International Journal of Engineering, Mathematical and Physical Sciences
ISSN: 2517-9934
Vol:3, No:2, 2009

A Comparative Study of Rigid and Modified Simplex Methods for Optimal Parameter Settings of ACO for Noisy Non-Linear Surfaces

Seksan Chunothaisawat, and Pongchanun Luangpaiboon

Abstract**—**There are two common types of operational research techniques, optimisation and metaheuristic methods. The latter may be defined as a sequential process that intelligently performs the exploration and exploitation adopted by natural intelligence and strong inspiration to form several iterative searches. An aim is to effectively determine near optimal solutions in a solution space. In this work, a type of metaheuristics called Ant Colonies Optimisation, ACO, inspired by a foraging behaviour of ants was adapted to find optimal solutions of eight non-linear continuous mathematical models. Under a consideration of a solution space in a specified region on each model, sub-solutions may contain global or multiple local optimum. Moreover, the algorithm has several common parameters; number of ants, moves, and iterations, which act as the algorithm's driver. A series of computational experiments for initialising parameters were conducted through methods of Rigid Simplex, RS, and Modified Simplex, MSM. Experimental results were analysed in terms of the best so far solutions, mean and standard deviation. Finally, they stated a recommendation of proper level settings of ACO parameters for all eight functions. These parameter settings can be applied as a guideline for future uses of ACO. This is to promote an ease of use of ACO in real industrial processes. It was found that the results obtained from MSM were pretty similar to those gained from RS. However, if these results with noise standard deviations of 1 and 3 are compared, MSM will reach optimal solutions more efficiently than RS, in terms of speed of convergence.

Keywords**—**Ant Colony Optimisation, Metaheuristics, Modified Simplex, Non-linear, Rigid Simplex.

I. Introduction

OPTIMISATION algorithms can be categorised as being either conventional or approximation optimisation algorithms [1]. Conventional optimisation algorithms are usually based upon mathematical procedures such as Integer Linear Programming, Branch and Bound or Dynamic Programming. These approaches were relatively well developed and attributed to the military services early in World War II. Based on the full enumerative search within these approaches, the optimal solutions are always guaranteed. However, the applications of these methods might need exponentially computational time in the worst cases. This becomes an impractical approach, especially for solving a very large size problem.

Seksan Chunothaisawat is with the Industrial Statistics and Operational Research Unit (ISO-RU), Department of Industrial Engineering, Faculty of Engineering, Thammasat University, 12120, Thailand (phone: (662)564-3002-9; fax: (662)564-3017; e-mail: uhooo@hotmail.com).

Pongchanun Luangpaiboon is an Associate Professor, ISO-RU, Department of Industrial Engineering, Faculty of Engineering, Thammasat University, 12120, Thailand.

This Research is sponsored by Lee Sae Import (1975) Ltd., Rungrojwattana Co., Ltd. and Sangfha Siam Industrial Co., Ltd.

Alternative approaches that can guide the search process to find near optimal solutions in acceptable computational time are therefore more practical and desirable. Approximation optimisation algorithms, called metaheuristics, have therefore received more attention in the last few decades. Metaheuristics iteratively conduct stochastic search processes inspired by natural intelligence. They can be categorised into three groups: physically-based inspiration such as Simulated Annealing [2]; socially-based inspiration for instance Taboo Search [3]; and biologically-based inspiration e.g. Ant Colony Optimisation [4], Artificial Immune System [5], Genetic Algorithm [6], Memetic Algorithm [7], Neural Network [8], Particle Swarm Optimisation [9], and Shuffled Frog Leaping [10]. These alternative approaches have been widely used to solve large-scale combinatorial optimisation problems [11]— [14].

Metaheuristics optimisation algorithms are considered to be more contemporary to solve larger-scale optimisation problems. The algorithms are mimicking natural intelligence to create algorithms and its processes of defining near optimal solutions, instead of using simple mathematic constraints to define an exact optimisation.

Metaheuristics are more complicated due to constraints of the algorithm itself not of the question. These constraints or their parameters are needed to be initialised to optimise the outcome of the solution, or in other word, constraints directly affect the quality of the solution. So it is in turn inspire an objective of this paper to examine the relation of constraints adjacent to the quality of solution of a chosen metaheuristic algorithm, Ant Colonies Optimisation (ACO), by two similar treatments; Rigid Simplex Method (RS) and Modified Simplex Method (MSM). Inspection and analysis are used to determine a recommendation on the proper levels of parameter settings for eight non-linear continuous mathematical models within three classes; unimodal, multimodal and curve ridge functions. Eight non-linear continuous mathematical models are considered being complicated optimisation problems when applied to real industrial processes.

This paper is organised as follows. Section 2 describes the selected metaheuristic; Ant Colonies Optimisation (ACO) and its pseudo code. Section 3 and 4 are briefing about proposed algorithms of Rigid Simplex and Modified Simplex, respectively. Section 5 presents eight tested problems, all of which are non-linear continuous mathematical functions. For each function, the optimal solution, the equation, considered ranges and its surface plot are provided. Section 6 presents design and analysis of computational experiments for comparing the performance of the proposed methods. The conclusion is also summarised and it is followed by acknowledgment and references.

II. Ant Colony Optimisation Algorithm (ACO)

Ant algorithm was first proposed by Dorigo and his colleagues [4] as a multi-agent approach to optimisation problems, such as a travelling salesman problem (TSP) and a quadratic assignment problem (QAP). There is currently a lot of ongoing activity in the scientific community to extend or apply ant-based algorithms to many different discrete optimisation problems. Recent applications cover problems like a vehicle routing, a plant layout and so on. Ant algorithm is inspired by observations of real ant colonies. Ants are social insects and they live in colonies. Behaviour is direct more to the survival of the colony as a whole than to that of a single individual component of the colony. Social insects have captured the attention from many scientists because of a structure of their colonies, especially when compared with a relative simplicity of the colony's individual. An important and interesting behaviour of ant colonies is their foraging behaviour and in particular how ants can find shortest paths between food sources and their nest [9], [15].

While walking from food sources to the nest and vice versa, ants deposit on the ground a substance called pheromone, forming a pheromone trail. With ants ability to smell pheromone they tend to choose a path marked by strong pheromone concentrations with the higher probability. The pheromone trail allows the ants to find their way back to the food source and vice versa. It can be also used by other ants to find the location of the food sources that found by their nest mates [10]. Generally, Ant Colony Optimisation algorithm consists of the iteration steps where each ant makes its own solution as follows;

1. Define parameters for Ant Colony Optimisation Algorithm, such as number of ants, moves, iterations and etc.
2. Each ant makes its own initial states (s), paths and communicate the responses (or yields) and coordinates where
   - Construct the feasible solution.
   - Evaluate the generated solution.
   - Decide to retrace the path that the ant has followed.
3. Random '*k*' variables for initial states (s) of each ant which turn on the ant activities and compare its responses and termination criteria.
4. From initial state (s), ant activities drive all ants in system and move to its neighbourhood state: r, s_r.
5. While each ant locates at neighbourhood states: r, s_r, a system compares the responses and its initial states (s). If any response of the same ant is better than its initial states (s), then move to a neighbourhood state: n, s_n.
6. In case of neighbourhood states: n, s_n less than the previous state, the system generates a probability number (q_1) and compare with a certain number (q_0). If q_1 is greater than q_0, a movement of each ant is going ahead. Otherwise, there is no movement.
7. In case of no better neighbourhood response, set this state as 'Local Optima' (L_i) and wait for a communication from other ants at other 'Local Optima' (L_j).
8. Compare among 'Local Optima' (L_1, L_2, …, L_i, L_j , …, L_n) and set a direction of the path to the best Local Optima.
9. Construct the solution by repeating steps 4-9, until the termination conditions are met.

The pseudo code is used to briefly explain to all the procedures of ACO shown in Fig. 1.

Procedure ***ACO Metaheuristic()***
While *(termination criterion not satisfied) – (line 1)*
  Schedule activities
    ants generation and starting point;
    makes path or step for each ant
    compare cost or response function
    if *no improvement of cost function* **then**
      communication with best ant cost function
      make path or step from local trap to best ant
    else
      if *ant found the better cost function* **then**
        go to *line 5.*
      else
        wait for best ant communication
      end if
    end if
  end schedule activities
end while
end procedure

Fig. 1 Pseudo Code of ACO Metaheuristic

As shown above, nature has always been a source of inspiration. Various types of nature-inspired algorithms have been developed during the last few decades. These algorithms iteratively conduct stochastic search processes adopted from natural intelligence. However, these metaheuristics seem to get the same problem of having sensitive parameters affecting the quality of solutions. In this work a nature-inspired algorithm called Ant Colony Algorithm (ACO) were proposed to review and give an aid on complicatedness of the proper levels of parameter settings via Rigid Simplex (RS) and Modified Simplex Methods (MSM).

III. The Basic Simplex /Rigid Simplex Method (RS)

The rigid simplex method (RS) has been first proposed by Spendley et al. [16]. The basic shape (design) is called the simplex. The simplex design in a problem with *k* variables consists of *k*+1 design points (vertices) but it is not necessary to have a property of equidistance. For *k* equal to two, this simplex is a triangle, for *k* equal to three it is a tetrahedron (Fig. 2).

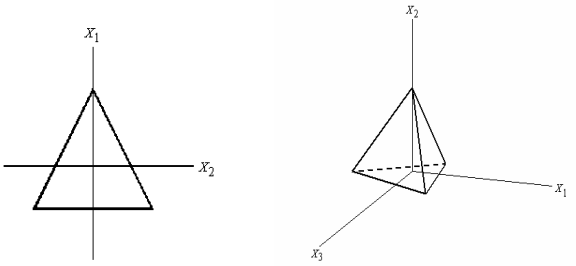


Fig. 2 Simplex Designs for *k* = 2 and 3.

The simplex design is first applied at arbitrary points within the safe region of operation. In practice this might be the current operating conditions or the centre of the safe operating region. Response function is computed for each of the *k*+1 design points. The vertex corresponding to the design point with the lowest yield, i.e. the vertex, W, is identified and reflected in the opposite hyper-face. The next computation is carried out with variables set at values corresponding to a new point, R, that is the reflection of W

along a line joining W to the centroid ($\overline{P}$) of the other points in the simplex. Thus

$$R = \overline{P} + (\overline{P} - W)$$

Where,

W = the vertex corresponding to the design point with the lowest yield

$\overline{P}$ = the centroid of the other points in the simplex.

- For variable $k = 2$ (Fig. 3);

$$M = \overline{P} = \frac{B+G}{2}$$

$$R = M + (M - W) = 2M - W$$

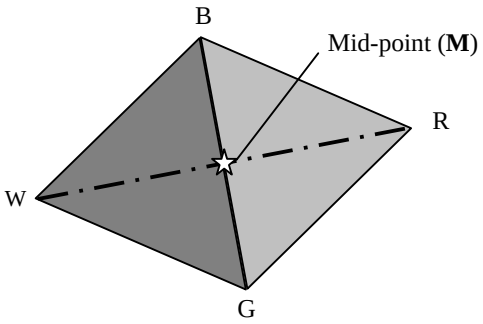


Fig. 3 Simplex Move for $k = 2$

- For Variable $k = 3$ (Fig. 4);

$$\frac{B+C}{2} = M,$$

$$\overline{P} = M + \frac{(D-M)}{2},$$

$$R = \overline{P} + (\overline{P} - W)$$

Where;

W = A, and R is the opposite vertex of A having BCD plane as a plane of symmetry.

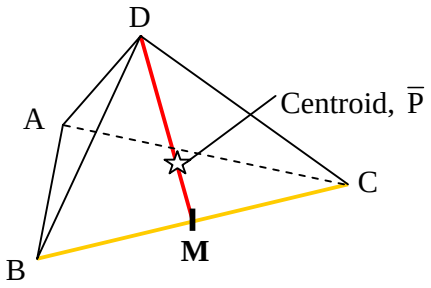


Fig. 4 Simplex Design and its Centroid for $k = 3$

The algorithm continues in this fashion. However, it is possible that the new design point leads to the least yield of the new simplex. A reversed reflection will be occurred in order to increase a chance of finding more favourable yield. If there is an oscillation the process will be stopped according to one of the preset stopping criteria, appeared in the pseudo code (Fig. 5). And the only best so far value will be counted for further conclusion. In order to avoid rotating about a spurious high yield, there's a probability of choosing an acceptable small size of the initial simplex design. This would indeed solve the problem more effectively but would consume more execution time. In the former case the finishing strategy is then applied. An idea of RS's operation is shown in Fig. 6.

The Rigid Simplex (RS) consists of a few basic rules:

- The first rule is to *reject* the *least favourable response* value in the current simplex in order to improve the trial towards the optimisation value.
- The second rule is never to return to control variable levels that have just been rejected.
- In which case the second lowest yield of the original simplex is rejected in an attempt to prevent the algorithm oscillating.

Besides three main rules, two more rules are also considered.

- Trials (vertices) may be reevaluated in order to avoid a probability of wondering around spurious optimum.
- Calculated trials outside the effective boundaries of the controllable variables are not applicable. Instead a very unfavourable response is applied, forcing the simplex to move away from the boundary.

However, these rules can be adjusted depending on prospective users' specifications and satisfaction. Lastly termination criteria should be set in order to run the process within the interested operating regions and its lower or upper limit. Preset termination criteria could be in logics of or/and, and should be analysed and evaluated to suit a particular practice.

Procedure *of RS ()*
While *(termination criterion not satisfied) – (line 1)*
  Schedule activities *(for maximisation)*
    Reflection of least yield W is processed
    Compute R and f(R)
    Compare cost or response function
    if *f(R) is the least* **then**
      reflect backward to prior point W
      recalculate f(W)
    else
      reflect the least cost function vertex
      if *R and f(R) continue to be the least* **then**
        reflect new least cost function's vertex
      end if
    end if
  end schedule activities
end while
end procedure

Fig. 5 Pseudo Code of RS

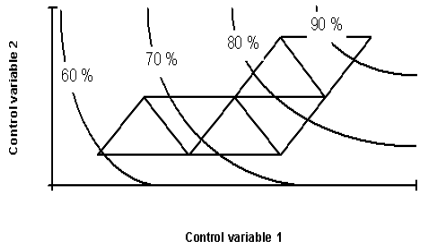


Fig. 6 An Example of a Typical Optimisation Sequences with the Rigid Simplex. Change in the Levels for Two Controllable Variables with the Response Marked as Contours

IV. Modified Simplex Method (MSM)

There are many extensions on the rigid simplex algorithm. One of the well-known is a modified simplex method (MSM) of Nelder and Mead [17]. In the MSM an expansion or contraction of the reflection is allowed at each step. Although there are many possible stopping criterions for simplex algorithms, this study follows Nelder and Mead and includes the standard deviation of the estimated yields at the vertices of the simplex. Various stopping rules and one based on the sample range were also tried on the literatures, but they appeared to offer no advantage over the stopping rule based on the standard deviation of process yields.

This work incorporated the MSM into the same manner of the first algorithm based on the RS. As before, the simplex design is first applied at an arbitrary point within the safe region of operation. The response is measured for each of the design points. In a maximisation process with three variables or a tetrahedron simplex, the vertex corresponding to the lowest yield (W) is identified and reflected in the opposite hyper-face to obtain (R) via the centroid ($\overline{P}$). The centroid obtained by other vertices in the simplex consists of V_H, V_S, and V_{SH}, or vertices of highest yield, second least yield and second highest yield, respectively. The new design point can be extended (E) in the direction of more favourable conditions, contracted (C- or C+) if a move is taken for least favourable conditions, and Shrunk toward best vertex if a contracted vertex is still the least but not less than the rejected trial condition (Fig. 7). The next run is carried out with variables set at values corresponding to this new design point. This MSM terminates, and the finishing strategy is applied. An idea of MSM's logical decision is shown in Fig. 8.

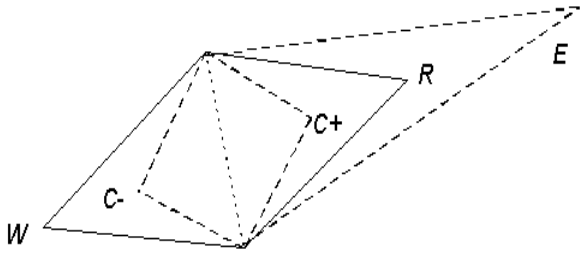


Fig. 7 Different Simplex Moves from the Rejected Trial Condition (W). R = Reflection, E = Expansion, C+ = Positive Contraction and C- = Negative contraction

Procedure ***of MSM ()***

While *(termination criterion not satisfied) – (line 1)*

Schedule activities
Reflection of least yield W is processed
Compute R and f(R)
Compare cost or response function
if *f(R) is highest* **then**
extension E will be processed
else
if *R and f(R) continue to be the least* **then**
reflect backward to prior point
recalculate W and f(W)
or
contraction C or shrinking S will be processed
recalculate f(C) or f(S)
else
go to *line 3.*
end if
end if
end schedule activities
end while
end procedure

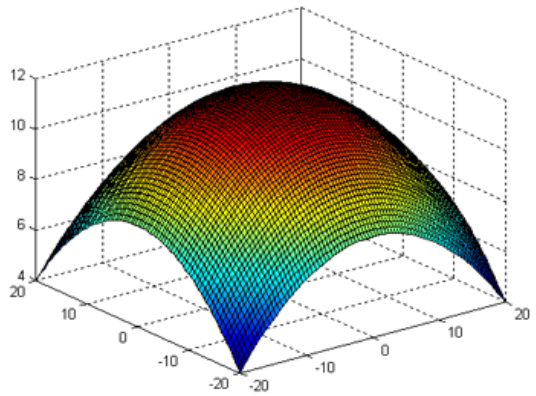
Fig. 8 Pseudo Code of MSM

V. Tested Functions

In this paper, eight non-linear continuous mathematical functions were used to test the performance of the proposed methods for searching the optimal solutions under a consideration of parameters adjacent to RS and MSM. The functions including the equations and its surface plot with ranges of $-20<x_1<20$ and $-20<x_2<20$ are illustrated in the following subsections. However, for both Rastrigin and Styblinski surfaces will be plotted within ranges of $-5< x_1<5$; $-5< x_2<5$ to clearly illustrate the texture and characteristic of the surfaces**.**

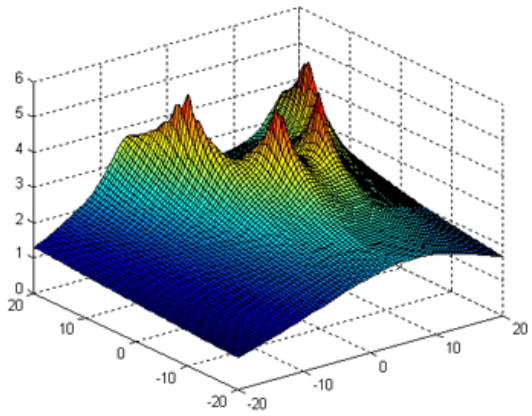
5.1 Parabolic Function

$$f(x_1, x_2) = 12 - \frac{x_1^2 + x_2^2}{100}$$



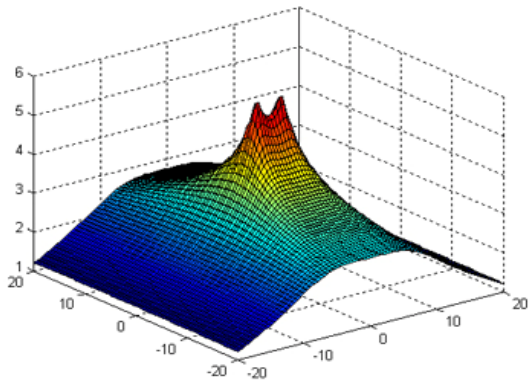
5.2 Branin Function

$$f(x_1,x_2) = 5 - \log_{10}[(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6)^2 + ((10 - \frac{5}{4\pi})\cos(x_1)) + 10]$$



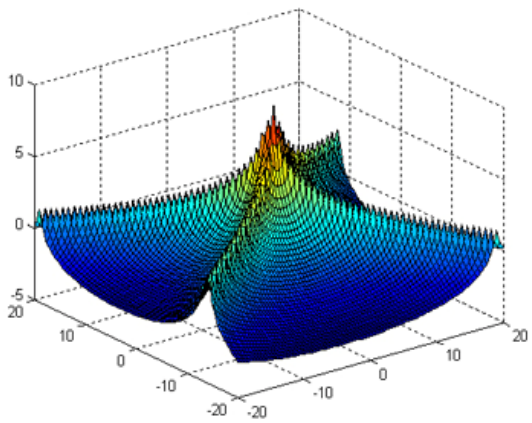
5.3 Camelback Function

$$f(x_1,x_2) = 10 - \log_{10}[x_1^2(4 - 2.1x_1^2 + \frac{1}{3}x_1^4) + x_1x_2 + 4x_2^2(x_2^2 - 1)]$$



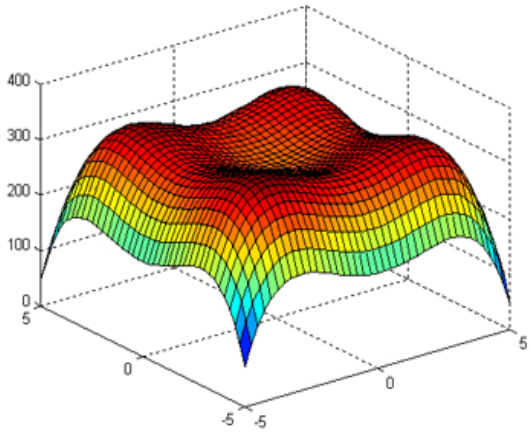
5.4 Goldstein-Price Function

$$f(x_1,x_2) = 10 + \log_{10}[1/\{(1+(1+x_1+x_2)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)) * (30 + (2x_1 - 3x_2)^2 (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2))\}]$$



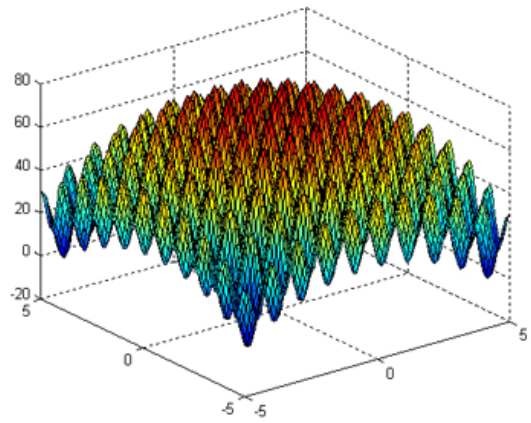
5.5 Shubert Function

$$f(x_1,x_2) = 275 - [(\frac{(x_1^4 - 16x_1^2 + 5x_1)}{2}) + (\frac{(x_2^4 - 16x_2^2 + 5x_2)}{2}) + 3]$$



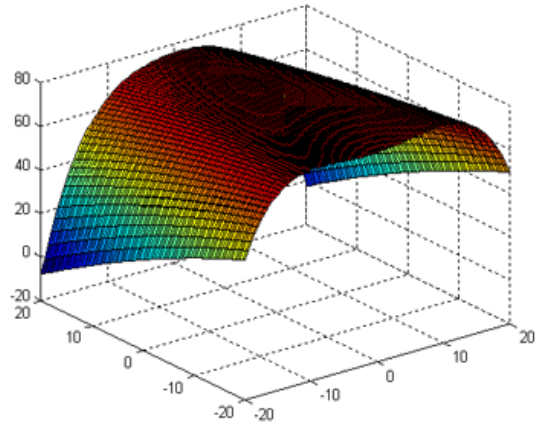
5.6 Rastrigin Function

$$f(x_1,x_2) = 80 - [20 + x_1^2 + x_2^2 - 10(\cos(2\pi x_1) + \cos(2\pi x_2))]$$



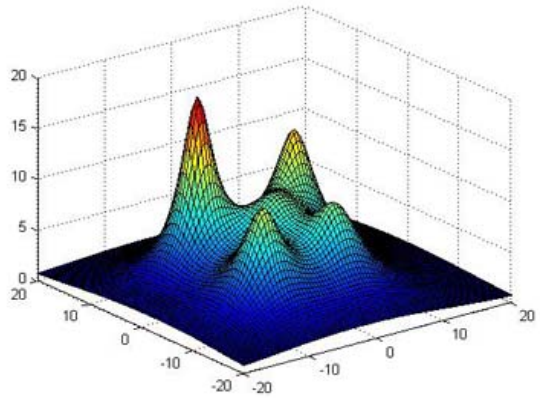
5.7 Rosenbrock Function

$$f(x_1,x_2) = 70\ [\frac{[20 - \{(\frac{1-x_1}{-7})^2 + (\frac{x_2}{6} + (\frac{x_1}{-7})^2)^2\}] + 150}{170}] + 10$$



5.8 Shekel Function

$$f(x_1, x_2) = 100\ [\frac{1}{9 + (x_1 - 4)^2 + (x_2 - 6)^2} + \frac{1}{20 + (x_1 - 0)^2 + (x_2 - 0)^2} + \frac{1}{14 + (x_1 + 8)^2 + (x_2 - 3)^2} + \frac{1}{11 + (x_1 + 8)^2 + (x_2 + 8)^2} + \frac{1}{6 + (x_1 - 6)^2 + (x_2 + 7)^2}$$



VI. Experimental Design and Analysis

In this work, a computer simulation program was developed using Matlab program 2006v.7.3B and EVOPtimiser program v.1.1.0. A desktop computer with IntelQ6600, RAM DDR2 4 GB and Geforce 9800GT was used for computational experiments. Eight non-linear continuous mathematical functions were used to test the performance of the proposed methods. For each function, the computational runs were repeated until it reaches the preset termination criteria. However, it has been stated that ACO's parameters have to be merely positive and integer, as a result it would obtain a quicker stop or face with some round-off errors. It might make the process stop faster than what it should. As a result, the development of ACO results can't be clearly seen or in other words, a termination of the algorithm may be prematurely occurred. In this paper, the stopping criteria, categorised by RS and MSM, are followed.

For RS

- Dispersion rule - when a standard deviation (SD) of the yields of a simplex's vertices is less than a preset value of 0.7 (obtained from several pilot trials), and
- Replication rule - *Opposing to the Second rule of RS*, if the yield of a new calculated reflection brings the least favourable result, a backward reflection is possible in order to increase chances of finding more favourable outcome or yield. However, a repeated path of reflection number should be set. The procedure would be terminated, after three repeated path of reflection. Maximal yield produced so far would be considered as the performance measures of the algorithms, or
- Parameter default rule - when the coordinates escape from the first quadrant or the upper or lower limit, or
- Adjacent rule - when each vertex of a simplex approximately brings the same level of the yields.

For MSM

- Size rule - when the size of simplex is as small as 2% of the ranges in the solution space. In this case ACO parameters have to be integer, as a result a premature stopping of the process may occur upon an oscillation rule, or
- Oscillation rule - when round-up coordinates give same numbers for three times or replicates, or
- Dispersion rule - when a standard deviation (SD) of the yields of a simplex's vertices is less than 0.7, or
- Parameter default rule - when the coordinates escape the first quadrant or of the upper or lower limit, or
- Replication rule - when yields of processed vertices approximately repeat at the same result for four times.

Using different random seed numbers, experimental results obtained from each method including best-so-far (BSF or Y) solutions and its error percentages (as shown in Table I) were compared to the optimal solutions of all eight tested functions described in the previous section. It should be noted that the achieved solutions of Camelback function cannot be found since the function does vary around optimal coordinates. Moreover, it is harder to identify solutions of a curve-ridge, Rosenbrock, function (the hardest function), due to an occurrence of "zigzag" effect circulating the maximal path (Section 5, 5.7). On a multimodal function (moderately hard function), there is probability that the search would find a local optimum rather than the global optimum and this lead to using more execution time to terminate processes, such as, Rastrigin function (Section 5, 5.6). In addition, if computational process exceeds upper or lower limit, the super modified simplex would be applied [18].

From Table I, it can be seen that both Rigid Simplex (RS) and Modified Simplex Methods (MSM) found the optimal solutions at about the same rate when applied without noise. MSM are more efficient for some surfaces. The algorithms also operate and analyse the results under levels of noise (N) standard deviation of one (Table II) and three (Table III). When levels of noises increase in the system, computational time is also taken longer due to complexities of ACO algorithm (ant activities and communications) for checking 'Local optima' [19]. Moreover, more complicate function is let to higher rate of resource consumptions; number of ants, moves and iterations (Table III).

From Table IV, preferable levels of parameters found by RS and MSM are determined and are set to be suggested levels for ACO's parameters, to promote an ease of use in every kind of equation. Under a consideration of recommended levels of its parameters, those may bring the benefit to solve industrial processes via ACO when the nature of the problems can be categorised as unimodal, multimodal or curve ridge.

TABLE I
EXPERIMENTAL RESULTS OBTAINED FROM THE PROPOSED METHODS ON EACH TESTED FUNCTION WITHOUT NOISE

No.	Function Name	% Difference of (BSF) MSM - RS	Modified Simplex Method (MSM)				Rigid Simplex (RS)			
			BSF and Round-up Parameters				BSF and Round-up Parameters			
			Y	Iterations	Ants	Moves	Y	Iterations	Ants	Moves
1	Branin	0.000017	5.92158	6	12	9	5.92158	6	38	10
2	Camelback	58.28752	72.52808	8	13	13	30.25326	4	8	9
3	Goldstein-Price	0.00001	8.90138	9	14	19	8.90138	9	7	15
4	Parabolic	0	12.00000	1	7	9	12.00000	1	2	10
5	Rastrigin	0	100.00000	4	9	24	100.00000	7	7	16
6	Rosenbrock	0	80.00000	1	13	13	80.00000	16	16	21
7	Shekel	0.000016	18.98052	1	15	12	18.98051	6	5	9
8	Styblinski	0.000009	353.33230	10	8	18	353.33233	16	10	12

TABLE II
EXPERIMENTAL RESULTS OBTAINED FROM THE PROPOSED METHODS ON EACH TESTED FUNCTION WITH NOISE STANDARD DEVIATION OF 1

No.	Function Name	% Difference of (BSF) MSM - RS	Modified Simplex Method (MSM)				Rigid Simplex (RS)			
			BSF and Round-up Parameters				BSF and Round-up Parameters			
			Y	Iterations	Ants	Moves	Y	Iterations	Ants	Moves
1	Branin	30.39	9.578468	7	20	11	9.361398	7	7	12
2	Camelback	85.48	66.594600	13	11	13	32.618253	10	11	14
3	Goldstein-Price	29.86	11.4073	11	4	9	12.091057	8	5	14
4	Parabolic	65.39	15.62516	8	8	10	15.860091	6	11	15
5	Rastrigin	89.35	103.3521	20	6	11	103.648839	11	5	11
6	Rosenbrock	89.93	91.59732	13	16	12	83.637236	10	8	11
7	Shekel	70.23	23.51877	10	11	23	22.805576	7	10	13
8	Styblinski	97.47	356.7795	9	5	12	356.563572	9	9	12

TABLE III
EXPERIMENTAL RESULTS OBTAINED FROM THE PROPOSED METHODS ON EACH TESTED FUNCTION WITH NOISE STANDARD DEVIATION OF 3

No.	Function Name	% Difference of (BSF) MSM - RS	Modified Simplex Method (MSM)				Rigid Simplex (RS)			
			BSF and Round-up Parameters				BSF and Round-up Parameters			
			Y	Iterations	Ants	Moves	Y	Iterations	Ants	Moves
1	Branin	17.98	16.93000	10	6	13	13.88597	8	11	9
2	Camelback	-13.24	23.78742	5	7	9	26.93891	13	15	10
3	Goldstein-Price	4.82	19.03004	7	15	11	18.11240	10	9	11
4	Parabolic	3.57	22.84964	8	9	9	22.03162	13	5	12
5	Rastrigin	-2.28	108.0246	5	5	7	110.49653	14	13	10
6	Rosenbrock	-0.91	90.14356	4	5	9	90.96466	11	11	14
7	Shekel	-0.19	27.30065	5	8	10	27.35385	11	4	8
8	Styblinski	-0.21	363.5414	10	10	15	364.32332	10	4	11

RECOMMENDED LEVELS OF PARAMETER SETTINGS WITHOUT NOISE (N=0) AND WITH NOISE (N=*x*)

Function No.	Function Name	Recommended Levels of Parameters		MSM	RS
1	Parabolic	N=0	(1,2,10)*		✓
		N=*x*	(8,9,9)	✓	
2	Branin	N=0	(6,12,9)	✓	
		N=*x*	(10,6,13)	✓	
3	Camelback	N=0	(8,13,13)	✓	
		N=*x*	(13,11,13)	✓	
4	Goldstein-Price	N=0	(9,14,19)	✓	
		N=*x*	(7,15,11)	✓	
5	Rastrigin	N=0	(7,7,16)		✓
		N=*x*	(14,13,10)		✓
6	Styblinski	N=0	(16,10,12)		✓
		N=*x*	(10,4,11)		✓
7	Shekel	N=0	(1,15,12)	✓	
		N=*x*	(11,4,8)		✓
8	Rosenbrock	N=0	(1,13,13)	✓	
		N=*x*	(13,16,12)	✓	

Note: (a,b,c)*: a = Iterations, b = Ants, c = Moves

ACKNOWLEDGMENT

The authors wish to thank Jate Ratanaphanyarat and Nattapon Rungruengsattaya on the early phase of this research.

REFERENCES

[1] C. Blum and A. Roli, "Metaheuristics in Combinatorial Optimisation: Overview and Conceptual Comparison". ACM Computing surveys, vol. 35, no. 3, pp. 268-308, 2003.
[2] S. Kirkpatrick, C.D. Gelatt and M.P. Vecchi, "Optimisation by Simulated Annealing". Science, vol. 220, no. 4598, pp. 671-679, 1983.
[3] F. Glover, "Tabu Search - part i". ORSA Journal on Computing, vol. 1, no. 3, pp. 190-206, 1986.
[4] M. Dorigo and T. Stutzle, Ant Colony Ooptimisation. Massachusetts, Bradford Book, 2004.
[5] E.A. Hart and J. Timmis, "Application Areas of AIS: The Past, the Present and the Future". Applied Soft Computing, vol. 8, no. 1, pp. 191-201, 2008.
[6] D.E. Goldberg, Genetic Algorithms in Search, Optimisation and machine learning. Massachusetts, Addison-Wesley, 1989.
[7] P. Merz and B. Freisleben, 1999. "A Comparison of Memetic Algorithms, Tabu Search, and Ant Colonies for the Quadratic Assignment Problem", presented at the 1999 Congress on Evolutionary Computation.
[8] S. Haykin, Neural Networks: A Comprehensive Foundation (2nd ed). NJ: Prentice Hall, 1999.
[9] J. Kennedy and R.C. Eberhart, Swarm Intelligence. San Francisco, CA: Morgan Kaufmann Publishers, 2001.
[10] M. Eusuff, K. Lansey and F. Pasha, "Shuffled Frog-Leaping Algorithm: A Memetic Metaheuristic for Discrete Optimisation". Engineering Optimisation, vol. 38, no. 2, pp. 129-154, 2006.
[11] H. Aytug, M. Knouja and F.E. Vergara, "Use of Genetic Algorithms to Solve Production and Operations Management Problems: A Review". International Journal of Production Research, vol.41, no. 17, pp. 3955–4009, 2003.
[12] S.S. Chaudhry and W. Luo, "Application of Genetic Algorithms in Production and Operations Management: A Review". International Journal of Production Research, vol.43, no. 19, pp. 4083-4101, 2005.
[13] D. Dasgupta, Artificial Immune Systems and their Applications. Springer-Verlag, 1998.
[14] M. Dorigo and C. Blum, "Ant Colony Optimisation Theory: A survey". Theoretical Computer Science, vol. 344, no.2-3, pp. 243-278, 2005.
[15] P. Ittipong, P. Luangpaiboon and P. Pongcharoen, 2008. "Solving Non-Linear Continuous Mathematical Models Using Shuffled Frog Leaping and Memetic Algorithm". Presented at the 2008 Operations Research Network Conference, Bangkok, Thailand.
[16] W.G.R. Spendley, G.R. Hext, and F.R. Himsworth, "Sequential Application of Simplex Designs in Optimisation and Evolutionary Operation". Technometrics, vol. 4, no. 4, pp. 441-461, 1962.
[17] J.A. Nelder and R. Mead, "A Simplex Method for Function Optimisation", Computer Journal, vol. 7, pp. 308-313, 1965.
[18] J.A.M. Pulgarin, A.A. Molina and M.T. Alanon Pardo, "The Use of Modified Simplex Method to Optimise the Room Temperature Phosphorescence Variables in the Determination of Antihypertensive Drug", Talanta, vol. 57, pp. 795-805, 2002.
[19] F.H. Walters, L.R. Parker, S.L. Jr.,Morgan and S.M. Deming,. Sequential Simplex Optimisation. CRC Press, Inc., Boca Raton, 1991.