1	1	0	0
8	0	0	1	0	0	1	0	0	0	0	0	0
9	0	0	0.50	0	0	1	0	1	0	0	0	0
10	0	0	0.2	0	0	1	0	1	1	0	0	0
11	0	0	0	1	0	0	1	0	0	0	0	1
12	0	0	0	0.5	0	1	1	0	0	0	0	1
13	0	0	0	0	1	0	0	1	1	1	1	0
14	0	0	0	0	0.5	1	0	1	1	1	1	0

In our study, the problem is to extract all the characteristics of some of the different experimental designs used in the laboratory (L.M.R.E.), and to arrange things so that the questions asked for, i.e. the network inputs, reflect really all the possibilities of these designs, and cover all the possible choices of the user. Tables I and II present the selected solutions, for the example that we have chosen: experimental designs with six factors, for a polynomial second-degree model in the spherical domain.

Choice of the number of network outputs

The number of networks outputs is determined by the number of designs that may be chosen.

Choice of the network structure parameters

Knowing the inputs number, the outputs number, and the learning set, a design of experiments has been used to optimize the network topology. The parameters have been chosen in order to avoid the chaotic learning dynamics [6,8,9]. This design has given us the number of hidden layers and the number of neurons per layer.

Application: Choice of experimental design with 6 factors, for a polynomial second-degree model in the spherical domain

Choice of the outputs: Six factors design

We can choose among eleven different designs, but seven of them distinguish by their qualities for the classic criteria: criterion-A, criterion-M, inflation factors, criterion-G, Khuri index [3].

We will allocate each output neuron one of these seven designs:

1. a standard Doelhert design,
2. a Doelhert design with levels number maximization,
3. a Doelhert design with levels number minimization,
4. a composite design with $\alpha \neq 1$,
5. a Box-Behnken design,
6. a hybrid design 628 A [10],
7. a simplex-sum design [11].

Choice of the inputs: Designs characteristics

We are going to study for these designs the following characteristics:

1. Number of experiments.
2. Possibility to displace or to extend the experimental domain, i.e. the sequentially in the space.
3. Possibility to add a variable.
4. Number of levels of each factor.

Table I recapitulates the properties of these designs.

Learning set

The numbers of learning set patterns are inscribed in the left column.

The seven outputs represent the seven designs:

1. Doelhert design,
2. Doelhert design with levels number maximization,
3. Doelhert design with levels number minimization,
4. composite design with $\alpha \neq 1$,
5. Box-Behnken design,
6. hybrid design 628 A,
7. simplex-sum design.

The learning set is done in the form of two parts, inputs and outputs (Tab. II).

The five inputs represent:

Input 1:

This input answers the question: *"Are your experiments expensive or hard to achieve?"* The patterns 1 to 5, corresponding to input 1, integrate the "save money wish":

- If the experiments are expensive, input 1 will take the value 1, and only the design number 6, the hybrid design 628 A, having only 28 experiments will be chosen.
- If the experiments are not expensive or are easily and quickly carried out, the input 1 will take the value 0 and all the possible designs will be selected.
- Between these two values, all the other possible designs have been selected according to their experiments number. In this way, for a value 0.25 of the input 1, only the simplex-sum design which requires 57 experiments has been excluded.

Input 2:

This input gives information on the sequentiality in the experimental domain.

- The patterns 6, 7 and 5 give information concerning the possibility of moving the experimental domain in the process of the study, without losing the benefit of the work already carried out. If it is to be hoped that the domain can be extended, value 1 for input 2, the three Doelhert designs allow easily this extension.
- In case it is preferable that all the factors have 3 or 5 levels and it is accepted to carry out some more experiments, it is possible, but less easy, to extend the domain with the composite designs or the Box-Behnken designs. This possibility will be numbered by input 2 = 0.5 (pattern number 7).
- The pattern number 5 is also representative of this input, if one does not expect to displace the experimental domain. All the inputs have the value equal to zero and all the outputs have a value equal to one, since all the designs are selected.

Input 3:

This input measures the possibility to add one variable during the work, without losing the benefit of the work already carried out.

- Only the Doelhert design allows the addition of a variable without great difficulty (pattern number 8).
- If it is to be hoped to add a variable while minimizing the levels, the Doelhert design with minimization of the levels also allows to add one variable, although it is less easy. This is numbered by an input 3 = 0.5 (pattern number 9).
- If it is to be hoped that all the six factors have five levels, and it is possible to carry out two additional experiments, although it is still less easy, it will be possible to choose a composite design $\alpha \neq 1$. This is numbered by input 4 = 0.2 (pattern number 10).

- The pattern number 5, input 4 = 0, represents the case where it is not expected to add a variable. In this case, all the designs are selected.

Input 4:

The input gives the possibility of maximizing the total number of factors levels.

- If the input 4 = 1 (pattern number 11), the Doelhert design with levels number maximization and the simplex-sum design will be selected. Both of them have 42 levels for the factors.
- If the input 4 = 0.5 (pattern number 12), the Doelhert design will be added. It has 36 levels for the six factors.
- The pattern number 5 is also representative for this input. If it is not to be hoped to maximize the number of factors levels, input 4 = 0, all the designs can be chosen.

Input 5:

This input gives the possibility of minimizing the total number of factors levels.

- If the input 5 = 1 (pattern number 13), the Doelhert design with level number minimization, the composite design $\alpha \neq 1$, the Box-Behnken design and the hybrid design 628 A will be selected.

It would have been possible to separate the Box-Behnken design, where the six factors have three levels (18 levels), from the Doelhert design with level number minimization, hybrid design 628 A, and composite design.

The Doelhert design has one factor with three levels, four factors with five levels and one factor with seven levels, corresponding to 30 levels, the hybrid design 628 A has 29 levels, (one factor with four levels, five factors with five levels), the composite design has thirty levels (six factors with five levels).

To separate these two designs groups, it is enough to add a pattern number 15 and to operate the following changes:

The pattern 13:	0,0,0,0,1	0,0,0,0,1,0,0
one pattern 15:	0,0,0,0,0.66	0,0,1,1,1,1,0
the pattern 14:	0,0,0,0,0.33	1,0,1,1,1,1,0.

- The pattern number 5 is also representative of this input. If it is not to be hoped to minimize the number of factors levels, input 5 = 0, all the designs can be chosen.

It would have been possible, and more logic, in the matter of Design of Experiments (D.O.E.) to gather the inputs 4 and 5 in order to have one input. This input would have, for instance, the values 0 to 0.4 in order to minimize the number of factors, 0.45 to 0.55 in order not to modify the numbers of factors, 0.6 to 1 in order to maximize the number of factors. We have chosen to part this choice in two possibilities for two reasons. The optimization of the network shape by an experimental design [6], is easier if each experimental design is described by an input vector; moreover this choice will be sharper if it is possible to quantize it from 0 to 1 for the maximization (or the minimization). Of course it is understood that the software do not allow to simultaneously answer input 4 > 0.5 and input 5 > 0.5 because these choices would be incompatible.

Building the network

The network will have five inputs and seven outputs, its learning set comprises fourteen patterns (cf. Tab. II).

We know the number of inputs and outputs of the network, as well as the number of the learning set patterns. The choice of the network parameters has been set after having carried out an experimental design [6]. It is a network 5-12-7, (cf. Fig. 1) completely connected by layers. The number of neurons of the hidden layer is chosen so as to make the network robust and to avoid that its learning dynamics be chaotic. So it must be able to accept the addition of some patterns without changing its behaviour, i.e. remaining robust and reliable after calculating the new bias and weight matrices.

All the tests carried out for this network 5-12-7 have been satisfactory: the mean square deviation between the various tests is 3×10^{-3} and the recognition average error of the learning set is 0.5%.

Bias and weight matrices achieved:

Bias matrix b

b is the first bias matrix. It corresponds to the ponderations of the connections between the input layer bias and the hidden layer. *b* is a column matrix with twelve lines, each element *b*(*j*) of which corresponding to the weight of the connection between the input layer bias and the neuron *j* of the hidden layer.

Table III represents the bias matrix *b*.

Weight matrix w

w is the matrix of the weight of the connections between the input layer and the hidden layer.

w(*i*,*j*) represents the weight affecting the connection between the input layer neuron *i* and the hidden layer neuron *j*, *i* varying between 1 to 5, and *j* varying between 1 to 12.

Table IV represent the weight matrix *w*.

Bias matrix B

B is the second bias matrix. It corresponds to the ponderations of the connections between the hidden layer bias to the output layer neurons.

Table III. Bias matrix *b*, 6 factors, network 5-12-7.

b(1)	=	3.653
b(2)	=	0.222
b(3)	=	0.604
b(4)	=	2.690
b(5)	=	–2.454
b(6)	=	–4.405
b(7)	=	–1.427
b(8)	=	–8.453
b(9)	=	–2.404
b(10)	=	–0.329
b(11)	=	–1.647
b(12)	=	–1.924

Table IV. Weight matrix *w*, 6 factors, network 5-12-7.

i = *j*=	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*
1	–4.372	–8.806	0.632	–6.934	–1.523	5.722	–2.788	9.811	–0.032	–1.627	5.352	–1.019
2	2.697	–1.782	3.812	–4.183	7.345	–0.704	2.488	0.989	–0.307	–1.118	1.100	–0.679
3	–4.559	2.110	–0.992	–14.665	–2.117	5.027	13.875	–0.011	–1.041	–2.529	–0.210	–1.387
4	–5.332	9.129	–2.694	–7.930	5.267	–0.076	–2.406	–0.798	–0.120	–3.342	2.043	4.006
5	–4.115	–3.542	–1.718	4.394	–2.336	–0.028	5.527	9.869	–0.593	5.170	0.262	–0.372

B is a column matrix with seven lines, each element *B*(*k*) of which corresponding to the weight of the connection between the hidden layer bias and the neuron *k* of the output layer.

Table V represents the bias matrix *B*.

Weight matrix W

W is the matrix of the weight of the connections between the hidden layer and the output layer.

W(*i*,*j*) represents the weight affecting the connection between the neuron *j* of the hidden layer and the neuron *k* of the output layer, *j* varying between 1 to 12, and *k* varying between 1 to 7.

Table VI represents the weight matrix *W*.

The network is ready to be used. It remains to include it in the software.

We give equations that permit to carry it out (Tab. VII, first part) as well as those that permit to generate the bias and weight matrices during the learning (Tab. VI, totally).

We can remark that a study with six factors is the kind of study with a great number of factors.

It could have been limited to the study of a smaller number of factors while keeping the possibility to add factors during the work by giving value 1 to the third input.

Table V. Bias matrix *B*, 6 factors, network 5-12-7.

B(1)	=	4.461
B(2)	=	1.661
B(3)	=	3.193
B(4)	=	0.583
B(5)	=	–5.831
B(6)	=	–0.367
B(7)	=	–5.984

Table VI. Weight matrix *W*, 6 factors, network 5-12-7.

i = *j*=	*1*	*2*	*3*	*4*	*5*	*6*	*7*
1	5.890	5.318	6.204	2.837	1.677	1.265	–0.700
2	–0.733	–3.516	–11.062	–5.303	–3.547	–6.637	10.483
3	3.609	2.708	3.210	–0.766	1.026	–1.441	–3.246
4	2.115	0.014	3.961	13.448	13.771	7.514	9.037
5	–5.306	4.936	–2.630	–6.342	2.388	–7.229	0.137
6	–1.951	–3.345	–5.989	–4.856	–4.345	3.549	–2.932
7	–0.060	–12.187	3.633	–1.736	–5.026	–5.253	–6.904
8	–10.683	–8.761	–6.048	–4.372	–1.237	3.167	–3.749
9	0.437	0.987	0.746	0.236	–0.565	0.717	0.579
10	–4.745	–1.826	2.602	3.522	2.955	3.228	–1.922
11	–2.353	0.801	–2.289	–3.504	–5.304	3.302	–3.270
12	–2.824	1.391	–1.998	–0.295	–0.198	0.132	2.672

Table VII. Equations governing a multilayer network.

The multilayer network: operation and learning equations by Error Gradient Backpropagation

1. propagation of a pattern and calculation of the output:

Calculation of $S_k = f\,(\Sigma_{j=1..p} W_{jk}\, f\,(\Sigma_{i=1..n} w_{ij} E_i))$ $k = 1..m$

with $f(x)=1/(1+e^{-x})$

$S_k = f\,(s_k)$

$s_k = \Sigma_{j=1..p} W_{jk} T_j + B_k$

$T_j = f\,(t_j)$

$t_j = \Sigma_{i=1..n} w_{ij} E_i + b_j$.

This propagation calculation is the calculation operated by the network in normal condition. Its bias and weights matrix was calculated by learning (cf. Tabs. 7.2 et 7.3 opposite column).

Names of variables are those of figure 1.

Backpropagation, modification of bias and weights during learning:

2. Calculation of weight variation of output layer:

Calculation of $dW_{jk} = -\alpha * \psi_k * T_j$ $k = 1..m$

$dB_k = -\alpha * \psi_k$

with $\psi_k = (S_k - \sigma_k) * f\,(s_k) * (1 - f\,(s_k))$,

3. Calculation of weights variation of hidden layer:

Calculation of $dw_{ij} = -\alpha * \varphi_j * E_i$ $j = 1..p$, $db_j = -\alpha * \varphi_j$

with $\varphi_j = f'\,(t_j) * \Sigma_{k=1..m} (\psi_k * W_{jk})$.

The learning set is presented to the network; this, little by little modifies the weight matrix. The algorithm stops when the sum square of errors, SEr^2, is less than a value chosen beforehand.

Conclusion

For the choice of the experimental designs, the multilayer networks appear to be an excellent alternative to the procedural algorithmic because they bring a flexibility that the conventional algorithms cannot produce and allow not to have to explain all the knowledge included in the learning set. Well conceived and well trained, the multilayer networks with learning by errors gradient backpropagation are reliable and robust tools that permit the automation of the experimental designs choice.

References

1. Fargin, E.; Sergent, M.; Mathieu, D.; Phan-Tan-Luu, R. Approche Méthodologique de la Recherche Expérimentale, *Bio Sci. 85* **1985**, *4*(4), 77-82.
2. Mathieu, D.; Phan-Tan-Luu, R. Approche méthodologique des surfaces de réponse, Plans d'expériences, applications à l'entreprise, Droesbeke, J. J.; Fine, J.; Saporta, G. Eds., Technip, Paris, 1997; pp 211-277.
3. Peissik, A. Propriétés des matrices d'expériences pour les modèles polynomiaux du second degré, Thèse de doctorat, Université Aix-Marseille III, 1995; pp 115-141, 27-53.
4. Common, P. Classification supervisée par réseaux multicouches *Traitement du signal* **1991,** *8*(6), 387-408.
5. Jodouin, J. F. Les réseaux neuromimétiques, modèles et applications, Hermès, I.S.B.N.2-86601-436-7, 1994
6. Courtois, S. Réseaux neuronaux appliqués à la recherche d'une stratégie optimale, Thèse de doctorat, Université d'Aix-Marseille III, 1996; pp 51-55, 103-121, 159-169.
7. Mathieu, D., Phan-Tan-Luu, R. Logiciel NEMROD, *LPRAI*, 1986.
8. Bergé, P. Le Chaos, théorie et expériences, collection C.E.A., Eyrolles, 1988.
9. Quoy, M. Apprentissage dans les réseaux neuromimétiques à dynamique chaotique, Thèse de doctorat, Université Paul Sabatier, Toulouse, 1994.
10. Roquemore, K. G. Hybrid designs for Quadratic Response Surfaces *Technometrics* **1976**, *18*, 419-424.
11. Box, G. E. P.; Behnken, D. W. Simplex-Sum Designs- A Class of Second Order Designs Derivable from those of First Order *Ann. Math. Stat*. **1960**, *31*, 838-864.