Global Journal of Researches in Engineering: G
Industrial Engineering
Volume 14 Issue 3 Version 1.0 Year 2014
Type: Double Blind Peer Reviewed International Research Journal
Publisher: Global Journals Inc. (USA)
Online ISSN: 2249-4596 & Print ISSN: 0975-5861

Development of a Hybrid Metamodel based Simulation Optimization Algorithm

By Farhad Ghassemi Tari & Zohreh Omranpour
Sharif University of Technology, Iran

Abstract- In this paper, a metamodel based hybrid algorithm was developed for optimization of digital computer simulation models. The simulation models are considered to be computationally expensive. It is also considered to have a single stochastic and unconstrained response function. The hybrid algorithm is developed by modification and integration of several concepts and routines. We employed the nested portioning and the particle swarm optimization algori-thms to develop an efficient search mechanism for the hybrid algorithm. Then we integrated the modified Kriging metamodel to the search mechanism for facilitating the function fitting processes of the simulation's output. The efficiency of the developed hybrid algorithm was then evaluated through computational experiments. Ten complex test problems were selected from the literatures and the efficiency of the developed hybrid algorithm was evaluated by comparing its performances against three known algorithm which are cited in the literature. The result of these computational experiments revealed that the developed hybrid algorithm can provide very robust solutions with a very low computational effort.

Keywords: *simulation, optimization, nested partitioning, stochastic kriging, particle swarm optimization.*

GJRE-G Classification : *FOR Code: 290502p*

Strictly as per the compliance and regulations of:

Development of a Hybrid Metamodel based Simulation Optimization Algorithm

Farhad Ghassemi Tari [α] & Zohreh Omranpour [σ]

Abstract- In this paper, a metamodel based hybrid algorithm was developed for optimization of digital computer simulation models. The simulation models are considered to be computationally expensive. It is also considered to have a single stochastic and unconstrained response function. The hybrid algorithm is developed by modification and integration of several concepts and routines. We employed the nested portioning and the particle swarm optimization algori-thms to develop an efficient search mechanism for the hybrid algorithm. Then we integrated the modified Kriging metamodel to the search mechanism for facilitating the function fitting processes of the simulation's output. The efficiency of the developed hybrid algorithm was then evaluated through computational experiments. Ten complex test problems were selected from the literatures and the efficiency of the developed hybrid algorithm was evaluated by comparing its performances against three known algorithm which are cited in the literature. The result of these computational experiments revealed that the developed hybrid algorithm can provide very robust solutions with a very low computational effort.

Keywords: simulation, optimization, nested partitioning, stochastic kriging, particle swarm optimization.

I. Introduction

Digital computer simulation models have been very successful approach for analyzing the complex systems. In simulation models the analytical expression for input/output relationship is generally unavailable and hence conducting its output performance analysis is a cumbersome task. Based on Barton and Meckesheimer [2] simulation optimization is defined as a repeated analysis of simulation models with different values of design parameters, in an attempt to identify best simulated system performance. Since 1950when simulation optimization has been introduced as a new area for research, many practical problems are modeled as simulation optimization problems and successful results have been achieved. Some recent achievements are the works of Liu and Maghsoodloo, [17] and Kleijnen et al.[14]. Also a number of software packages have been developed with the ability of simulation optimization and have been added to some well-known simulation software's. However, the related literatures are still waiting for developments of more new robust methods with the high results efficiency. For recent surveys in this field, we acknowledge the research studies conducted by Fuand Glover [7], Tekin and Sabuncuoglu [28], Henderson and Nelson [8] and Kleijnen [13].

Since simulation models have stochastic response they potentially require extensive runtime. When an optimization routine is incorporated to the simulation models, this problem will intense dramatically. Due to the stochastic nature of the simulation models, the output must be estimated by averaging of the outputs values over reasonable number of replications [10]. In addition the simulation models are usually considered as the black-box models in which, the output function is not usually expressed explicitly. Therefore determining the best combination of controllable variables which provide the best output value can be done either by fitting an explicit function to the output values or by one of the algorithmic input-output optimization routines.

One of the approaches is the use of metamodel-based methods. Metamodels provide deterministic objectives, which surrogate simulation models, and generally need fewer computational efforts. One of the powerful metamodel, addressed in the literatures, is Kriging method. Kriging is an interpolation method which initially developed for using in spatial statistics. Thismethod which is initially proposed by Krige [15]was applied by Cressie [5] in geology. Later it was applied in areas like economic and modeling black box computer experiments [9].

The efficiency of this method in deterministic simulation models has proven [11,20, 26]. Based on the good results obtained in deterministic simulation, Beer and Klijnen [3] applied this method to the random simulation models. With considering successful application of Kriging methodology in design and analysis of deterministic computer experiments Ankenman et al [1]. Introduced stochastic Kriging (SK) metamodel inspired by Kriging methodology. Actually they extend Kriging methodology which applied successfully in deterministic simulation as a global metamodel, to the stochastic simulation. Despite of high accuracy and successful results, SK has not been use as metamodel in numerous papers and has not been use to any of the commercial simulation software package.

A powerful population based metaheuristics which can be employed for solving large-scale optimization problems efficiently is nested partition (NP) method. This method introduced by Shi and Ólafsson

Author α σ : Sharif University of Technology, Tehran, Iran.
e-mails: ghasemi@sharif.edu, zohreh_omran@yahoo.com

[21] inspired by adaptive partitioned random search (APRS) [27] and branch and bound algorithm. Concentrating search effort in some regions which are most likely to have global optimum has been the key idea in developing NP method and this goal can be achieved by partitioning.

NP method has been used in both deterministic [21] and stochastic [22] optimization problems and also has been successfully applied in many areas, such as planning and scheduling, logistics and transportation, supply chain design, data mining and health care and task assignment. This method also is used for solving some difficult optimization problems like traveling sales man problem [23] and production scheduling problems [21]. More details about these applications are provided in [4, 11]. Based on Shi and Ólafsson [25] the NP results significantly depend on the method of sampling and low quality samples can affect the final result. In the young literature of NP, using other heuristics in order to improve samples quality is common.

Using heuristic search method for optimizing simulation models has been very successful approaches. The majority of researches in this area focus on some well-known algorithms like genetic algorithm (GA), simulated annealing (SA), and particle swarm optimization (PSO). PSO method is a global population-based metaheuristic which developed by Kennedy [12] and Eberhart [6] in 1995for optimizing nonlinear programming problems. The key idea of this algorithm was derived from movement of flying birds. There is a population (swarm) which consists of some particles. These particles are representative of solutions in feasible region. Each particle position and velocity updated based on the best performance of the particle achieving so far (its own previous best position) and the best performance obtained by any other particles.

In this paper we developed a metamodel-base simulation optimization hybrid algorithm. The developed algorithm is hybrid by the fact that it is constructed by modification and combination of several optimization routines. We used stochastic Kriging (SK) metamodel for fitting a functional relation to the input output of our simulation models. SK actually is a global metamodel, since it is fitted on the whole feasible space. We then incorporated the NP method into our hybrid algorithm. Through the NP method, the developed hybrid algorithm will concentrate its search effort in the regions which are most likely contained the global optimum. And finally we integrated the PSO method as the searching mechanism for improving the searching process of the proposed hybrid algorithm. Through these integrations a hybrid algorithm for optimization of the digital simulation models is developed, which is described in the next sections of this manuscript in more detailed.

II. Problem Statements

In this article we focus on the simplest optimization problem which is an unconstrained simulation model with the continuous variables and a single output. We aim to minimize the expected value of this univariant output. Despite the simplicity, these kinds of problems have many applications in practice. Examples are(*s, Q*)inventory management simulation, inventory production simulation models in logistics and operation management and queuing simulation models [13].

This problem is formulated as follow:

$$\min_{x} E(w(x)) \tag{1}$$

$$l_j \le x_j \le u_j \quad j = 1,2,...,d \tag{2}$$

Where $x = [x_1, x_2, ..., x_d]^T$ denotes a *d*-dimensional input vector and w indicates the output value. The expected value in the objective function is estimated by simulation and we consider only box constraints where l_j and u_j are the lower and upper bounds of the input variables x_j respectively.

III. Development of the Hybrid Algorithm

The developed hybrid algorithm is a metamodel-based optimization process. We used the stochastic Kriging (SK) metamodel and employed a sequential experiment design for validating its results. In optimization part we used the hybrid metaheuristic algorithm of nested partition (NP) method, which benefits of quick convergence for large scale problems with high computational efforts and we also used the advantages of Particle swarm optimization (PSO) algorithm for its simplicity and good local search ability.

During the process of developing the hybrid algorithm, we first developed an algorithm with the integration of the NP and the PSO algorithm, and we called it PSPO algorithm [19]. The efficiency of the PSPO algorithm was then tested through a computation experiment. As the result, we found that although the solution obtained by this algorithm was either optimal or closed to the optimal, but the computational efforts to obtain the solution were extensive. Due to this problem we directed the path of our research to the metamodel based algorithm and we developed the hybrid algorithm.

Fig.1 represents a flow diagram of the developed hybrid algorithm. According to Fig.1, the first part of the hybrid algorithm includes initial sampling (step 1) and simulating this sample points with a number of specified iteration (step2). The second stage

is an iterative process for fitting metamodel using the input output data (step3) and then validating metamodel by the method developed by Liu, Nelson, and Staum [18]. Their method is actually based on cross-validation method and developed for stochastic Kriging metamodels (step4). Finally the last stage, algorithm starts with primary partitioning of feasible space (step 5) and then sampling and improving these sample points in each partition (step6). By using the validated metamodel and improved samples we computed the fitness value of each partition and then determine the best partition (the most promising partition) and consequently the best point (step7). Actually this point was used for re-partitioning the solution space and adds it to the experiment design process, which can be used for updating the metamodel (step8). After this step, the algorithm checks the existence of any significant improvement. If a significant improvement is not achieved, the hybrid algorithm performs a pre-specified number of simulation runs, denoted by $I^{\max}$, and then stops and accepts the best obtained point. Otherwise it loops back to step 6.

For presenting a detailed description of the algorithm steps, let us show the whole feasible space by Θ, the number of algorithmic iteration by k and the most promising partition by $\sigma(k)$. Now, the following sections provide more details for each step.

Step 1 : Performing the initial experiment design

The first step of the hybrid algorithm includes generating some initial simulation observations, based on a statistical sampling. Selecting an efficient experiment design is an important step in the process of developing a metamodel and can affect the hybrid algorithm performances.

Since we intend to use stochastic Kriging metamodel and based on the assumption that there is no former information about output values, we employed the max-min Latin Hyper Cube sampling [13] while checking the box constraints. Considering a *d*-dimensional space, we used the sample size suggested by Kleijnen [14]. Therefore the sample size was designated as $5+2d$ of the initial sample points. In step 5 we will explain more detail regarding the sequential experiment design.

Step 2 : Simulating the initial design points

Stochastic simulation models have stochastic output values, so we should obtain an average of simulation outputs over the number of replications. If we denote n_i as the number of simulation runs in a design point x_i, the average will be obtained by:

$$\overline{w}(x_i) = \frac{\sum_{l=1}^{n_i} w_l(x_i)}{n_i} \qquad (3)$$

Where $w_l(x_i)$ represent simulation output in the l^{th} iteration. Fordetermining the number of simulation run (replication) in each design point, we used the method of Liu et al. [18] through which the output variance is reduced. At first we performed n_0 replications for every design point x_i. Then we calculated mean $\overline{w}(x_i)$ and the variance $S^2(x_i)$ of these replications. We targeted a relative precision for simulation output γ and then determined the total replication size n_i in each design point. The relative precision for n_i replication in design point x_i was calculated by $l(x_i, n_i; \alpha) \big/ |w(x_i)|$ which we required it to be less than γ. Where $l(x_i, n_i; \alpha)$ is a half-width of the $(1-\alpha)$ confidence interval for output $w(x_i)$. We then used the following condition, proposed by Law [16], for obtaining n_i :

$$\frac{l(x_i, n_i; \alpha)}{|\overline{W}(x_i)|} \le \frac{\gamma}{1+\gamma} \qquad (4)$$

And therefore n_i is obtained as:

$$n_i = \max\left\{n_0, \left\lceil \left(\frac{(1+\gamma) t_{n_0-1, 1-\alpha/2} S(x_i)}{\gamma \overline{W}(x_i)} \right)^2 \right\rceil\right\} \qquad (5)$$

As the result, $n_i - n_0$ more replication runs in the design point x_i should be executed. After determining design points and number of replication in each design point, simulation of all the design points can be executed.

Step 3 : Fitting stochastic Kriging metamodel

By using initial design points and simulation model output (I/O data) we fit stochastic Kriging motamodel, and used it for estimation of the simulation output. Considering m initial design points, the stochastic Kriging predictorwhich optimizes mean square error (MSE) has the form of:

$$\hat{w}(x_c) = \hat{\beta}_0 + \hat{\tau}^2 R_M(x_c,.;\hat{\theta})^T \left[\hat{\tau}^2 R_M(\hat{\theta}) + \hat{\Sigma}_\varepsilon\right]^{-1} \times \left(\overline{w} - \hat{\beta}_0 1_m\right) \quad (6)$$

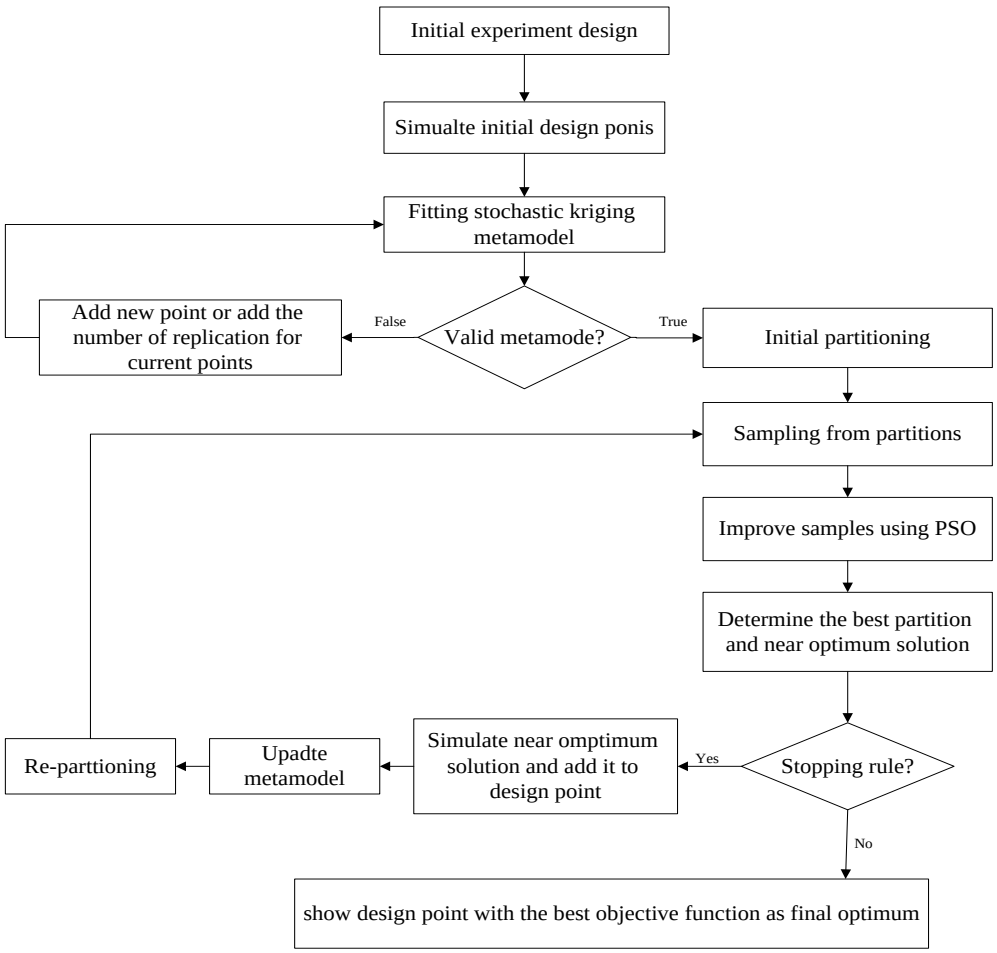

Figure 1 : The flow diagram of the hybrid algorithm

Where x_c is a point to be predicted, R_M is the correlation between two points which depends on the distance between points and a function of $\hat{\theta}$, $\hat{\Sigma}_\varepsilon$ is an $m \times m$ matrix and is calculated by the following formula:

$$\hat{\Sigma}_\varepsilon = Diag\left\{\hat{V}(x_1)/n_1, \hat{V}(x_2)/n_2, ..., \hat{V}(x_m)/n_m\right\}$$

where

$$\hat{V}(x_i) = \frac{1}{n_i - 1}\sum_{j=1}^{n_i}\left(w_j(x_i) - \overline{w}(x_i)\right)^2,$$

$\hat{\tau}^2$, $\hat{\theta}$, $\hat{\beta}_0$ are estimated by solving the following likelihood equation:

$$l(\beta_0, \tau^2, \theta) = -\ln\left[(2\pi)^{m/2}\right] - \frac{1}{2}\ln\left[\left|\tau^2 R_M(\theta) + \Sigma_\varepsilon\right|\right] - \frac{1}{2}(\overline{w} - \beta_0 1_m)^T\left[\tau^2 R_M(\theta) + \Sigma_\varepsilon\right]^{-1}(\overline{w} - \beta_0 1_k), \quad (7)$$

and $\overline{w} = \left(\overline{w}(x_1), \overline{w}(x_2), ..., \overline{w}(x_m)\right)^T$, where $\overline{w}(x_i)$ calculated by equation (3) and 1_m is avector of ones with a compatible dimension.
For more details, we refer the reader to Ankenman, Nelson, and Staum [1].

Step 4 : Validating of the metamodel

For checking the validity of metamodel we used the leave-one-out cross-validation method. The main goal of this method is to decrease the difference between real simulation output in a specific design point, $w(x_i)$, and the metamodel prediction for the same design point, after eliminating the point designated by $\hat{w}^{(-i)}(x_i)$. In other words, the key idea is to control the relative leave-one-out prediction error at each design point which is calculated as follow:

$$\frac{\left|\hat{w}^{(-i)}(x_i) - w(x_i)\right|}{\left|w(x_i)\right|} \qquad (8)$$

Liu and coworkers (2010) demonstrated achieving this goal need to control E_i value which is calculated as follow:

$$E_i = \frac{l(x_i, n_i; \alpha)}{\left|\overline{w}(x_i)\right| - l(x_i, n_i; \alpha)} + \frac{\left|w^{(-i)}(x_i) - \overline{w}(x_i)\right|}{\left|\overline{w}(x_i)\right| - l(x_i, n_i; \alpha)} \qquad (9)$$

Where the half-width is a measure of uncertainty in $\overline{w}(x_i)$.

Before cross-validation we should separate design points which are not on the edges, i.e. for design point $x_i = (x_1, x_2, \cdots, x_d)$ in the *d* dimensions, we select point which none of the factors contains the extreme value. This subset of designing points is showed by set *II* and E_i is calculated for this subset.

Based on equation (9) the lack of credibility of E_i originated from two sources:

- Relative difference between simulation output and true value (relative precision of simulation output) which need to add more replication.
- Relative difference between true value and leave-one-out prediction which need to add more design point.

In each iteration, if $E_i > \beta$, even just for one design point, we should add some more design points or add more replication to existing design points. If the first term of E_i is less than $\lambda\beta$, we select a new design point and we add it to the experimental design points. Otherwise, we add more simulation replications for the point with maximum value of E_i. This selection is based on covering complex areas, when λ is a predefined parameter which aims to control the effect of Mont Carlo variability during cross-validation [17]for $\lambda \in (0,1)$. We selected $\lambda = 1/4$ in our experiments;
The procedure for the validation process can be summarized by the following steps:

1. Separate design points which are not on the edges and show this subset by *II*.
2. Determine number of simulation replication n_i in each design points as in equation (5) and set $N_i \leftarrow n_i$.
3. Calculate $\overline{w}(x_i)$, and $S^2(x_i)$ for all design points in set *II*.
4. Calculate E_i as in equation (9) for all design point in set *II*.
5. $i^* \leftarrow \arg\max_{i \in II} E_i$.
6. If $E_{i^*} > \beta$,

6.1. If $\frac{l(x_{i^*}, n_{i^*}; \alpha)}{\left|\overline{w}(x_{i^*})\right| - l(x_{i^*}, n_{i^*}; \alpha)} > \lambda\beta$

a) Run simulation model at x_{i^*}, N_{i^*} more replication and set $N_{i^*} \leftarrow 2N_{i^*}$.
b) Calculate $\overline{w}(x_{i^*})$, and $S^2(x_{i^*})$ again.
c) Go to step 4.

6.2. Else

a) Add a new design point in the middle of distance between x_{i^*} and the nearest design point.
b) Calculate the number of simulation replication in new design point and then calculate $\overline{w}(x_{k+1})$, and $S^2(x_{k+1})$.
c) $k \leftarrow k+1$.
d) Go to step 4.

The main advantage of this validation method is considering the possibility of adding simulation replications in some points which has high error.

Step 5 : Initial partitioning

In optimization process first we should partition experimental area. Different methods have been developed for partitioning in the NP algorithm. In this article for initial partitioning we divided the range of each variable, which is defined by the box constraint, into two halves. Through this way each variable's dimension is divided into two equal parts. Then we use the SK algorithm for determining the potential areas.

Step 6 : Sampling and improving samples

After defining partition boundaries, a uniform random sampling procedure was used for each partition. In this article the PSO algorithm is used to improve the quality of initial samples and generate feasible solutions which improve the effectiveness and efficiency of the NP algorithm. Suppose we have *N* initial random samples in the j^{th} partition denoted by $D_I^j = \{x_I^{j1}, x_I^{J2}, \cdots x_I^{jN}\}$. Using PSO algorithm, the hybrid algorithm generates a sequence of improving solutions until no further improvement is possible. The final solutions which are more likely to be near optimal solution is shown by $D_F^j = \{x_F^{j1}, x_F^{J2}, \cdots x_F^{jN}\}$.

By using high quality samples we increase the probability of selecting correct partition and making correct moves. Assuming simulation runs are computationally very time consuming, in the process of running the PSO, we used the metamodel for estimating outputs instead of using the simulation run outputs.

It should be noted that in the next iterations (except first iteration) we will have five partitions for sampling. More details provide in step8.

Step 7 : Determining winner partition and near optimal solution

Final population is used for estimating promising index and finally determining the winner partition (next most promising region). To achieve this goal we use the validated metamodel to estimate the output for all the final samples which are shown by $Y(x_F^{Ji})$ for i^{th} sample in j^{th} partition.

The best answer for each partition shows fitness coefficient for that partition. For example in minimization problems it is calculated as follow:

$$Y(\sigma_j) = \min_{i\in\{1,2,\cdots N\}} Y(x_F^{ji}) \qquad j = 1,2,\cdots,5 \tag{10}$$

And the partition with the best overall fitness coefficient is chosen as the next most promising partition, with the index as follow:

$$\hat{J}_k \leftarrow \arg Y(\sigma_j) \qquad j = 1,2,\cdots,5 \tag{11}$$

If this index corresponds to a sub region of $\sigma(k), \hat{J}_k \le 4,$ we let this to be the next most promising partition, ie:

$$\sigma(k+1) = \sigma_{\hat{J}_k}(k) \tag{12}$$

But if it belongs to complementary region we backtrack to previous iteration, ie:

$$\sigma(k+1) = \sigma(k-1) \tag{13}$$

All partitions features in different iterations are saved.Also the backtrack-flag is set on. This flag is used for counting the number of back tracking simulation runs.

$$\hat{J}_i = \arg\min_{i\in\{1,2,\cdots N\}} x_F^{ji} \qquad j = 1,2,\cdots,5 \tag{14}$$

So the best overall point will be denoted by $x_k^{opt} = x_F^{\hat{J}_k \hat{i}_{\hat{J}_k}}$ which will be used in next partitioning (step 8). We also compare this point with the best answer which is obtained so far (x^{opt}) and we record the point which has the better performance measure.

We also record the best point which is obtained. The index of the best point in j^{th} partition is as follow:

Step 8 : Updating the metamodel and re-partitioning solution space

In this step the near-optimal point which is obtained from previous step, (x_k^{opt}), is simulated and then it will be added to the experimental design. With this new experimental design we re-fit and re-validate the metamodel. This new metamodel which may be

more likely to have better prediction is used in the next iteration.

As we explained before, if the current near-optimal point belongs to the complementary region we back track and we use previous partitioning, but if this point belongs to the promising region, the algorithm partitions the promising region into four new partitions (four rectangles) somehow that the obtained best point becomes one of the corner of these rectangular while the other corners are kept as the corners of the promising region. It is clear that these regions may not have equal area. All the remaining parts of the region except these four partitions are aggregated together and will be formed as the complementary partition. Having these five partitions, the algorithm checks the stopping rule and if the stopping rule is not satisfied returns back to step 6 for the new sampling.

Step 9 : Checking stopping rule

The hybrid algorithm is iteratively proceeds until no significant improvement is realized. In order to check this condition the algorithm compares the objective function value of the current best point, $\overline{w}(x_k^{opt})$, with the best point obtained among all the previous iterations, $\overline{w}(x^{opt})$. To conduct this it first calculate the*t* - student prediction error as follow:

$$t_0 = \frac{\overline{w}(x_k^{opt}) - \overline{w}(x^{opt})}{\sqrt{\hat{v}(\overline{w}(x_k^{opt})) + \hat{v}(\overline{w}(x^{opt}))}} \tag{15}$$

The current best point is accepted as the best solution point until now, only if $t_0 < t_{1-\alpha,df}$, otherwise, we conclude that there was not any improvement, where is the degree of freedomand is calculated as follow:

$$df = \min(n_k, n_{opt}) \tag{16}$$

In our experiments we set α=0.1. We also set a threshold for maximum number of unimproved replications equal to 30, i.e. I^{max}= 30.
Whenever this threshold is reached, the iterations are terminated. The algorithm is stopped and the best solution point is acknowledged as the final solution point (x^{opt}).

IV. Computational Experiments

For evaluating the performances of the developed algorithm, it is common to employ some known response surface functions, which are presented in the optimization literatures. We selected 10 complex test problems as the bench mark. Due to deterministic nature of these problems, we add a noise to their objective function relations. The main advantage of using these test problems was their complexity of obtaining their optimal solutions.

In this section the developed hybrid algorithm is evaluated against three known algorithms cited in the literatures. Ten unconstrained hard test problems with the following mathematical structures were selected from the literatures.

1. P1; which has just one global minimum, with the following mathematical form:

$$P_1 = \frac{x_1}{\left(1 + x_1^2 + x_2^2\right)} - 2 \le x_j \le 2 \quad j = 1,2 \tag{17}$$

2. P2; which has one local minimum and one global minimum with the following mathematical form:

$$P_2 = 3\left(1 - x_1^2\right)\exp\left(-x_1^2 - (x_2+1)^2\right) - 10\left(\left(x_1/5\right) - x_1^3 - x_2^5\right)\exp\left(-x_1^2 - x_2^2\right) - \left(1/3\right)$$
$$-\exp\left(-(x_1+1)^2 - x_2^2\right) \qquad -3 \le x_j \le 3 \quad \text{for } j = 1,2 \tag{18}$$

3. P3; which has two local minimum and one global minimum, with the following mathematical form:

$$P_3 = -\left(10\left(1 - x_1^2\right)\right)\exp\left(-x_1^2 - (x_2+1)^2\right) - 15\left(\left(x_1/5\right) - x_1^3 - x_2^5\right)\exp\left(-x_1^2 - x_2^2\right) - \left(1/3\right)$$
$$-\exp\left(-(x_1+1)^2 - x_2^2\right) \qquad -3 \le x_j \le 3 \quad \text{for } j = 1,2 \tag{19}$$

4. P4; which is called Six Hump Camel back (SHC) has six local minimum and two global minimum, with the following mathematical form:

$$P_4 = \left(4 - 2.1x_1^2 + {x_1^4}/{3}\right)x_1^2 + x_1x_2 + \left(-4 + 4x_2^2\right)x_2^2 \qquad -3 \le x_j \le 3 \quad \text{for } j = 1,2 \qquad (20)$$

5. P5; which called Himmelblau function (Hmb) and has several local minimum and one global minimum with the following mathematical form:

$$P_5 = \left(x_1^2 + x_2 - 11\right)^2 + \left(x_1 + x_2^2 - 7\right)^2 + 0.1\left(\left(x_1 - 3\right)^2 + \left(x_2 - 2\right)^2\right)$$
$$-6 \le x_j \le 6 \quad \text{for } j = 1,2 \qquad (21)$$

6. P6; which has several local minimum and one global minimum, with the following mathematical form
7. :

$$P_6 = x_1^2 + 2x_2^2 - 0.3\cos\left(3\pi x_2\right) + 0.7 \qquad -100 \le x_j \le 100 \quad \text{for } j = 1,2 \qquad (22)$$

7. P7; which called Booth function and has several local minimum and one global minimum with the following mathematical form:

$$P_7 = \left(x_1 + 2x_2 - 7\right)^2 + \left(2x_1 + x_2 - 5\right)^2 \qquad -100 \le x_j \le 100 \quad \text{for } j = 1,2 \qquad (23)$$

8. P8; which called Michalewics function and has several local minimum and one global minimum with the following mathematical form:

$$P_8 = -\sin\left(x_1\right)\sin^{20}\left({x_1^2}/{\pi}\right) - \sin\left(x_2\right)\sin^{20}\left({\left(2x_2^2\right)}/{\pi}\right)$$
$$-100 \le x_j \le 100 \quad \text{for } j = 1,2 \qquad (24)$$

9. P9; which called Sphere function and has no local minimum except the global one with the following mathematical form:

$$P_9 = x_1^2 + x_2^2 \quad -5.12 \le x_j \le 5.12 \quad \text{for} \quad j = 1,2 \qquad (25)$$

10. P10; which called Brownian function and has three global minimum with the following mathematical form:

$$P_{10} = \left(x_2 - \left({5}/{\left(4\pi^2\right)}\right)x_1^2 + \left({5}/{\pi}\right)x_1 - 6\right)^2 + 10\left(1 - \left({1}/{8\pi}\right)\right)\cos\left(x_1\right) + 10$$
$$-5.12 \le x_j \le 5.12 \quad \text{for } j = 1,2 \qquad (26)$$

Each test problem was solved using the developed algorithm and three above mentioned algorithms. To overcome the problem of stochastic output of the simulation models, twenty replication of simulation model was considered and the test problems are solved by each method. The performance criteria were selected as the number of simulated points and the quality of final results.

Table (1) summarizes the computational results over 20 replications for the four considered algorithms in term of the best result. As it is seen in eight out of ten test problems the developed hybrid algorithm provided the optimal solutions. For two other problems, problems #4 and #5, the objective function values are very closed to the optimal, less than 0.1 and 0.3 percents.

Table (2) summarizes the computational results over 20 replications for the four considered algorithms in term of the average best result. As it is seen in four out of ten test problems the developed hybrid algorithm provided the average solutions equal to the optimal solutions. For the other problems, except problem #3 and #5 the average solutions have the deviations less than 1.1 percent from the optimal solutions. For problem #3 the dilation is 4.06 per cent from the optimal solution. Only for problem #5 we obtained a solution point which is far away from the optimum which is caused to have an average solution with unacceptable deviation from the optimal solution.

Table (3) summarizes the number of simulated points for obtaining the final solution. As we emphasized earlier, it is assumed that the simulation runs are

expensive or from computation points of view they are assumed to be very time consuming. Therefore the number simulation points for obtaining the optimal solution is very critical performance measure in evaluation of the optimization algorithm. As it is seen considering this performance measure, the developed hybrid algorithm outperforms the other thee algorithms. Actually in most of the ten test problems, the developed algorithm provided more than 80 percent saving in the number of simulated points comparing with the average number of points simulated by the other three algorithms.

Table 1 : Best objective function value obtained comparing to other methods

Problem No.	Optimum	Best objective function value			
		NP	PSO	PSPO	Hybrid A.
P1	-0.500	-0.493	-0.500	-0.500	-0.500
P2	-6.081	-6.079	-6.081	-6.081	-6.081
P3	-19.935	-19.934	-19.934	-19.935	-19.935
P4	-1.032	-1.032	-1.032	-1.032	-1.031
P5	0.000	0.006	0.001	0.000	0.003
P6	0.000	0.008	0.000	0.000	0.000
P7	0.000	0.000	0.000	0.000	0.000
P8	-1.801	-1.780	-1.801	-1.801	-1.801
P9	0.000	0.000	0.000	0.000	0.000
P10	0.398	0.398	0.398	0.398	0.398

Table 2 : Average objective function value obtained comparing to other methods

Problem No.	Optimum	Average objective function over replications			
		NP	PSO	PSPO	Hybrid A.
P1	-0.500	-0.344	0.498	-0.500	-0.496
P2	-6.081	-6.080	-5.670	-6.080	-6.080
P3	-19.935	-19.916	-19.935	-19.935	-19.126
P4	-1.032	-1.031	-1.031	-1.031	-1.032
P5	0.000	1.350	1.506	0.002	3.830
P6	0.000	0.000	0.488	0.000	0.011
P7	0.000	0.080	0.000	0.000	0.000
P8	-1.801	-1.733	-1.681	-1.800	-1.821
P9	0.000	0.000	0.000	0.000	0.000
P10	0.398	0.498	0.399	0.398	0.398

Table 3 : Number of simulated points to obtain the final solution

Problem No.	Number of simulation points			
	NP	PSO	PSPO	Hybrid A.
P1	490	220	900	102
P2	570	300	1400	126
P3	520	800	1275	134
P4	810	300	3275	114
P5	530	550	1730	93
P6	610	600	2035	99
P7	290	436	1555	123
P8	390	328	2490	140
P9	240	264	305	112
P10	340	500	1150	137

V. Conclusions and Remarks

In this paper, a metamodel based hybrid algorithm was developed for optimization of the digital computer simulations models. It is assumed that the considered simulation models are computationally expensive, and have a single output function which is nonlinear continuous unconstrained function. By the computationally expensive we mean that, each simulation run is required an extensive computational time.

The hybrid algorithm is developed by modifying and integrating of several concepts and routines. We incorporated NP and PSO algorithms to develop an efficient search mechanism and we modified K metamodel to be applied in stochastic simulation-optimization model (SK), and is integrated to the search mechanism as a metamodel for facilitating the function fitting processes of the simulation's output. As the result a hybrid algorithm is developed.

The efficiency of the developed hybrid algorithm was then evaluated through computational experiments. Six complex test problems were selected from the literatures and the efficiency of the developed hybrid algorithm was evaluated by comparing its performances against three other algorithms. Two algorithms were selected from the existing literatures, and one algorithm was a preliminary developed algorithm by the authors. The result of these computational experiments revealed that the developed hybrid algorithm can provide very robust solutions with a very low computational effort.

References Références Referencias

1. B.L. Ankenman, B.L. Nelson, J. Staum, Stochastic Kriging for simulation metamodeling. Operations Research 58(2002) 371-382.
2. R.R. Barton, M. Meckesheimer,Metamodel-Based Simulation Optimization; Handbooks in Operations Research and Management Science. Elsevier/North Holland,2006.
3. W.C.M. Beers, J.P.C. Kleijnen, Kriging interpolation in simulation: a survey. Proceedings of the 36th Conference on Winter Simulation,2004.
4. W. Chen, L. Pi, L. Shi,Optimization and Logistics Challenges in the Enterprise.Springer Optimization and its Applications,Dordrecht Heidelberg London, New York, 2009.
5. N. Cressie, The origins of Kriging. Mathematical Geology 22(1990) 239-252.
6. R.C. Eberhart, J.Kennedy,A new optimizer using particle swarm theory, Proceeding of 6th International Symposium on Micro Machine and Human Science, 1995.
7. M.C. Fu,F.W. Glover,J. April,Simulation optimization: a review, new developments, and applications. Proceeding of Winter Simulation Conference,2005.
8. S.G. Henderson, B.L. Nelson, Handbooks in Operations Research and Management Science. Elsevier, North Holland, 2006.
9. D. Huang,T.T. Allen,W.I. Notz,N. Zheng, Global optimization of stochastic black- box systems via sequential Kriging metamodels. Journal of Global Optimization 34 (2006)441–466.
10. R.D. Hurrion, An example of simulation optimization using a neural network metamodel: finding the optimum number of kanbans in a manufacturing system. Journal of the Operational Research Society 48(1997)1105-1112.
11. D.R. Jones,M. Schonlau,W.J. Welch,Efficient global optimization of expensive black-box functions. Journal of Global Optimization13(1998)455–492.
12. J. Kennedy, R.C. Eberhart, Particle swarm optimization. Proceeding of the IEEE International Conference on Neural Network, 1995.
13. J.P.C. Kleijnen, Response surface methodology for constrained simulation optimization: an overview. Simulation Modeling, Practice and Theory16(2008) 50-64.
14. J.P.C. Kleijnen, W.Beers, I. Nieuwenhuyse, Constrained optimization in expensive simulation; Novel approach. European Journal of Operation Research 202(2010) 164-174.
15. D.G. Krige,Kriging metamodeling for simulation, Ph.D. Dissertation, Faculty of Economics and Business Administration, Tilburg University, Netherlands,1950.
16. A.M. Law,Simulation Modeling and Analysis. Fourth ed. McGraw-Hill, Boston, 2007.
17. H. Liu,S. Maghsoodloo, Simulation optimization based on Taylor Kriging and evolutionary algorithm. Applied Soft Computing11(2011) 3451-3462.
18. M. Liu,B.L. Nelson,J. Staum,Simulation on demand for pricing many securities. Proceedings of the 2010 Winter Simulation Conference,2010.
19. Z. Omranpour, F. Ghassemi-Tari F, Development of PSPO simulation optimization algorithm. Journal of Industrial & Systems Engineering 5(2012) 352-367.
20. T.J. Santner, B.J. Williams, W.I. Notz,The Design and Analysis of Computer Experiments. Springer-Verlag, New York,2003.
21. L. Shi,S. Ólafsson, Nested partitions method for global optimization.Operation Research48(1998) 390-407.
22. L. Shi,S. Ólafsson,Nested partitions method for stochastic optimization. Methodology and Computing in Applied Probability2 (2000)271-291.
23. L. Shi,S. Ólafsson, N.Sun,A New Parallel Randomized Algorithm for Traveling Salesman Problem. Computer and Operations Research26 (2000)371-394.
24. L. Shi,S. Ólafsson,Nested Partitions Method. Theory and Applications; International Series in Operations Research & Management Science, Springer Publisher, 2008.
25. L. Shi,S. Ólafsson,Nested Partition Optimization. Encyclopedia of Optimization, Springer, 2009.
26. T.W. Simpson, T.M. Mauery, J.J. Korte, F. Mistree, Kriging metamodels for global approximation in simulation-based multidisciplinary design optimization. AIAA Journal 39 (2001)2233– 2241.
27. Z.B. Tang,Adaptive partitioned random search to global optimization. IEEE Transaction on Automatic Control39(1994)2235-2244.
28. E. Tekin,I. Sabuncuoglu, Simulation optimization: a comprehensive review on theory and applications. IIE Transactions36(2004)1067–1081.
29. [1] J. van der Geer, J.A.J. Hanraads, R.A. Lupton, The art of writing a scientific article, J. Sci. Commun. 163 (2010) 51–59.

30. Reference to a book:
31. [2] W. Strunk Jr., E.B. White, The Elements of Style, fourth ed., Longman, New York, 2000.
32. Reference to a chapter in an edited book:
33. [3] G.R. Mettam, L.B. Adams, How to prepare an electronic version of your article, in: B.S. Jones, R.Z.
34. Smith (Eds.), Introduction to the Electronic Age, E-Publishing Inc., New York, 2009, pp. 281–304.

This page is intentionally left blank