Proceedings of the ASME 2007 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference
IDETC/CIE 2007
September 4-7, 2007, Las Vegas, Nevada, USA

DETC2007-34839

APPROXIMATED UNIMODAL REGION ELIMINATION BASED GLOBAL OPTIMIZATION METHOD FOR ENGINEERING DESIGN

Adel Younis[1], Ruoning Xu[2] and Zuomin Dong[1]

[1]Department of Mechanical Engineering
University of Victoria
Victoria, BC Canada (Email: zdong@me.uvic.ca)

[2]School of Information and Electro-Mechanical Engineering
Guangzhou University, Guangzhou, China

ABSTRACT

Computer analysis and simulation based design optimization requires more computationally efficient global optimization tools. In this work, a new global optimization algorithm based on design experiments, region elimination and response surface model, namely Approximated Unimodal Region Elimination Method (AUREM), is introduced. The approach divides the field of interest into several unimodal regions using design experiment data; identify and rank the regions that most likely contain the global minimum; form a response surface model with additional design experiment data over the most promising region; identify its minimum, remove this processed region, and move to the next most promising region. By avoiding redundant searches, the approach identifies the global optimum with reduced number of objective function evaluations and computation effort. The new algorithm was tested using a variety of benchmark global optimization problems and compared with several widely used global optimization algorithms. The experiments results present comparable search accuracy and superior computation efficiency, making the new algorithm an ideal tool for computer analysis and simulation black-box based global design optimization.

1 INTRUDUCTION

1.1 Background

With the rapid advances in Computer Aided Design, Engineering and Manufacturing (CAD/CAE/CAM), virtual prototyping of a new design using computer modeling, analysis and simulation tools has become more common. The computational function modules in CAD/CAE/CAM, including finite element analysis (FEA), computational fluid dynamics (CFD), kinematics/dynamics analysis, motion animation and CNC tool path simulation, automatically evaluate and accurately predict the performance of a mechanical design. It is quite natural to further extend the practice to allow design optimizations be carried out using these virtual-prototyping black-boxes as the objective and constraint functions. These optimizations are used to identify the best combination of design parameters in the complex; black-box based multidisciplinary design problems. However, this type of optimizations often has non-unimodal objective function and non-convex feasible regions, requiring special global optimization search tools. Conventional optimization methods, such as conjugate gradient, quasi-Newton, and sequential quadratic programming algorithms, which perform brilliantly on a typical local optimization problem, are often trapped into a local minimum and unable to identify the global minimum of the design problem. On the other hand, the computation intensive nature of engineering analysis and simulation software makes the use of many mature stochastic global optimization methods very difficult due to the need of extensive and costly evaluations of the objective and constraint functions (Wang *et al*., 2001). An effective method for identifying the global optimum with a reduced number of objective function evaluations is needed to make this new paradigm for design automation and optimization viable.

1.2 Related Work

Efficient global optimization algorithms have found a wide range of applications in sciences and engineering. Much research has already been carried out in their continuous development and improvements. Widely known global optimization techniques include stochastic based algorithms, including Simulated Annealing (SA), Genetic Algorithms (GAs), Particle Swarm Optimization (PSO), and Ant Colony (AC) optimization method, as well as metamodel and approximation based global optimizations.

Simulated annealing was introduced by Kirkpatrick *et al*. (1983), as an intriguing technique for optimizing highly nonlinear functions of many variables and multiple local minima. As a probabilistic combinatorial optimization technique, it is based on analogy to the statistical mechanics of disordered systems. The method searches for the global optimum through a process that simulates the physical annealing process of a solid. Since then simulated annealing has been used in various combinatorial optimization problems, and has been particularly successful in solving various global optimization problems in circuit design, truss design, robotic path planning, and automated generation of sculptured surface models.

Genetic Algorithms were introduced by Holland (1975), and further developed by him and many others. Genetic Algorithms are a class of search procedures based on the mechanics of natural genetics and natural selection (Goldberg, 1989). In the genetic processes of biological organisms, natural populations evolve according to the principles of natural selection and survival of the fittest over many generations. By mimicking this process, genetic algorithms are able to "evolve" solutions to real world problems, if these problems are suitably encoded (Holland, 1975). Genetic algorithms strongly differ in conception from other search methods, including traditional optimization methods. The basic difference is that while other methods always process single points in the search space, genetic algorithms maintain a population of potential solutions (Renner *et al*., 2003). Genetic Algorithms have been used widely in solving engineering optimization problems.

Particle Swarm Optimization is a recently introduced global optimization technique that has been used with great success in the Computational Intelligent area. The method is a population based stochastic optimization technique, developed by Kennedy and Eberhart (1995) and inspired by social behavior of bird flocking or fish schooling. PSO shares many similarities with evolutionary computation techniques such as GAs. The system is initialized with a population of random solutions and searches for optima by updating generations. However, unlike GA, PSO has no evolution operators such as crossover and mutation. In PSO, the potential solutions, called particles, fly through the problem space by following the current optimum particles.

Ant Colony optimization algorithms are multi-agent systems in which the behavior of each ant is inspired by the foraging of the real ants to solve optimization problems (Toksari, 2006). The idea of imitating the behavior of ants for finding good solutions to combinatorial optimization was initiated by Dorigo (1992). The principle of these methods is based on the way ants search for food and find their way back to the nest. During trips of ants a chemical trial called pheromone is left on the ground. The role of pheromone is to guide the other ants towards the target point.

Wang *et al*. (2004) proposed a new global optimization method for expensive black-box functions, assuming the design space cannot be confidently reduced. The method is developed based on novel mode-pursuing sampling (MPS) method which systematically generates more sample points in the neighborhood of the function mode while statistically covers the entire space. The method is applicable to both continuous and discontinuous functions and applies to both constrained and unconstrained optimization problems.

There also exist many global optimization methods that explicitly take into account the high computation cost involved with the evaluation of the objective and constraint functions. These techniques can be roughly divided into two groups, sequential and non-sequential methods. Non-sequential method are aimed at modeling the whole design space with help of dedicated Design of Experiments (DOE) techniques, using a response surface models (Pary *et al*., 2004). This approximation-based optimization method has attracted many attentions in recent years. The approach approximates computation-intensive functions with a simple analytical model, or metamodel. Metamodeling evolves from classical DOE theory, in which polynomial functions are used as response surfaces or metamodels to considerably reduce the number of objective and constraint function evaluations. A recursive approach is often used to improve the accuracy of the modeling and the search. Today, solving complex design and design optimization problems using computation intensive computer models that are constructed using advanced CAD/CAE/CAM systems is becoming increasingly common. However, when the dimension of the search increases and/or when the feasible region of the search presents irregular shape, application of this method becomes increasingly difficult. A sequential method, on the other hand, divides the entire design space into a number of smaller and more manageable regions and identifies the optimum in each region based on their likelihood of containing the global optimum. This approximated unimodal region elimination scheme is more suitable for multidimensional design spaces containing many infeasible areas. Both metamodeling and region elimination are promising computationally efficient global optimization methods that focus on the accurate identification the global optimum with as few objective function evaluations as possible.

Response surface method (RSM) is used for solving a complex optimization problem through approximation, in which a regression model is used to fit a series of planned design of experiments data points to estimate the complex relationship between the design variables and the objective functions (Montgomery, 2001). RSM was initially developed to represent the relationship between the input and the output of a physical experiment by a simple mathematical expression. Later the use of RSM has been extended to engineering analyses that involve the execution of complex computer analysis codes, where RSM found many applications to alleviate the computational burden of such analyses (Kaymaz *et al*., 2005). Having been used effectively as metamodels (Barton, 1998), RSM considers the correlation between the parameters of a process and the obtained results as surfaces in the dimensional space of the variables (Tiernan *et al*., 2005). The simulation community has used metamodels to study the behavior of computer simulations for over twenty-five years. The most popular techniques have been based on parametric polynomial response surface approximations (Barton, 1998). RSM bears a number of appealing features for analysis and simulation based global optimization, including robustness, supporting distributed computation, providing variable sensitivities, and allowing both continuous and discrete variables (Kuehl, 2000). An improved RSM algorithm, Adaptive Response Surface Method (ARSM) was introduced in the group's earlier research (Wang *et al*., 2001), to improve the accuracy and efficiency of global optimal design. Tests on benchmark optimization problems and on industrial fuel cell component and system design optimizations (Wang *et al*., 2001) showed considerable improvements. However, the robustness and capability of the method to handle more complex design problems need to be improved.

In this work, a new sequential, global optimization algorithm based on design experiments, region elimination and response surface model, namely Approximated Unimodal Region Elimination Method (AUREM), is introduced. The method divides the design space into many sub-areas to ensure that the optimization in the sub-area becomes a unimodal function so that a second order polynomial response surface model can be used to replace the black box objective function of numerical analysis and simulation, and conventional optimization algorithms can be used to locate its minimum. The search starts from the most promising sub-area, and identify the global optimum of the design problem by comparing the optima of all sub-areas processed. In this paper, the new algorithm and its implementation are first explained. Tests on the robustness and efficiency of the new algorithm are carried out using a number of benchmark problems, in comparison of several other global optimization methods.

To illustrate the advantages of this newly introduced algorithm, a number of benchmark global optimization problems were solved using the newly proposed global optimization algorithm and other reviewed global optimization algorithms. AUREM obtained better results in most cases, and consistently out performed the stochastic global optimization techniques, making it an ideal tool for identifying the optimal design using complex, black-box based computer analysis and simulation in multidisciplinary design optimization.

2 REGION ELIMINATION IN GLOBAL OPTIMIZATION

2.1 Generic Global Optimization Problem

In the multidisciplinary design optimization, the objective function of the optimization is evaluated through complex; black-box based computer analysis and simulation. These evaluations lead to a group of experiment data

$$S \triangleq \{(\boldsymbol{X}_i, f(\boldsymbol{X}_i)) : \boldsymbol{X}_i \in A\}$$

where $\boldsymbol{X}_i = (x_{i1}, x_{i2}, \cdots, x_{in})$, and n is the dimension of the design problem under study; $\boldsymbol{X}_i$ $(i = 1, \cdots, m)$ is the *ith* value of the design variables in the *n*-dimensional space with *n* vector components; $y_i = f(\boldsymbol{X}_i)$ is the value of the objective function for point $\boldsymbol{X}_i$; and A is the field of interest or the feasible region of the design optimization problem. If *n=2*, the field of interest and several points of the design variable ($\boldsymbol{X}_i$) can then be illustrated as shown in Fig. 1.

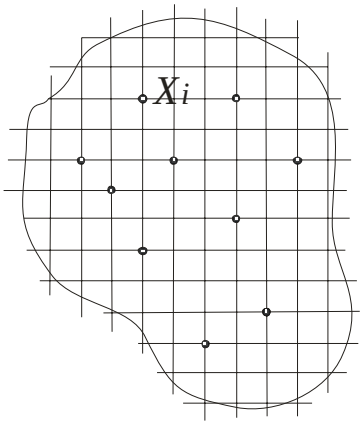

Figure 1: Field of Interest and Points of Design Variable

Note that S forms our design space, and several design points appear in the design space randomly. In addition, the objective function, $f(\boldsymbol{X}_i)$ is normally in an implicit form, non-unimodal, and expensive to calculate since complex computer analysis and simulation are needed to obtain its value. The objective of this work is to identify the global optimum of $f(\boldsymbol{X}_i)$ with a small or considerably reduced number of these black box numerical function evaluations.

2.2 Major Steps of the Proposed Algorithm

The proposed search algorithm consists of the following major steps:

1) Generate a set of DOE data points, $f(X_i)$ $(i=1,\cdots,m)$ over the field of interest;
2) Divide the field of interest S into many unimodal regions;
3) Identify the region that most likely contains the global minimum, and rank the others;
4) Refine the most promising region by adding more experiment points in the region, and introduce an RSM approximation model over the region;
5) Carry out design optimization in the most promising unimodal region to obtain the design optimum of the region and to remove the processed region from the field of interest;
6) Repeat the previous steps until the local optima of the most promising unimodal regions are all located, and the global optimum is identified from these local optima.

3 METAMODELING FOR THE DIVIDED REGION

Properly designed experiments are essential for effective data sampling. In engineering, traditionally a single parameter is varied (perturbed) and the effects are observed. Alternatively, combinations of factor settings are assigned either systematically (e.g. grid search) or randomly to provide an alternative for comparison. Experimental design techniques which were developed for physical experiments are being applied to the design of computer experiments to increase the efficiency of these analyses (Simpson *et al*., 2001).

In this work, due to the implicit form of the black box computer analysis and animation tools, a systematic collection and evaluation on the values of objective function in the global design optimization are carried out using DOE method. RSM is used for solving a complex optimization problem through approximation, in which the regression model is used to fit a series of planned experiments to estimate the complex relationship between design variables and objective functions.

Suppose that we are concerned with a system involving response y which depends on the given input variables $x_1, x_2, ..., x_n$ and their relations can be formulated as

$$y = f(x_1, x_2, ..., x_n)$$

where the form of $f(x_1, x_2, ..., x_n)$ is unknown and perhaps very complicated. The response surface method assumed that f can be approximated by a polynomial function of low order. For quadratic approximation, the second order response surface model has the form:

$$y=\beta_0 + \sum_{i=1}^{n} \beta_i x_i + \sum_{i=1}^{n} \beta_{ii} x_i^2 + \sum_{i=1}^{n} \sum_{j<i}^{n} \beta_{ij} x_i x_j + \varepsilon \quad (1)$$

where ε is the disturbance or model error, and the estimators $\beta_0, \beta_i, \beta_{ii}$ and β_{ij} is determined using least squares regression analysis by fitting of the response surface approximation to result data.

The main aim in the formation of the RSM is to fit a response surface as closely as possible. The coefficients of the response surface using least square method are

$$b = \left(X' X\right)^{-1} X' y \quad (2)$$

where X denotes the design matrix comprising the experimental points, and y represents the response vector obtained from the performance function corresponding to the experimental points (Myers, 1971). The use of RSM has many advantages. The method works very well for unimodal functions, reduce the number of function evaluations, and considerably increase the computation efficiency.

4 REGION ELIMINATION ALGORITHM

Step 1: introduce finite grids in the space of interest, and obtain a set of design experiment data, $f(X_i)$ $(i=1,\cdots,m)$ using DOE method on the selected grid points; set unimodal region counter $l=1$.

Step 2: for unimodal region, *l* carry out the following

Step 2.1: find the minimum function value and its coordinates

$$y^{(l)} \triangleq \min\left\{ f(X_i) : (X_i, f(X_i)) \in S \right\}$$

and denote,

$$X^{(l)} = (x_1^l, x_2^l, \cdots, x_n^l) : f(X^{(l)}) = y^{(l)}$$

If there are several X's corresponding to $y^{(l)}$, just choose one. Put $(X^{(l)}, f(X^{(l)}))$ into C_l, and $X^{(l)}$ into C_l^x. Carry out the following partition:

$$S = \bigcup_{l=1} C_l ;$$

If f is continuous and the number of elements in C_l^x is greater than 2, then the corresponding area of C_l^x exists at least one minimum. At each round, these partitions can be ranked based on their possibility to contain the global minimum from high to low as $C_1^x, C_2^x, \cdots, C_l^x, \cdots$

Step 2.2: Use the coordinates of the minimum function value x_k^l ($k = 1:n$) as the center point to start the search; and assign the function value at this center point as $q = y^{(l)}$

Step 2.3: Identify the turning points (when $f(X) \prec q$, go back in the opposite direction) of the unimodal region through a search for the next point X in S by moving along the positive and the negative directions of the coordinates (e.g. moving right, left, up, and down if n = 2), comparing the function value at each point with the previous function value.

If $f(X) \geq q$, put $(X, f(X))$ into C_l and X into C_l^x, and let $q = f(X)$; go to Step 2.3 for next design space coordinate.

Step 2.4: Subtract the identified unimodal region C_l from the design space S until all of the points in the design space S are visited. Specifically, Let

$$S = S \setminus C_l$$

If $S \neq \varnothing$, let $l = l + 1$, and go to Step 2,

Step 3: After visiting the entire space of interest S, and obtaining divided regions C_l^x by dividing S into C_l^x, generate a field A_l which covers the area of C_l^x $(l = 1, 2, \cdots)$.

Consider the first several C_l^x, which are most likely to contain the global minimum

$$l_i = \min\{x_i^l : (x_1^l, \cdots, x_i^l, \cdots x_n^l) \in C_l^x\}$$

$$r_i = \max\{x_i^l : (x_1^l, \cdots, x_i^l, \cdots x_n^l) \in C_l^x\}$$

where l_i and r_i are the minimum and the maximum values of the points in C_l^x, which can be determined using the relations given previously.

$$A_l = [l_1, r_1] \times \cdots \times [l_n, r_n]$$

Let $A = A_l \cap A$.

Carry out new design experiments in A, and denote the set of experiment data contained in A as S, repeat Step 2.

Terminate the process when the sub region A_l can no longer be changed.

Step 4: For C_l^x, denote;

$$\alpha = \min\left\{|x_i^- - x_i^l|, |x_i^+ - x_i^l|, i = 1, \cdots, n\right\},$$

where;

$x_i^- = \min\left\{x_i : x_i \text{ are the } ith \text{ coordinate of the } x, x \in C_l^x\right\}$ and;

$x_i^+ = \max\left\{x_i : x_i \text{ are the } ith \text{ coordinate of the } x, x \in C_l^x\right\}$

Let

$$x_{ij} = x^{(l)} \pm \frac{\alpha}{2}(e_i \pm e_j)$$

where $e_i = (i = 1, \cdots, n)$ is a unit vector of the ith coordinate direction in R

If $f(x_{ij}) \geq f(x^{(l)})$, next

else

end

Let $\alpha = 0.5\alpha$, repeat the previous step until $\alpha = 1$.

For a two dimensional problem, n = 2, the process is illustrated in Fig. 2. The example shows how a convex and unimodal region is obtained by from a large feasible area. Since we have the center point, x_l, which represents roughly the minimum point in that sub-region. By obtaining x_i^- and x_i^+ we can find the initial value of α, using $\alpha = \min\left\{|x_i^- - x_i^l|, |x_i^+ - x_i^l|, i = 1, \cdots, n\right\}$. By drawing a circle (since n = 2) with radius α, one can obtain the four extreme coordinate points, two in each dimension, as the circle intersects with the axes of the coordinates, represented as X_1, X_2, X_3 and X_4. It is desirable to have more data points inside the circle so that the approximated fitted model will be more accurate. More points can be obtained using the equation for calculating x_{ij} as previously mentioned. These are the middle points on the straight lines that connects X_1, X_2, X_3 and X_4, as shown in Fig. 2. By evaluating the function values at all of these points and compare them with the function value at the center point, a decision can be made weather further reduction of α and a repeat of the same procedure are needed, or the current value of α can be accepted. For a problem with dimension n > 2, α is the radius of an n-D spherical region.

Step 5: Create spheres with radius α that hold the unimodal region inside and use RSM to construct the response surface and find its local minimum using local optimization methods. Specifically, denote $Sp_l^x = \left\{x : \left\|x - x^{(l)}\right\| \leq \alpha\right\}$; find the local minimum in each Sp_l^x by using the second order response surface model.

Step 6: Find the absolute minimum $f^*(x^*)$ among the local minimum $f_{c_i}^*(x^*)$ and identify it as the global minimum of the optimization problem.

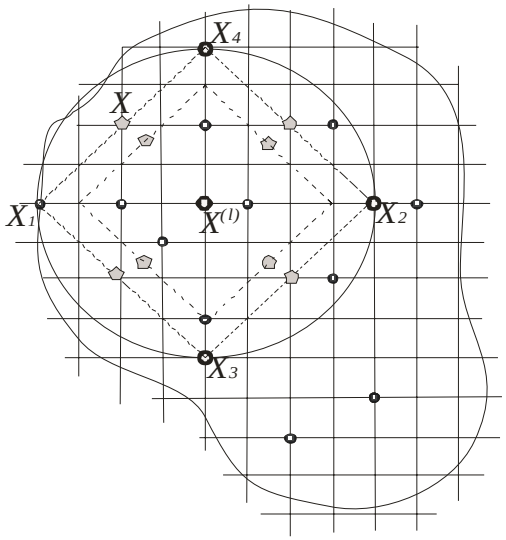

Figure 2: Reduction of Identified Unimodal Area

5 TESTING USING BENCHMARK PROPLEMS

To access the performance of the newly introduced AURE algorithm, in terms of its ability to provide correct search results, robustness, and computation efficiency, the method has been tested using a variety of commonly used benchmark optimization problems. Furthermore, its performance is compared with other well-known global optimization search methods, including SA, GA, PSO, and MPS. The results are presented in this section.

5.1 Benchmark Test 1 - Alpine Function

First the Alpine function is used as the objective function of the optimization in the tests due to its many local minima and one distinct global minimum. The function is given in Eq. 3, and illustrated in Fig. 4. The results are presented in Table 1.

$$f(x_1,...,x_D) = \begin{cases} \sin(x_1)\times...\times\sin(x_D)\sqrt{x_1...x_D} \\ (x_1,...,x_D)\in[0,x_{\max}]^D \end{cases} \qquad (3)$$

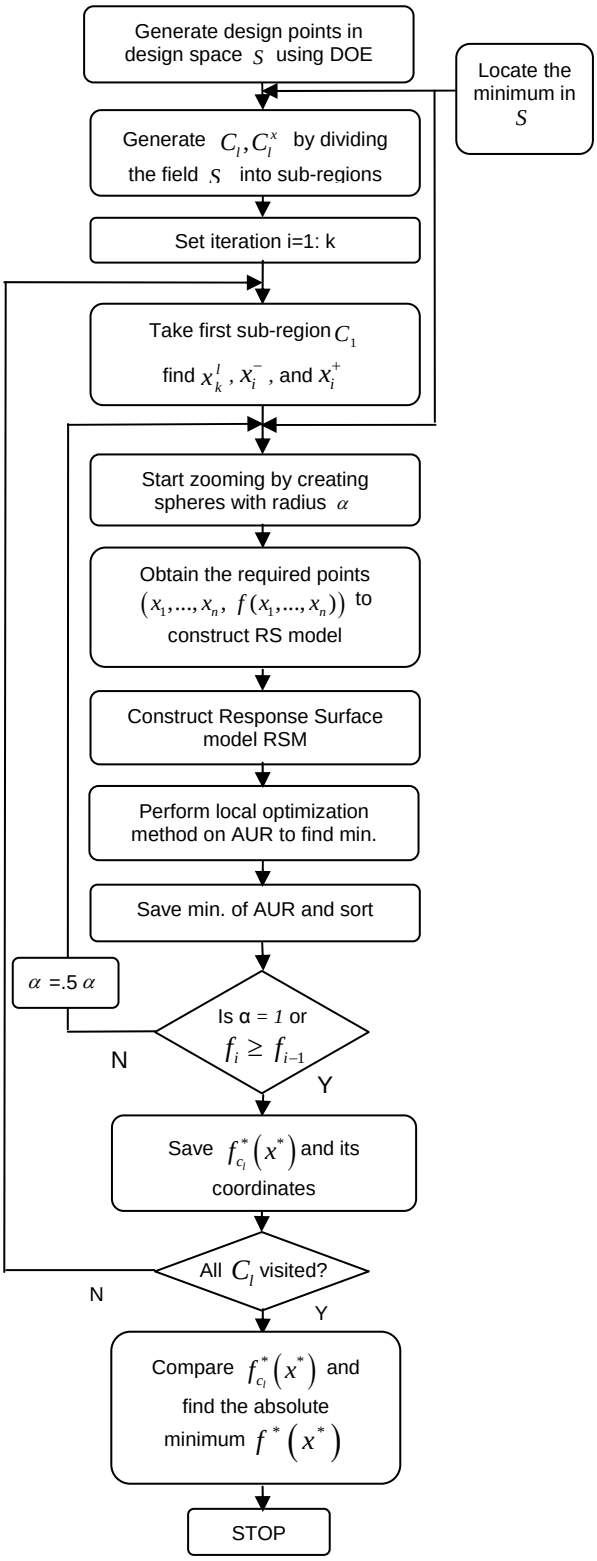

Figure 3: Flow diagram of the proposed method (AUREM)

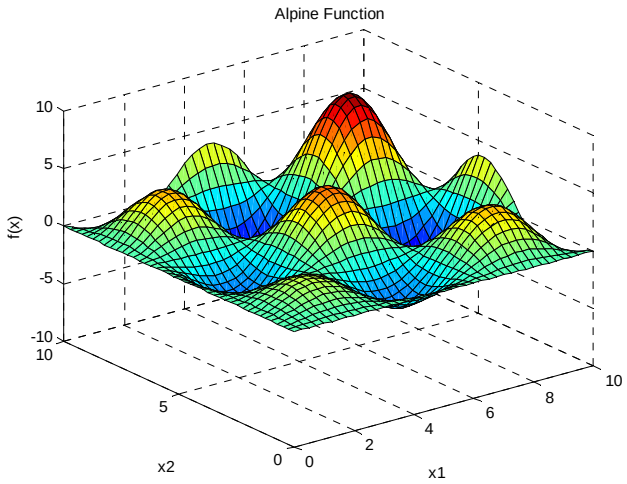

Figure 4: The objective function of benchmark test 1

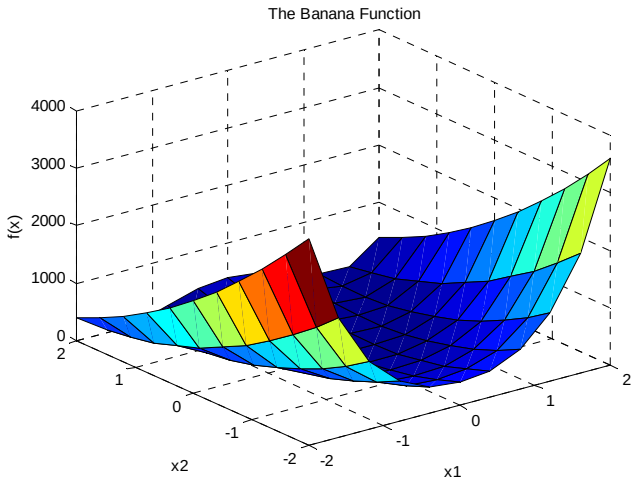

Figure 5: The Objective function of benchmark test 2

Table 1: Test Results on Alpine function

Algorithm Used	Calculated Optimum, X^* and $f(X^*)$			CPU time (sec)
	x_1	x_2	$f(x_1, x_2)$	
PSO	7.9180	4.8046	-6.1291	19.738
GA	7.9171	4.8151	-6.1295	13.649
MPS	7.9040	4.8356	-6.1277	20.139
SA	7.9171	4.8158	-6.1295	12.097
SQP	0.0000	10.000	0.0000	2.5100
AURE	7.9082	4.8244	-6.1290	7.9180

Table 2: Test Results on Banana Function

Algorithm Used	Calculated Optimum, X^* and $f(X^*)$			CPU time (sec)
	x_1	x_2	$f(x_1, x_2)$	
PSO	1.0005	1.0111	0.0011	40.819
GA	1.0796	1.1656	0.0063	10.325
MPS	1.0244	1.0448	0.0027	58.141
SA	0.9531	0.9085	0.0022	31.950
SQP	-	-	-	-
AURE	1.0476	1.1000	0.0029	10.232

5.2 Benchmark Test 2 - Banana Function

Rosenbrock's valley or Banana function is a classic test problem for optimization algorithms due to its challenge on the convergence and robustness of the algorithm. The global minimum is inside a long, narrow, parabolic shaped flat valley, and converges to the solution at point (1, 1) is well known to be difficult. The Banana function is given in Eq. 4, and illustrated in Fig. 5. The results are presented in Table 2.

$$f(x_1, x_2) = 100\left(x_2 - x_1^2\right)^2 + \left(1 - x_1\right)^2 \qquad (4)$$

5.3 Benchmark Test 3 - Beak Function

The Beak function is used as a benchmark problem due to it's the challenge to find its global minimum. The Beak function is given in Eq. 5, and illustrated in Fig. 6. The results are presented in Table 3.

$$f(x_1, x_2) = 3(1 - x_1)^2 e^{\left(-x_1^2 - (x_2+1)^2\right)} - 10\left(\frac{x_1}{5} - {x_1}^3 - x_1^5\right) e^{\left(-x_1^2 - x_1^2\right)} - \frac{1}{3} e^{\left((1(x_1+1)^2 - x_2^2\right)} \qquad (5)$$

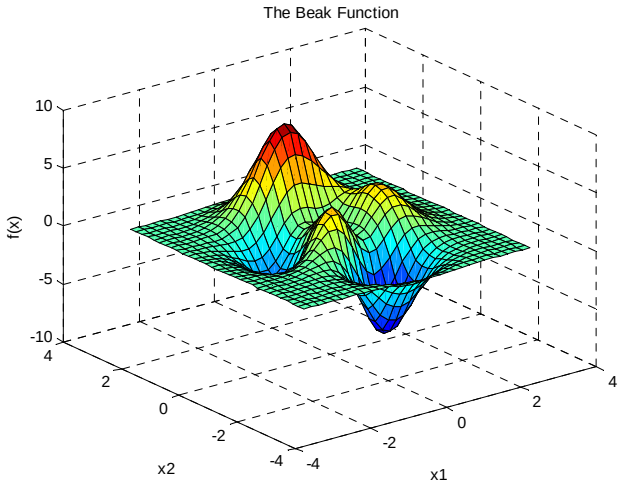

Figure 6: The Objective function of benchmark test 3

Table 3: Test Results on the Beak Function

Algorithm Used	Calculated Optimum, X^* and $f(X^*)$			CPU time (sec)
	x_1	x_2	$f(x_1, x_2)$	
PSO	0.2283	-1.6255	-6.5511	38.145
GA	0.2303	-1.6261	-6.5511	23.924
MPS	0.2398	-1.6140	-6.5486	16.626
SA	0.2283	-1.6255	-6.5511	19.378
SQP	-0.0000	-3.0000	-0.2500	0.1200
AURE	0.2467	-1.7863	-6.1928	9.5730

5.4 Benchmark Test 4 - Goldstein and Price Function (GP)

Goldstein and Price function is often used as a benchmark function due to the difficulty to find its global minimum; the function is given in Eq. 6, and illustrated in Fig. 7. The results are presented in Table 4.

$$f(x_1, x_2) = 1 + (x_1 + x_2 + 1)^2 (19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)(30 + 2x_1 - 3x_2)^2 (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2) \quad (6)$$

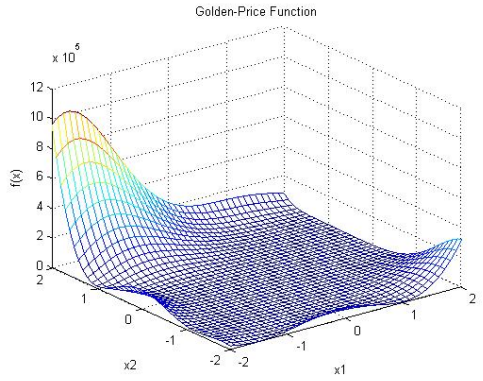

Figure 7: The Objective function of benchmark test 4

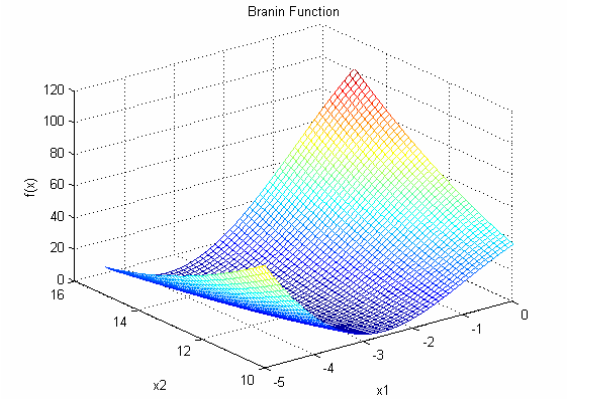

Figure 8: The Objective function of benchmark test 5

Table 4: Test Results on Goldstein and Price Function

Algorithm Used	Calculated Optimum, X^* and $f(X^*)$			CPU time (sec)
	x_1	x_2	$f(x_1, x_2)$	
PSO	-0.0003	-0.9999	3.0005	19.829
GA	0.0005	-0.9999	3.0001	11.837
MPS	0.0022	-0.9940	3.0010	43.500
SA	-0.0000	-1.0000	3.0000	19.630
SQP	-	-	-	-
AURE	-0.0007	-0.9997	3.0002	8.1560

5.5 Benchmark Test 5 - Branin function (BR)

Branin function is also used as a benchmark function due to the difficulty to find its global minimum; the function is given in Eq. 7, and illustrated in Fig. 8. The results are presented in Table 5.

$$f(x) = \left(x_2 - (5.1/4\pi^2)x_1^2 + (5/\pi)x_1 - 6\right)^2 + \left(10(1 - 1/8\pi)\cos x_1 + 10\right) \quad (7)$$

Table 5: Test Results on Branin Function

Algorithm Used	Calculated Optimum, X^* and $f(X^*)$			CPU time (sec)
	x_1	x_2	$f(x_1, x_2)$	
PSO	-3.1403	12.2790	0.3979	28.781
GA	-3.1418	12.2767	0.3979	15.112
MPS	-3.1444	12.0996	0.4311	14.669
SA	-3.1416	12.2750	0.3979	20.189
SQP	-3.1416	12.2750	0.3979	0.6600
AURE	-3.1040	12.2119	0.4053	14.300

5.6 Benchmark problem 6- Schaffer's Function (F6)

Schaffer's, or what is known as F6, function has been considered due to the challenge to converge to its global minimum. The function is given in Eq. 8, and illustrated in Fig. 9. The results are presented in Table 6.

$$f(x_1, x_2) = 0.5 + \frac{\sin\left(\sqrt{\left(x_1^2 + x_2^2\right)^2}\right)}{\left(1 + 0.01\left(x_1^2 + x_2^2\right)\right)^2} \quad (8)$$

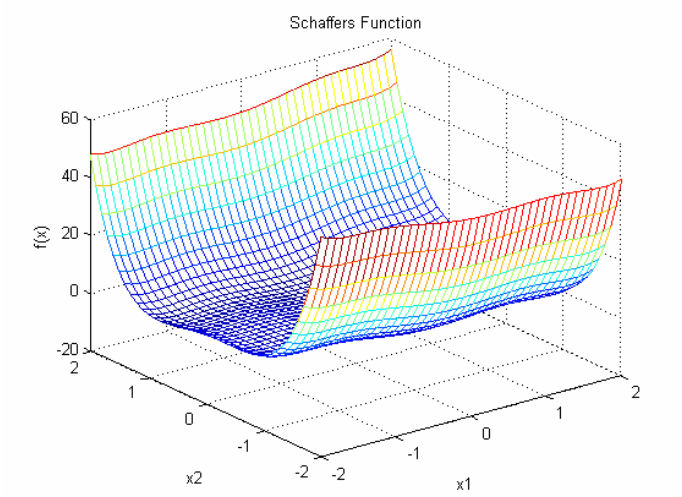

Figure 9: The Objective function of benchmark test 6

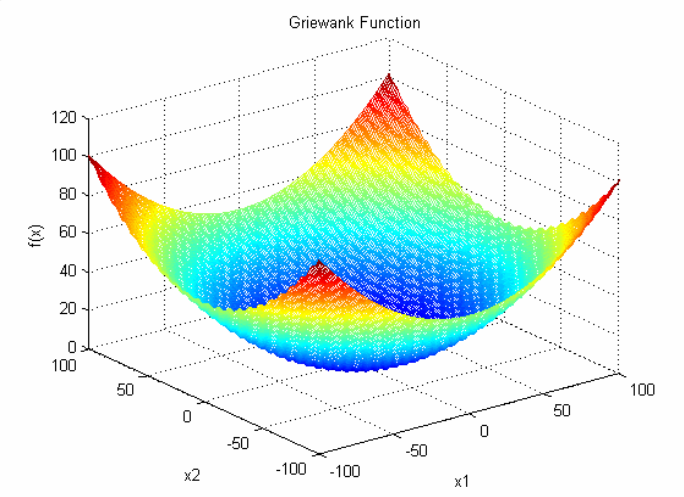

Figure 10: The Objective function of benchmark test 7

Table 6: Test Results on Schaffer's Function

Algorithm Used	Calculated Optimum, X^* and $f(X^*)$			CPU time (sec)
	x_1	x_2	$f(x_1, x_2)$	
PSO	-2.0000	-0.8296	-0.4122	20.149
GA	1.64062	1.41405	-0.4122	8.4920
MPS	0.0022	-0.0027	0.0000	9.9420
SA	-1.9024	-1.0364	-0.4122	16.704
SQP	0.0000	0.0000	0.0000	4.0600
AURE	2.1497	-0.2681	-0.4122	5.0220

Table 7: Test Results on Griewank Function

Algorithm Used	Calculated Optimum, X^* and $f(X^*)$			CPU time (sec)
	x_1	x_2	$f(x_1, x_2)$	
PSO	-0.0006	0.0014	0.0000	41.910
GA	0.0015	-0.0015	0.0000	21.451
MPS	-0.0146	0.0705	0.0014	30.766
SA	-0.0000	-0.0000	0.0000	26.668
SQP	---	---	---	---
AURE	-0.0009	0.0007	0.0000	12.562

5.7 Benchmark problem 7- Griewank function (GN)

Griewank function is another benchmark problem that we have tested in this work. Griewank function is given in Eq. 9, and illustrated in Fig. 10. The results are presented in Table 7.

$$f(x_1, x_2) = \frac{\left(x_1^2 + x_2^2\right)}{\left(200\text{-}\cos x_1\right)\left(\cos\left(x_2/\sqrt{2}\right)\right)+1} \qquad (9)$$

5.8 Benchmark problem 8- Generalized polynomial function (GF)

Generalized polynomial function is used in the test also due to the challenge of convergence to the global minimum. The Generalized polynomial function is given in Eq. 10, and illustrated in Fig. 11. The results are presented in Table 8.

$$f(x_1, x_2) = \left(1.5 - x_1\left(1 - x_2\right)\right)^2 + \left(2.25 - x_1\left(1 - x_2^2\right)\right)^2 \times \left(2.625 - x_1\left(1 - x_2^3\right)\right)^2 \qquad (10)$$

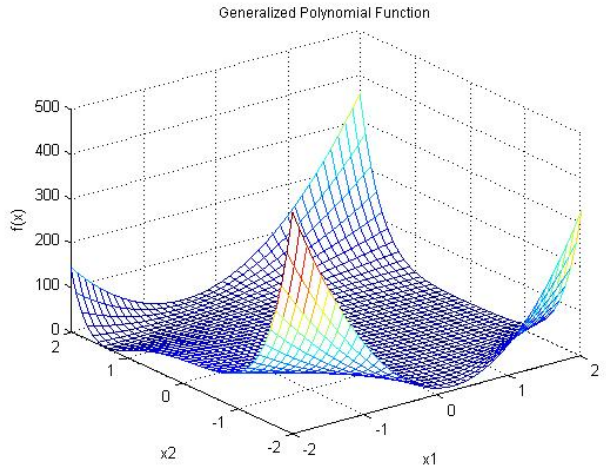

Figure 11: The Objective function of benchmark test 8

Table 8: Test Results on Generalized Polynomial Function

Algorithm Used	Calculated Optimum, X^* and $f(X^*)$			CPU time (sec)
	x_1	x_2	$f(x_1, x_2)$	
PSO	2.0000	0.1703	0.5233	13.990
GA	2.0000	0.1701	0.5233	18.967
MPS	2.0000	0.1702	0.5233	272.991
SA	2.0000	0.1701	0.5233	16.694
SQP	2.0000	0.1700	0.5300	3.1500
AURE	2.0000	0.1500	0.5126	9.3440

5.9 Summary of Test Results

The results from the performance tests carried out in this work have demonstrated that the newly introduced AURE method has many advantages. The algorithm is capable of solving the challenging benchmark problems in global optimization; locating the optimum with comparable accuracy; and obtaining the results with much reduced computation time.

The use of DOE and RSF, and the avoidance of redundant examination of searched areas all contributed to the improved performance. Historically, the benchmark problems commonly used, and adopted here, are all limited to two design variables for the ease of illustration. It is expected that the computation efficiency advantage of the AURE approach can be more significant with the increased number of design variables and/or more computation demanding objective functions. Specifically, the comparison studies showed that;

- Performance or convergence accuracy
  The new AURE method can locate the global minimum more accurately or reach comparable accuracy in most of the test cases (7 out of 8) and obtain slightly less accurate result in one of the 8 cases.
- Robustness
  The new AURE method is able to handle all challenge local/global optimization problems.
- Computational efficiency
  The CPU time needed by the new AURE method is only 50.44% of GA, 49.05% of SA, 35.90% of PSO, and 15.46% of MPS in average.

In Table 9, the relative strength of each competing algorithm is illustrated. The minimum objective function value reached by all algorithms in the global minimization and the computation (or CPU) times needed by the SA algorithm to solve the 8 benchmark problems are used as the references.

Table 9: Algorithm Performance Comparison

Method		GA	PSO	MPS	SA	SQP	AURE
Alpine Function	Obj. Fun. Value	-6.1295	-6.1291	-6.1277	-6.1295	0.0000	-6.1290
	Rel. Comp. Time	1.13	1.63	1.66	**1.00**	0.21	0.65
Banana Function	Obj. Fun. Value	0.0063	0.0011	0.0027	0.0022	---	0.0029
	Rel. Comp. Time	0.32	1.28	1.82	**1.00**	---	0.32
Beak Function	Obj. Fun. Value	-6.5511	-6.5511	-6.5486	-6.5511	-0.25	-6.1928
	Rel. Comp. Time	1.23	1.97	0.86	**1.00**	0.01	0.49
GP Function	Obj. Fun. Value	3.0001	3.0005	3.001	3.0000	---	3.0002
	Rel. Comp. Time	0.60	1.01	2.22	**1.00**	---	0.42
BR Function	Obj. Fun. Value	0.3979	0.3979	0.4311	0.3979	0.3979	0.4053
	Rel. Comp. Time	0.75	1.43	0.73	**1.00**	0.03	0.71
F6 Function	Obj. Fun. Value	-0.4122	-0.4122	0.0000	-0.4122	0.0000	-0.4122
	Rel. Comp. Time	0.51	1.21	0.56	**1.00**	0.24	0.30
GN Function	Obj. Fun. Value	0.0000	0.0000	0.0014	0.0000	---	0.0000
	Rel. Comp. Time	0.80	1.57	1.15	**1.00**	---	0.47
GF Function	Obj. Fun. Value	0.5233	0.5233	0.5233	0.5233	0.5300	0.5126
	Rel. Comp. Time	1.14	0.84	16.35	**1.00**	0.19	0.56

Note: Obj. Fun. Value: Objective Function Value; Rel. Comp. Time: Real Computation Time.

6 CONCLUSIONS

In this work, a new global optimization algorithm based on design experiments, region elimination and response surface model, namely Approximated Unimodal Region Elimination Method (AUREM), is introduced. The approach divides the field of interest into several unimodal regions using design experiment data; identify and rank the regions that most likely contain the global minimum; form a response surface model with additional design experiment data over the most promising region; identify its minimum, remove this processed region, and move to the next most promising region. By avoiding redundant searches, the approach identifies the global optimum with reduced number of objective function evaluations and computation effort. The new algorithm was tested using a variety of benchmark global optimization problems and compared with several widely used global optimization algorithms. The experiments results present robust performance, comparable search accuracy and superior computation efficiency, making the new algorithm an ideal tool for computer analysis and simulation black-box based global design optimization

REFERENCES

Barton, R. R., 1998, "Simulation Metamodels," Winter Simulation Conference.

Dorigo, M., 1992, *Optimization, Learning and Natural Algorithms*, Ph.D. Thesis, Politecnico di Milano, Italy.

Goldberg, D. E., 1989, *Genetic Algorithms in search, Optimization, and Machine Learning*, Addison- Wesley.

Holland, J. H., 1975, Adaptation in Natural and Artificial Systems, MIT Press.

Kaymaz, I., and McMathon C. A., 2005, "A Response Surface Method Based on Weighted Regression for Structural Reliability Analysis," *Probabilistic Engineering Mechanics*, 20, pp. 11-17.

Kennedy, J. and Eberhart R. C., 1995, "*Particle Swarm Optimization*," *Proceedings of IEEE Int. Conference on Neural Networks*, Perth, Australia, IEEE Service Center, Piscataway, NJ.

Kirkpatrick, S., Gelatt, C., D., and Vecchi, M., P., 1983, "Optimization by Simulated Annealing," *Science* 220, pp. 671-680.

Kuehl, R. O., 2000, *Design of Experiments: Statistical Properties of Research Design and Analysis.*

Montgomery, D. C., 2001, Design and Analysis of Experiments.

Myers, H. Raymond, 1971, *Response Surface Methodology*, Allyn and Bacon Inc., Boston, USA, pp. 26-30.

Pary, J., Bornoff, R. B., Stehouwer, P., Driessen, L. T, and Stinstra, E., 2004, "Simulation-Based Design Optimization Methodologies Applied to CFD," *IEEE Transactions on Components and Packaging Technologies*, **27**(2), pp. 391-397.

Renner, G. and Ekart, A., 2003, "Genetic Algorithms in Computer Aided Design," *Computer-Aided Design*, 35, pp. 709-726.

Simpson, T. W., Pepliniski, J. D. Koch. P. N., and Allen, J. K., 2001, "Metamodeling for Computer-Based Engineering Design: Survey and Recommendations," *Engineering with Computers*, **17**, pp. 129-150.

Tiernan, P., Draganescu, B., and Hillery, M. T., 2005, "Modeling of Extrusion Force Using the Surface Response Method," *International Journal of Advanced Manufacturing Technology*, 27, pp. 48-52.

Toksari, M. D., 2006, "Ant Colony Optimization for Finding the Global Minimum," *Applied Mathematics and Computation*, 176, pp. 308-316.

Wang, G., Dong, Z., and Aitchison, P., 2001, "Adaptive Response surface method-A Global Optimization Scheme for Approximation-Based Design Problems," *Journal of Mechanical Engineering*, 33, pp. 707-733.

Wang, L., Shan, S., and Wang, G., 2004, "Mode-Pursuing Sampling Method for Global Optimization on Expensive Black-box Functions," *Engineering Optimization*, 36(4), pp. 419-438.