Optimal allocation of resource for surrogate modelling

V. Picheny, N-H Kim, R.T. Haftka

University of Florida
Gainesville, FL-32611, USA

E-mail: picheny@emse.fr

Abstract. We address the issue of fitting a surrogate to data generated from numerical simulators that have tunable fidelity. In particular, we focus on problems where data is generated using Monte-Carlo simulations. We show that this problem is similar to optimal design of experiments (DOE) but includes additional variables: the accuracy of the response at each point, and a computational time constraint. Since full optimization is often infeasible, we propose two alternatives to obtain nearly optimal designs: fixing a spatial DoE and optimizing only with respect to the fidelities, or constructing sequentially the design for both training point locations and fidelities. Finally, we apply the two techniques to the fitting of polynomial models, and show substantial improvement in accuracy using optimal allocation of resource.

1. Introduction

In most engineering fields, numerical simulators are used to model complex phenomena and obtain high-fidelity analysis. Despite the growth of computer capabilities, such simulators are limited by their computational cost, since it can take up to days for a single simulation. Surrogate modeling is a popular method to limit the computational expense. It consists of replacing the expensive model by a simpler model (surrogate) fitted to a few chosen simulations at a set of points called a design of experiments (DoE).

In many applications, the fidelity of numerical simulations depends on tunable factors that control the complexity of the model. For instance, the precision of the response of a finite element analysis (FEA) can be controlled by the meshing density or element order. Another example is when the response is obtained using Monte-Carlo methods: the accuracy (measured by the standard deviation or confidence intervals) is inversely proportional to the square root of the number of samples.

In the field of surrogate modeling, it is common to work with two or three models of different fidelity. In this work, we propose to expand this to tunable fidelity at each sampling point. Indeed, by allocating different computational time for each simulation, we can improve the quality of the surrogate without additional cost compared to a uniform fidelity situation. Such formulation belongs to the field of optimal DoE, which consists initially of selecting the points to maximize a criterion of quality for a metamodel construction [1] [2] [3]. Here we can, in addition, tune the model fidelities to improve further the criterion. We seek to optimally allocate computational resources to the points.

Elfving [4] pioneered work in the area of optimal allocation. For a linear model he proposed to repeat experiments at some training points, and allocate a different number of repetitions to each point according to an optimality criterion. Kiefer and Wolfowitz [1] generalized this idea of proportioning experiments in order to obtain the most accurate regression coefficients. Fedorov [5] proposed a continuous version of the same problem, and developed iterative strategies to find optimal designs.

The objective of this paper is to propose a methodology to achieve optimal designs for numerical simulators with tunable accuracies, in particular when the simulator is based on Monte-Carlo methods. A general formulation of the optimal allocation problem is developed, and practical solutions are given to solve numerically the problem. We also aim at finding similarities with the work of Fedorov for some particular configurations.

The paper is organized as follow: first, we present a typical reliability-based design optimization (RBDO) problem and show how to define an optimization problem for allocation of computational resource; since such problem is very challenging, we propose alternatives to obtain nearly-optimal solutions with reasonable expense. Finally, we apply our strategy to polynomial regression examples.

2. Allocation of resource and design optimality

2.1. Using metamodeling for reliability based design optimization

To introduce the reader to the problem of resource allocation, we present the example of a classical RBDO problem. Let *F* be a cost function to minimize (e.g., weight) and *G* a reliability measure (e.g., probability of failure). The typical RBDO problem associated with them is:

$$\begin{aligned} &\underset{\mathbf{x}}{\text{Min}} && F(\mathbf{x}) \\ &\text{s.t.} && G(\mathbf{x},\mu_{\boldsymbol{\Theta}}) \le G_{\text{target}} \end{aligned} \tag{1}$$

x are the design variable and $\mu_{\boldsymbol{\Theta}}$ are the distributions of some random parameters **Θ**. We assume here that *F* is independent of the random parameters. The function $G(\mathbf{x},\mu_{\boldsymbol{\Theta}})$ is deterministic, but it is, generally, not known analytically. Hence, we estimate it using Monte-Carlo simulations (MCS). For a given design **x**, we write:

$$\hat{G}(\mathbf{x},\boldsymbol{\theta}_{i=1\ldots k}) = \hat{G}\left(\mathbf{x};\quad \boldsymbol{\theta}_1,\boldsymbol{\theta}_2,\ldots,\boldsymbol{\theta}_k\right) \tag{2}$$

Where $\hat{G}$ is the estimator of *G* and $\boldsymbol{\theta}_1,\boldsymbol{\theta}_2,\ldots,\boldsymbol{\theta}_k$ are realizations of **Θ**.

$\hat{G}$ is a random response which variability depends on the sample size *k*. In order to use directly $\hat{G}$ to solve the optimization problem, *k* must be chosen very large so that the variability does not prevent the convergence of the optimizer. Such procedure may become very expensive, especially if the constraint is evaluated numerous times during the optimization. Thus, we replace the costly estimate $\hat{G}(\mathbf{x},\boldsymbol{\theta}_{i=1\ldots k})$ by a deterministic metamodel *M(***x***)* based on a few selected estimations:

$$\left\{\hat{G}\left(\mathbf{x}_1,\boldsymbol{\theta}_{i=1\ldots k}\right),\hat{G}\left(\mathbf{x}_2,\boldsymbol{\theta}_{i=1\ldots k}\right),\ldots,\hat{G}\left(\mathbf{x}_n,\boldsymbol{\theta}_{i=1\ldots k}\right)\right\} \tag{3}$$

Then, the optimization is performed with the constraint: $M(\mathbf{x}) \le G_{\text{target}}$.

In (3), the same number *k* of MCS is allocated for each **x**, for a total number of simulations *N* = *n k*. The idea of optimal allocation of resources is to increase the accuracy of M by allocating a different number of samples for each **x**, such that the total number *N* remains the same. By extension, one can also think of choosing a smaller number of points *n* and have a greater precision at these points, or inversely, decrease the quality of the responses but increase the number of points.

We assume that k_i is proportional to the computational time. Hence, the problem consists of allocating the computational time between the training points. Also, we relate the accuracy of the response with the computational time. For instance, the variance of probability of failure obtained from MCS is:

$$\text{var}\left(\hat{P}_f\right) = k^{-1} P_f\left(1-P_f\right) \tag{4}$$

P_f is the actual probability of failure and *k* the sample size. If we assume that the range of P_f is very small, it is reasonable to assume that the variance is only related to *k*. Then, the variance of the response is inversely proportional to the computational time:

$$\text{var}\left(\hat{P}_f\right) = \frac{\alpha}{t} \qquad (\alpha \text{ is a constant}) \tag{5}$$

Finally, we can write a computational time constraint in terms of the variances at training points:

$$\frac{1}{v_1}+\frac{1}{v_2}+\ldots+\frac{1}{v_n}=\frac{T_{\max}}{\alpha} \tag{6}$$

If we define the information b_i as the inverse of the variance v_i, we get a linear constraint:

$$b_1+b_2+\ldots+b_n=T_{\max}/\alpha \tag{7}$$

Such formulation is not limited to MCS problems. For most numerical simulators, the fidelity can be represented by a variance, and this variance can be related to the computational time. One can also consider a stochastic simulator with fixed variance; then, adjustable fidelity is obtained by computing several times the response for a given training point: the final response at the training point is taken as the mean of the independent responses, and the response variance is the variance of the simulator divided by the number of repetitions.

2.2. Optimal designs of experiment and allocation of resources

Design optimality is based on the idea that experiments should be chosen to maximize the quality of the statistical inference. Several criteria of quality have been proposed over the years; the most popular include D-, A-, E-, G-, I- optimality [1] [2] [3]. A- and D-optimality aim at minimizing the uncertainty in the parameters of the metamodel. In the framework of linear regression, D-optimal designs minimize the volume of the ellipsoid of confidence of the coefficients, and A-optimal designs minimize its perimeter. Formally, the A and D criterion are respectively the trace and determinant of Fisher's information matrix. I-optimality minimizes the integrated mean square error (or IMSE).

The standard design problem (without resource allocation) consists of choosing the training points $\mathbf{x}_i$ from the design space E such that the design defined by these n points is optimal in the sense of a criterion Q. The problem can formalized as follow:

$$\begin{aligned} &\underset{\mathbf{x}_1,\mathbf{x}_2,\ldots,\mathbf{x}_n}{\text{Max}} && Q(\mathbf{x}_1,\mathbf{x}_2,\ldots,\mathbf{x}_n) \\ &\text{s.t.} && \mathbf{x}_i\in E \end{aligned} \tag{8}$$

Many methods exist to solve the optimization problem. In particular, when D-optimality is used, the designs can be constructed iteratively, using for example Fedorov exchange algorithms or the Wynn-Mitchell algorithm [6].

It is obvious that for variable fidelity responses, Q depends also on the accuracy of each response. Let $\{\mathbf{x}_1,\mathbf{x}_2,\ldots,\mathbf{x}_n\}$ be a set of training points and v_i the variance at point i. Then, given an available computational time T_{max}, we can define an optimization problem for resource allocation:

$$\begin{aligned} &\underset{\substack{\mathbf{x}_1,\mathbf{x}_2,\ldots,\mathbf{x}_n\\ v_1,v_2,\ldots,v_n}}{\text{Max}} && Q(\{\mathbf{x}_1,\mathbf{x}_2,\ldots,\mathbf{x}_n\},\{v_1,v_2,\ldots,v_n\}) \\ &\text{s.t.} && v_1^{-1}+v_2^{-1}+\ldots+v_n^{-1}=T_{\max}/\alpha \\ & && \mathbf{x}_i\in E \end{aligned} \tag{9}$$

Note that n can be considered as an optimization variable too.

Fedorov [5] introduced the notion of normalized design, defined by a set of training points and a proportion of experiments for each point. The sum of the proportions is equal to one; when the total number of experiments is large, the proportions are treated as continuous variables.

3. Strategies for affordable optimizations

The optimization problem, as defined above, is in practice very difficult to solve. Indeed, optimal design problems are known to have many local optima, and the number of design variables increases dramatically with the complexity of the surrogate and the dimension. For instance, fitting a second order polynomial in a two dimensional space requires typically 12 points, which makes a total of 36 design variables (24 coordinates plus 12 variances). In three dimensions, using 20 points, this number rises to 80. Thus, we present here two alternatives to obtain nearly-optimal solutions with reasonable computational expense.

3.1. Optimal information for fixed training points

The first strategy is to perform the optimization only for the variance. The training points $\{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n\}$ are chosen from a classical DoE (orthogonal array, Latin Hypercube). Then, the reduced optimization problem becomes:

$$\begin{aligned} &\underset{v_1, v_2, ..., v_n}{\text{Max}} \quad Q(v_1, v_2, ..., v_n) \\ &\text{s.t.} \quad \sum v_i^{-1} \leq T_{\max} / \alpha \end{aligned} \tag{10}$$

However, choosing a DoE that is known to minimize the criterion for uniform information may not be optimal in the general case.

3.2. Fedorov's algorithm

In his book (pp. 97-114), Fedorov proposed an algorithm to build D and G-optimal designs iteratively as follows: given a current DoE, an optimization is performed to find the point where the prediction variance is maximal. This point is added to the DoE, and the proportion p_{n+1} of experiments allowed to the new point is uniformly transferred from the proportions p_i of the previous training points:

$$\begin{aligned} &p_{n+1} = \alpha \qquad && 0 \leq \alpha \leq 1 \\ &p_i = (1-\alpha)\, p_i, && i = 1...n \end{aligned} \tag{11}$$

α can be chosen such that the criterion is maximum.

After a few iterations, a nearly-optimal DoE is obtained that consists of a set of training points and a proportion of computational time given to each training point. This algorithm often leads to designs with many points with high variance. The number of points can be reduced by agglomerating points close to each other and discarding points with high variance not close to any other point.

Fedorov's algorithm has some strong advantages: indeed, if an infinite number of iterations is performed, the algorithm converges to an optimum[1]. Also, it is computationally attractive since it requires a single optimization in order to find the point with maximum prediction variance (the optimal α can be obtained analytically). However, one of the drawbacks is that it does not discriminate between the training points of the current DoE: their variances are increased by the same quantity α, even if one point is useful and the other is not.

3.3. Simultaneous optimization of training points and observations

The algorithm we propose also constructs nearly-optimal DoEs sequentially, in order to optimize both variances and training point locations at the same time. At each iteration, we consider both options of adding a point with given variance, or decrease the variances at existing data points. The algorithm is described in Table 1; we use the following notations:

- T_{init}: computational time used to compute the initial DoE
- $T_{current}$: computational time already used
- $\mathbf{X}_{\text{init}}$, $\mathbf{X}_{\text{current}}$: initial and current DoE
- $\mathbf{x}_{new}$: new training point
- v_i, v_{new}: variances at $\mathbf{x}_i$ and $\mathbf{x}_{new}$; v_{new}^{-1} is the time 'invested' for each iteration
- n_{init}, $n_{current}$: initial and current number of training points

[1] Theorem 2.5.3, pp. 102-104

Table 1. Algorithmic description of the iterative construction of DoE.

Define: T_{max}, T_{init}, n_{init} and b_{new} (insuring: $T_{init} \le T_{\max}$)		
Generate the initial DoE: $\mathbf{X}_{init} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_{n_{init}}\}$		
Optimize the v_i's for X_{init}: $\underset{v_1, v_2, ..., v_{n_{init}}}{\text{Max}} \quad Q\left(\mathbf{X}_{init,} v_1, v_2, ..., v_{n_{init}}\right)$ s.t. $\sum v_i^{-1} \le T_{init}$		
Set: $T_{current} = T_{init} \quad \mathbf{X}_{current} = \mathbf{X}_{init} \quad n_{current} = n_{init}$		
While $T_{current} \le T_{\max}$		
	1- Find the best x_{new}: $\underset{\mathbf{x}_{new}}{\text{Max}} \quad Q\begin{pmatrix} \{\mathbf{X}_{current}, \mathbf{x}_{new}\}, \\ \{v_1, v_2, ..., v_{n_{current}}, v_{new}\} \end{pmatrix}$ s.t. $\mathbf{x}_{new} \in E$	2- Find the best reduction of variance: $\underset{v_1, v_2, ..., v_{n_{current}}}{\text{Max}} \quad Q\left(\mathbf{X}_{current}, \{v_1, v_2, ..., v_{n_{current}}\}\right)$ s.t. $\sum_{i=1}^{n_{current}} v_i^{-1} \le T_{current} + v_{new}^{-1}$ $\quad v_i \le v_{i\_current} \quad i = 1, ..., n_{current}$ $v_{i\_current}$ is the variance at $\mathbf{x}_i$ at the previous iteration (the variances can only increase)
	Choose the best option between 1 and 2. If 1 is better: $\mathbf{X}_{current} = \{\mathbf{X}_{current}, \mathbf{x}_{new}\}$ $n_{current} = n_{current} + 1$	
	Update time: $T_{current} = T_{current} + v_{new}^{-1}$	

One advantage of such formulation is that the total number of points does not need to be decided beforehand. Choosing a small value of T_{init} and a large v_{new} will increase the number of iterations and the quality of the optimum found; on the other hand, a large T_{init} and small v_{new} will reduce the chance of having too many training points (a small dataset is often preferable for many applications).

4. Application to linear regression analysis

4.1. The Regression model

In this study, we limit our investigation to linear regression models. Since the variances are different at each point, the framework considered here is linear regression under heteroskedastic conditions.

The polynomial response surface model defines the response as the sum of a linear component plus measurement error. Given a set of design points: $\{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n\}$, the polynomial response surface model, in matrix notation, is defined as follow:

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon} \tag{12}$$

Where:

$$\mathbf{Y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \qquad \mathbf{X} = \begin{bmatrix} \boldsymbol{\xi}(\mathbf{x}_1) \\ \boldsymbol{\xi}(\mathbf{x}_2) \\ \vdots \\ \boldsymbol{\xi}(\mathbf{x}_n) \end{bmatrix} \qquad \boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_p \end{bmatrix} \qquad \boldsymbol{\varepsilon} = \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix} \quad \boldsymbol{\xi}(\mathbf{x}) = \left[\xi_0(\mathbf{x}), \xi_1(\mathbf{x}), ..., \xi_p(\mathbf{x})\right]$$

y_i is the simulator response at $\mathbf{x}_i$, $\xi_j(\mathbf{x})$ are the polynomial basis functions, β_j the weights and ε_i the measurement errors. We denote by v_i the variance of the noise ε_i; we assume that v_i is inversely proportional to the computational time used to compute the response y_i.

Since all the v_i are not equal, the ordinary least square estimator of $\boldsymbol{\beta}$ is not appropriate. Instead, we estimate the coefficients $\boldsymbol{\beta}$ by the generalized least square estimator $\boldsymbol{\beta}^*$ [8]:

$$\boldsymbol{\beta}^* = \left(\mathbf{X}'\mathbf{V}^{-1}\mathbf{X}\right)^{-1}\mathbf{X}'\mathbf{V}^{-1}\mathbf{Y} \tag{13}$$

With: $V = diag\left[v_1, v_2, \ldots, v_N\right]$.

At any unsampled location $\mathbf{x}_{pred}$, the prediction variance is [8]:

$$Var\left[y(\mathbf{x}_{pred})\right] = \mathbf{x}_{pred}'\left(\mathbf{X}'\mathbf{V}^{-1}\mathbf{X}\right)^{-1}\mathbf{x}_{pred} \tag{14}$$

For this example, we choose to minimize the IMSE over the design space E:

$$IMSE = \int_E \operatorname{var}\hat{y}(\mathbf{x})dx = \int_E \mathbf{x}'\left(\mathbf{X}'\mathbf{V}^{-1}\mathbf{X}\right)^{-1}\mathbf{x} \tag{15}$$

We compute the integral by numerical quadrature using Simpson's rule.

4.2. Results for 2nd order polynomial in a 2D space

First, we fit a second order polynomial in a square target domain E. We assume that our computational budget corresponds to 12 training points with a variance of 1000 (two times more points than coefficients).

We consider three standard DoEs: a Full factorial (FF) design, a FF design without the central point and a FF design with four additional points situated in the center of the four subsquares. For each of these designs, we compare the IMSE criterion for uniform variances and optimal variances (found as described in *3-1*). Since the DoE's have different number of points, the uniform variances take different values in order to satisfy the same time constraint: 667 for 8 points, 750 for 9, 1083 for 13.

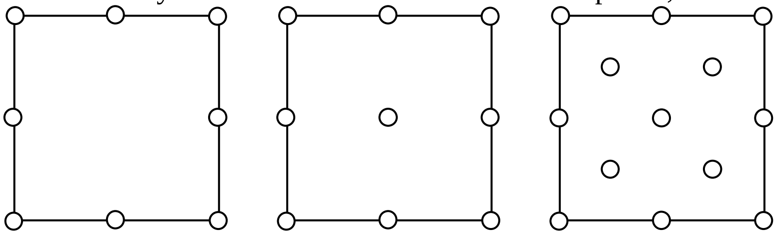

Figure 1. Three reference DoE's: Full factorial (FF) (center), FF without center point (left), FF with four additional points (right).

The optimization is performed using Covariance Matrix Adaptation Evolution Strategy (CMAES) (which is known to be a competitive algorithm for global optimization with a large number of variables [10]). A linear penalty function with offset is used for the time constraint. The IMSE criterion is computed using a 32x32 grid. The iterative procedure is not evaluated here, since only a few iterations (five) can be performed here. Results are reported in Table 2.

Table 2. IMSE criteria for classical DoEs with uniform and optimal variances.

Type	Nb of pts	Variance	IMSE	Improvement
FF without central point	8	Uniform (667)	503.61	8.5 %
	8	Optimal	460.58	
FF	9	Uniform (750)	352.16	9.2 %
	9	Optimal	**319.63**	
FF + 4 center points	13	Uniform (1083)	349.25	7.7 %
	13	Optimal	322.27	

For all three DoEs, optimizing variances improves slightly the IMSE criterion. For the 13-point DoE, the optimal variances correspond to the optimal FF: the values at the additional four points tend to infinity. The difference is due to computational limitations (the variances are bounded by 10^6).

The FF design with optimal variances is represented in Figure 2. The optimal design consists of equal variances on the edges and a smaller variance in the center of the domain.

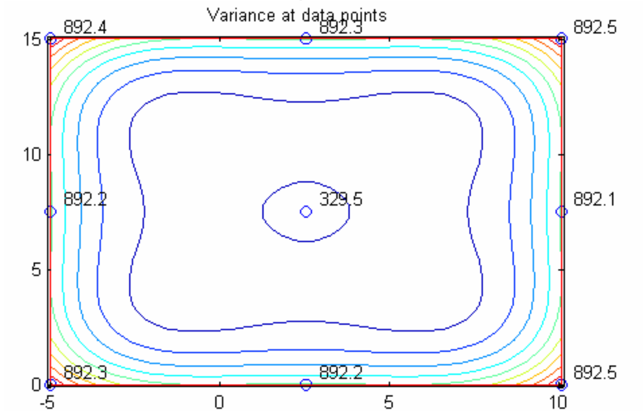

Figure 2. Full factorial design with optimal variances and prediction variance contour lines. The numbers next to the circles represent the variances at these points; the contour lines show the profile of the prediction variance.

Let us consider the RBDO problem described in 2.1. and assume that the constraint is accurately modeled with a second order polynomial. Using the relation between variance and number of simulations described in (6), we can allocate optimally the MCS to the 9 points of the orthogonal array to obtain the best IMSE. For instance, if a total of 1000 MCS can be afforded, the optimal allocation would be to use 248 of them for the center point, and 94 for each of the other points.

4.3. Results for 3rd order polynomial in a 3D space

Now, we consider a substantially more complicated problem which is the fitting of a 3rd order polynomial with interaction terms in a 3D (cubic) space. The polynomial has 20 coefficients; our computational budget corresponds to a 40-point DoE with uniform response variances equal to 1000.

We propose to use two classical designs: an orthogonal array (OA) with 27 points, completed with the centers of the 8 inner cubes, and an LHS design with 40 points (with maximum minimum distance criterion). For these designs, we present the results for uniform variances (875 for OA, 1000 for LHS) and optimal variances (found using CMAES).

We also present the results for the iterative process previously described. The initial design is chosen as a 21-point LHS design with maximin criterion; 19 iterations are performed. The variance of each new point is chosen as 1000; the time used to compute the initial design corresponds to the time needed to compute 21 points with a uniform variance equal to 1000.

To measure the respective effects of space filling and optimal variances, we also run a simplified version of the iterative algorithm, where a new point is generated at each iteration with a fixed variance. Since all the DoE's, except the OA, are random, results are averaged over 10 repetitions. Results are presented in Table 3.

Table 3. IMSE for several DoEs. Numbers in parenthesis are standard deviations over 10 repetitions

DoE	Variance	IMSE	Improvement
LHS, 40 points	Uniform	1755 (350)	17 %
	Optimal	1456 (280)	
OA 27 points + 8 centers of inner cubes	Uniform	374	12 %
	Optimal	330	
Iterative with uniform variances	Uniform	357 (10)	x
Full iterative process	Optimal	248 (13)	

First, we see that the LHS DoE is not as good as the other DOEs. Indeed, LHS is not aimed to provide a low IMSE. Using optimal variances on such DoE allows us to reduce the IMSE by 17%. The orthogonal array provides much lower IMSE; and using optimal variances reduces the IMSE by 12%.

The iterative process provides the best IMSE criterion. It is interesting to note that it is substantially better than an iterative process with uniform variances. This shows that the algorithm combines well the space filling and the optimal variances effects. From the 10 repetitions, we saw that in average, adding a point is chosen 13 times (for a total of 34 points) and reducing variances 6 times. Compared to a 'good' classical DoE for IMSE, which is the OA, we improve the criterion by 34%.

In the present configuration, we invested more than half of the computational time in generating the first 21 points, and used quite large steps to limit the number of iterations to 19. It may be possible to obtain better results by allocation less time to generate the initial design (that is, to increase the variances for the initial design) and use smaller steps in order to allow more iterations.

5. Conclusions

We addressed the issue of allocating optimally the computational time to fit a metamodel, when the response at training points has a tunable fidelity. We showed that the problem can be formulated as an optimal design problem, with additional variables and a constraint on the computational time. Since the optimization is very challenging, we proposed two suboptimal strategies: one is to fix the training points and optimize the response variance only; one is to use an iterative process to get a trade-off between space filling and response precision.

We applied our procedures to the framework of linear regression. We showed that for polynomial approximations, using optimal variances allows us to increase significantly the quality of the surrogate fitting. When higher dimension and model complexity are considered, the iterative process seems promising compared to classical strategies.

References

[1] Kiefer, J., Wolfowitz, J.: *Optimum designs in regression problems*, Annals of Mathematics Statistics, 30, 271-294 (1959)

[2] St. John, R.C. , Draper, N.R., *D-optimality for regression designs: a review*, Technometrics, Vol. 17, No. 1 (Feb., 1975), pp. 15-23

[3] Steinberg, D. M., Hunter, W.G.: *Experimental design: review and comment*, Technometrics, Vol. 26, No. 2 (May, 1984), pp. 71-97

[4] Elfving, G., *Optimum allocation in linear regression theory*, Ann. Math. Stat. 23 (1952), p. 255/262.

[5] Fedorov, V. V., *Theory of Optimal Experiments*, Academic Press, New York, 1972.

[6] Cook, R.D., and Nachtsheim, C.J., *A Comparison of Algorithms for Constructing Exact D-optimal Designs*, Technometrics, 22 (1980) 315-324

[7] J. Sacks, S.B. Schiller, W.J. Welch, *Designs for Computer Experiments*, Technometrics, 31, No. 1, 1989, p. 41-47.

[8] Draper, N.R. and Smith, H. *Applied Regression Analysis*, Wiley Series in Probability and Statistics (1998).

[9] Myers, R.H., Montgomery, D.C., *Response Surface Methodology: Process and Product Optimization Using Designed Experiments*, 2nd Edition, Wiley Series in Probability and Statistics (2002).

[10] Hansen, N, S. Kern (2004). Evaluating the CMA Evolution Strategy on Multimodal Test Functions. In *Eighth International Conference on Parallel Problem Solving from Nature PPSN VIII, Proceedings*, pp. 282-291, Berlin: Springer