Engineered Science
DOI: https://dx.doi.org/10.30919/es1140

Implementation of Covariance Function to Improve Ant Colony Algorithm for Common Chemical Engineering Optimization

Orn-anong Winyutrakoon, Saharat Rattanapairom, Pichita Petpraphan, and Thongchai Rohitadisha Srinophakun*

Abstract

An improved real-value ant colony optimization algorithm ($IACO_R$) focused on a weight associated with the solution by applying the covariance function to dominate the probability of selection. The performance of the developed algorithm was validated using 9 common engineering benchmarks, followed by 19 classical benchmarks. Furthermore, $IACO_R$ was introduced to typical chemical process simulations and developed to work seamlessly with a process simulator (ASPEN), as illustrated in 2 additional examples of distillation column and heat exchanger network. $IACO_R$ produced better optima and decreased the computational time with a lower number of iterations to solve complicated problems compared to the conventional method. The highest error compared to some available exact solutions are not over 1 % at low dimension of decision variables.

Keywords: Metaheuristic optimization; Covariance function; Improved real-value ant colony algorithm; Process optimization.
Received: 29 September 2023; Revised: 20 February 2024; Accepted: 29 February 2024.
Article type: Research article.

1. Introduction

There are alternative optimum solutions with complex economic and performance interactions in chemical engineering applications, especially in engineering design. The best decision-making is more than essential but sufficient.[1,2] Metaheuristic optimization has been used to model ant behavior in finding the optimal route from the nest to the food source by releasing a communicative chemical called a pheromone to mark the path for other ants to follow.

The developed real-value ant colony optimization algorithm (ACOR) is one of these optimization technics. Ants are social insects and live in colonies. Ants explore the area near their nest randomly for food. While searching, ants deposit an organic compound called pheromone on the ground as guidance to a food source. This pheromone trail will indirectly communicate with others in the colony to follow the path. Both food quantity and quality influence the amount of pheromones, with the higher the pheromone level, the higher the probability path will be chosen. The more ants follow the path, the more pheromone will be deposited. As the pheromone evaporates, this reduces the path's attraction over time. Similarly, frequently used routes will have a higher concentration of the pheromone trail and remain attractive. This could indicate a shorter route between the nest and the food source, providing a short cycle time for the ants. Of course, such routes accumulate a higher pheromone concentration. Accordingly, more ants are attracted to the shorter paths.[3–5] There are many ways to improve search algorithms. The adjustable crossover for biography-based optimization and the balanced pattern for Krill Herd have been successfully implemented in common engineering problems.[6,7] Using an extended random number during searching has been proposed in the classical genetic algorithm and implemented in various engineering problems.[8,9] Hybrid codes can be embedded as a mixed particle swarm with a sequential quadratic algorithm and an atom search for multi-objective optimization.[10-12]

Department of Chemical Engineering, Faculty of Engineering, Kasetsart University, Bangkok 10900, Thailand.
**Email:* fengtcs@ku.ac.th (T. R. Srinophakun)

2. Improvement of ant colony algorithm

A weighted solution can improve performance in finding the optimum path. This paper proposed a function to improve the solution by considering a statistical approach using the covariance function, as shown in equation 1:[13]

$$k(x_i, x_j) = \sigma^2 \exp\left(-\frac{(x_i - x_j)^2}{2l^2}\right) + \delta_{ij}\sigma^2_{noise} \quad (1)$$

where $\sigma^2>0$ is the signal variance, $l>0$ is the length scale, and $\sigma^2_{noise} \geq 0$ is the noise covariance only when i=j. A small length scale value means that function values can change quickly. On the other hand, large values characterize functions that change only slowly. In addition, the length scale determines the extrapolation of the training data. The signal variance σ^2 can scale the variation in function values from their meaning. A small value of σ^2 characterizes functions that stay close to their mean value, while larger values allow greater variation. The noise variance σ^2 noise is formally not a part of the covariance function itself. The Gaussian process model embeds this function to allow for noise present in training data. This can be formulated and gives more possibility functions regarding the conservation used in the ant algorithm, as shown in Fig. 1.

The $IACO_R$ has the following steps. The first step initializes the necessary parameters, such as the number of iterations, number of solutions or population, and algorithm (q, random value). Then, the initial x value is randomized. Next, we calculate an objective function, the weighted initial solution, and each ant's probability. For the constraint problem case, we ensure all the random values are still within the constraints. If the ant values from the random method conflict with the constraint, we must repeat the initialization step until the ant values satisfy the constraints. After that, the standard deviation of the ant is calculated to create the new ant in collaboration with improvement associated with the weighted solution and probability. However, first, the new ants created by the random value have to be checked in the domain. Then, all new ants will be updated and replaced with the old ants to improve the objective function value. Eventually, we check the convergence of the solution and update that until the solution is satisfactory.[13,14]

The improved real-value ant colony optimization algorithm ($IACO_R$) and the step-by-step procedure for the ant colony optimization algorithm are described in Fig. 2.

3. Results and discussion

This section demonstrates the performance of the improved algorithm ($IACO_R$) in adjusting the appropriate values based on various experiments regarding benchmark and design problems in chemical engineering. We aimed to modify an optimal solution and to compare the performance of the standard ACO_R with $IACO_R$. Both optimization methods used the same personal computer, with the same problems and associated factors.

Based on known optimization problems, 7 benchmarks and 4 chemical engineering problems were used to evaluate the algorithm's efficiency. The method name, function detail, and characteristic of these problems are summarized in the appendix. The letter "B" stands for benchmark, and "C" stands for chemical engineering, followed by the problem number. For example, Fig. 3 shows the comparison between ACO_R and $IACO_R$; for example, benchmark function B7 with full convergence behavior. This figure represents the best optimum value from parameters, for a population (nPop) of 15, where the number of samples (nSample) was 200, and there were 500 iterations (q value). The X-axis is the numbers of iterations. The lower number shows the faster to optimum value (Y-axis). Table 1 shows the output results for all problems for each algorithm, indicating that $IACO_R$ performed better than ACO_R in solving the optimization problem for 7 benchmarks and 2 chemical engineering solutions (B1, B2, B3, B4, B5, B6, B7, C1, and C2). Even though these two algorithms cannot get the optimum solutions better than some algorithms, the solutions are acceptable compared to the exact solutions. Fig. 3 compares $IACO_R$ and ACO_R for the one benchmark function and the chemical engineering problems with full convergence behavior, presenting the best optimum values of the parameters (15 nPop, 20 nSample, and 500 iterations), as well as the area of each machinery batch. The letter "X" followed by a number is the desired variable needed to achieve the optimal solution.

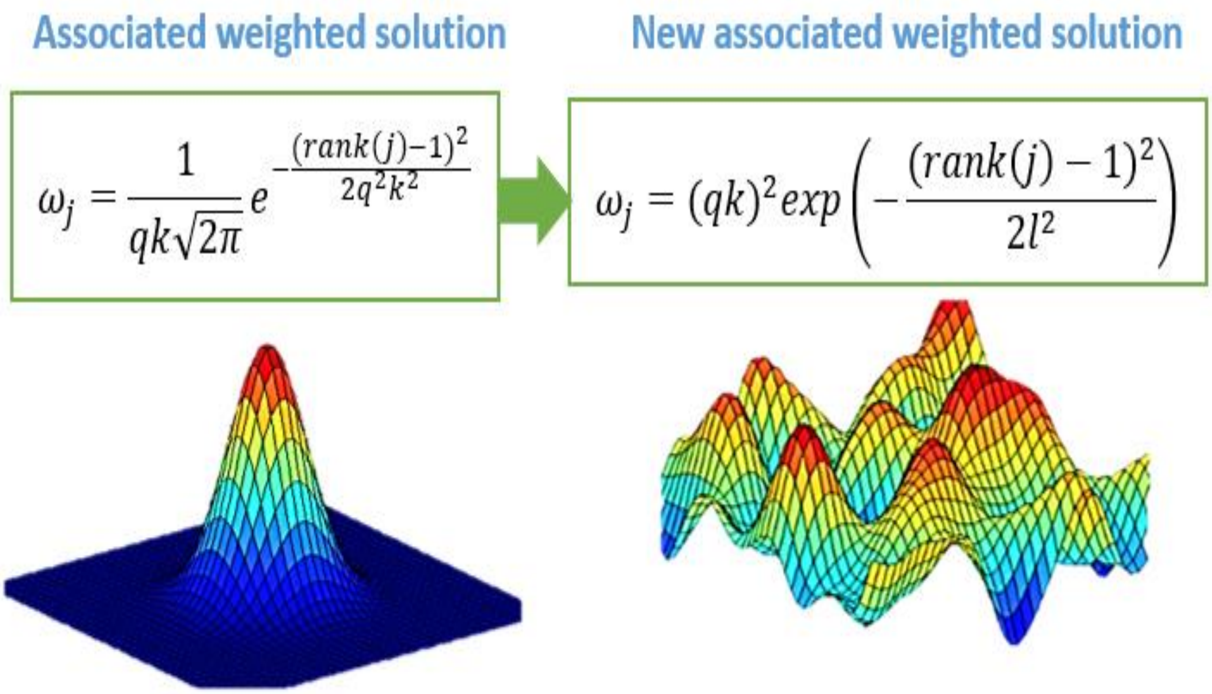


Fig. 1 Comparison of associated weighted solution.

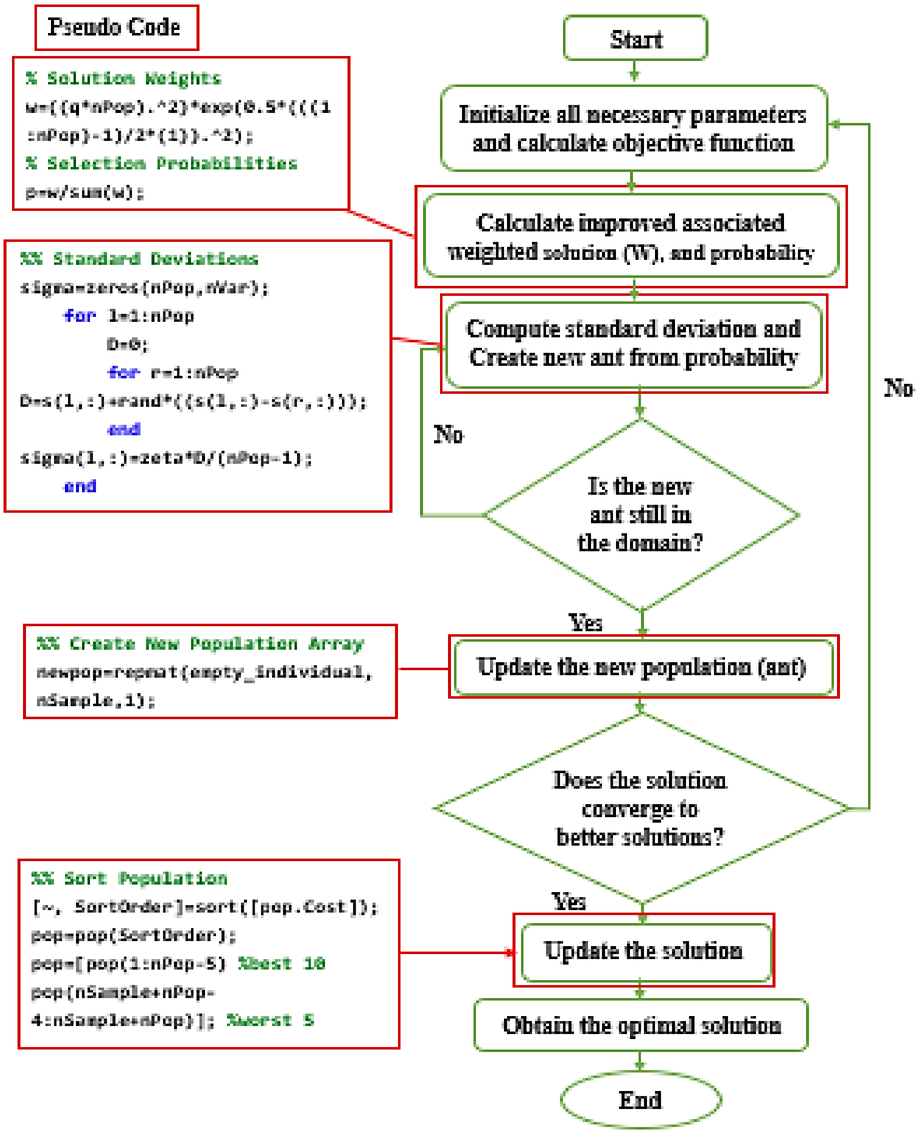
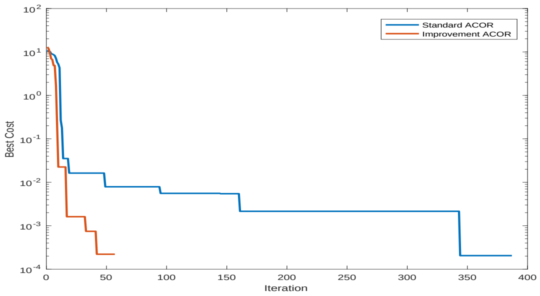


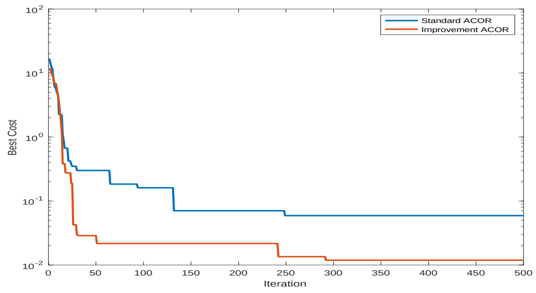
Fig. 2 Processing flowchart of improved real-value ant colony algorithm.

Table 1. Output results for common engineering problems for each algorithm

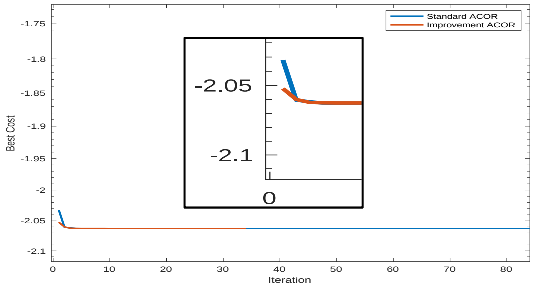
Code	Dimension	Exact Solution	Krill Herd Algorithm[7]	Teaching-Learning-Based Optimization	Sine Cosine Algorithm [11,12]	Ant Colony Algorithm (ACO_R)	Improved Ant Colony Algorithm ($IACO_R$)	$IACO_R$ Error (%)
B_1	2	0	2.1669e-06	8.8818e-16	8.8818e-16	0.000205	0.00022	0.0220
	5	0	9.6998e-05	8.8818e-16	8.8818e-16	0.059102	0.01190	1.1900
B_2	2	-2.0626	-2.0626	-2.0626	-2.0626	-2.0626	-2.0626	0
	2	-1.8013	-1.9936	-1.8013	-1.8011	-1.8013	-1.8008	0.02775
B_3	5	-4.68766	-3.8150	-4.67930	-3.1858	-4.64454	-4.64300	0.95371
	10	-9.6602	-6.4647	-9.4609	-5.3232	-9.31820	-9.31670	3.5558
	2	-0.36498	-0.3650	-0.36498	-0.3646	-0.36498	-0.36500	1.19112
B_4	5	-0.63445	-0.4373	-0.634449	-0.6176	-0.63433	-0.63440	0.00788
	20	-0.80362	-0.1792	-0.76754	-0.4125	-0.78695	-0.63324	21.2010
B_5	2	263.8959	275.6835	263.8958	263.8561	263.8959	263.8959	0
B_6	3	0.012665	0.0127	0.012668	0.0128	0.012852	0.01270	0.2763
B_7	4	1.724852	1.9039	1.72485	1.7751	1.72680	1.72780	0.1708
C_1	2	7049.249	7049.2000	7049.24927	7049.260	7049.2493	7049.2496	8.5115E-06
C_2	2	0.38881	0.3888	0.388811	0.3821	0.388671	0.38871	0.02572



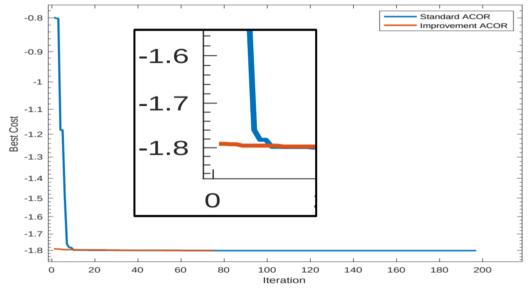
(A) Benchmark problem 1 with d=2



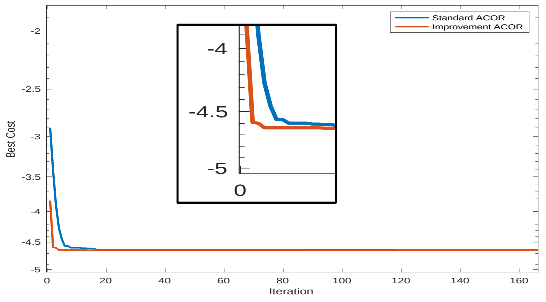
(B) Benchmark problem 1 with d=5



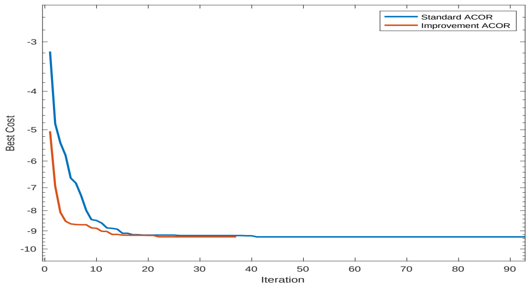
(C) Benchmark problem 2



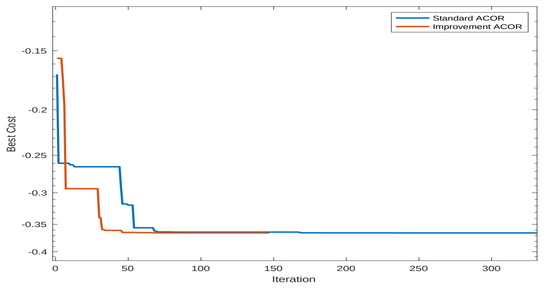
(D) Benchmark problem 3 with d=2



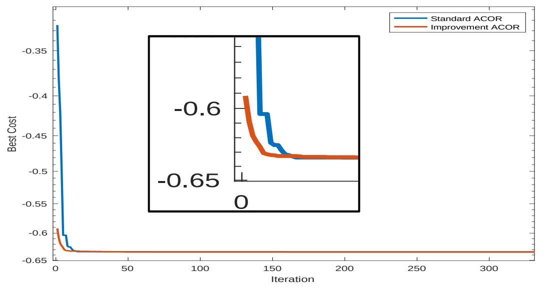
(E) Benchmark problem 3 with d=5



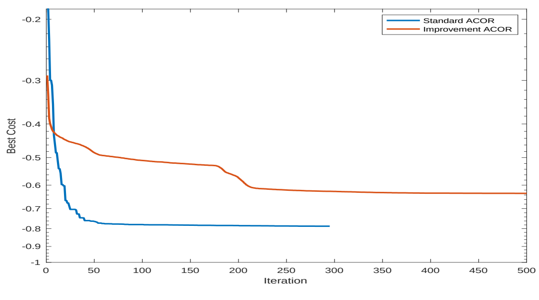
(F) Benchmark problem 3 with d=10



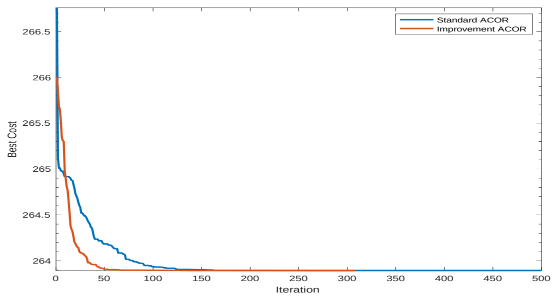
(G) Benchmark problem 4 with d=2



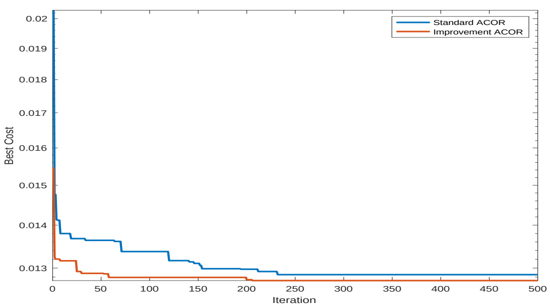
(H) Benchmark problem 4 with d=5



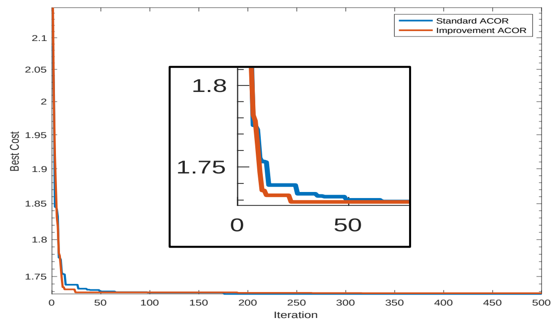
(I) Benchmark problem 4 with d=20



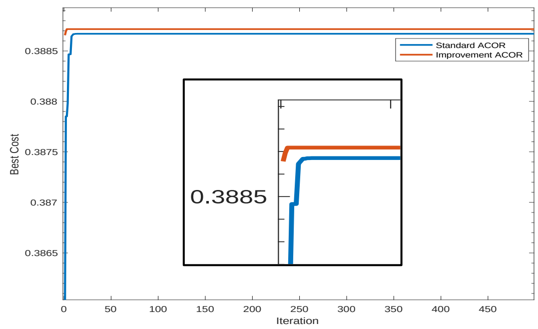
(J) Benchmark problem 5 with d=2



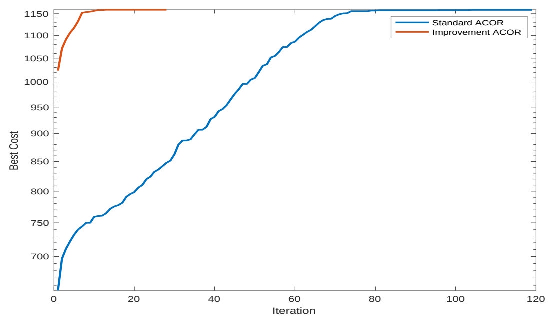
(K) Benchmark problem 6



(L) Benchmark problem 7



(M) Chemical engineering problem: Reactor network design



(N) Chemical engineering problem: Alkylation process optimization

Fig. 3 Convergence comparison of best value between improved ACOR and standard ACOR where the computation time is substantially reduced by various positions after adjusting the associated weight.

An additional 19 benchmark testing functions were used to test whether our algorithm could solve more complex and multidimensional problems. The 9 additional benchmarks are new functions, so they have no published answers. Nonetheless, the results are shown in Table B in the appendix.

3.1 Application of IACOR with process simulation using aspen plus

The proposed optimizer should work seamlessly with the process simulator.[15] This section demonstrates 2 examples of the implementation of the proposed algorithm using the simulation package Aspen Plus.

3.1.1 C1: Optimization of ethanol column

Ethanol column optimization for capital and operating expenditures was calculated using the Aspen Plus and MATLAB software packages. There were 3 optimum operations (extracted stages, reflux ratio, and feed stage data) used to calculate capital expenditures (CAPEX) and operating expenses (OPEX), as shown in Fig. 4. CAPEX consists of the primary purchases a company makes that are intended to be used over the long term. OPEX consists of the day-to-day expenses a company incurs to maintain its business operations. The $IACO_R$ implemented the data for an ethanol column based on simulation in the Aspen Plus software. The proposed algorithm interacts with the steady state data from the simulation and improves the specific conditions to approach the optimum.

As shown in Fig. 5, the MATLAB code for the ethanol column problem in which the variable stage is set to represent the number of stages in the distillation tower, the variable Reflux Ratio represents the ratio of returning substances to the distillation tower, and the variable Feed Stage represents the stages used to feed substances into the distillation tower. All three variables carry values obtained from manual calculations. The algorithm to think about is the variable X. To apply it to Aspen, we will use the command Aspen.Tree.FindNode receives all 3 variables to set up the ethanol distillation tower. In using Aspen with MATLAB. We have to set the Error and Time values. In running, we have to get the Error value equal to 0 in the time we specified which is 40 and the problem, there is a constraint value set as a variable as the Purity value must be greater than 0.79 and the Recovery value must be greater than 0.95 in the word section. The answer we need is the value from the Obj variable that stores the CAPEX and OPEX variables.

The objective functions were CAPEX and OPEX, which were calculated based on the same decision variables in each

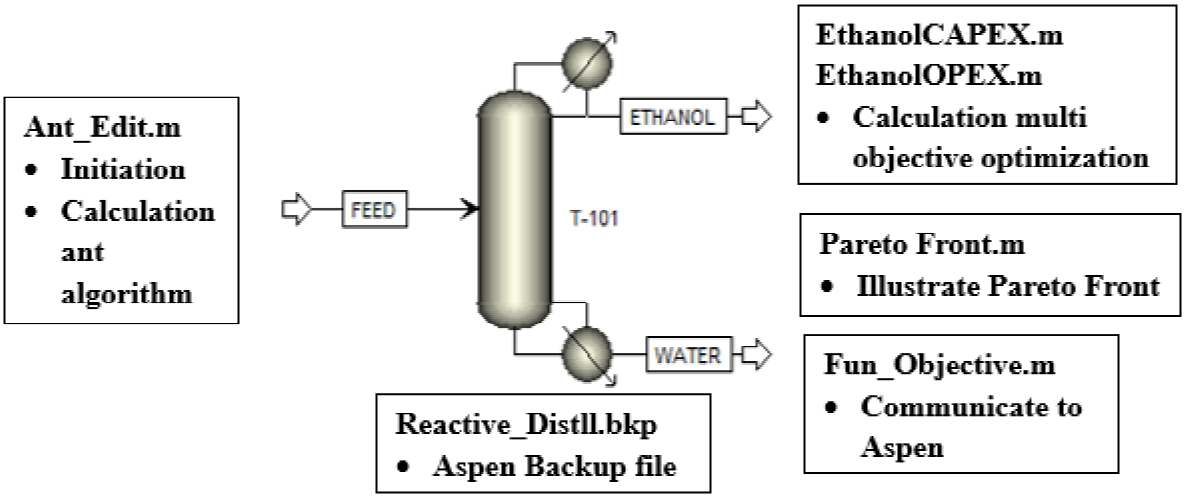


Fig. 4 Ethanol column model.

```
function [Obj, Cons] = Fun_Objective(X)
global Aspen
Stages = X(1) + 2;
Reflux_Ratio = round(X(2),4);
Feed_Stage = round((Stages-6)*X(3) + 3); %x3: Normalized Dist. Feed Stage.
Aspen.Tree.FindNode("\Data\Blocks\T-101\Input\NSTAGE").Value = Stages;
Aspen.Tree.FindNode("\Data\Blocks\T-101\Input\BASIS_RR").Value =
Reflux_Ratio;
Aspen.Tree.FindNode("\Data\Blocks\T-101\Subobjects\Tray
Sizing\1\Input\TS_STAGE2\1").Value = Stages - 1;
Aspen.Tree.FindNode("\Data\Blocks\T-101\Input\FEED_STAGE\FEED").Value =
Feed_Stage;

    Aspen.Reinit; % Reinit simulation
    Aspen.Engine.Run2(1); %Run the simulation. (1) ---> Matlab isnt busy; (0)
Matlab is Busy;
    time = 1;
    Error = 0;
    while Aspen.Engine.IsRunning == 1 % 1 --> If Aspen is running; 0 ---> If
Aspen stop.
        pause(0.5);
        time = time+1;
        if time==40 % Control of simulation time.
            Aspen.Engine.Stop;
            Error = 1;
        end
    end
    if Error == 0
        %Pur : Ethanol Purity at distillation column top
        Pur =
Aspen.Tree.FindNode("\Data\Streams\ETHANOL\Output\MOLEFLOW\MIXED\ETHANOL").Va
lue/Aspen.Tree.FindNode("\Data\Streams\ETHANOL\Output\TOT_FLOW").Value;
        Purity = 0.79 - Pur; %This means: Pur > 0.79 mol frac.
        %Recovery: Ethanol Recovery at distillation column top.
        %Recovery > 0.95
        Recovery = 0.95 -
(Aspen.Tree.FindNode("\Data\Streams\ETHANOL\Output\MOLEFLOW\MIXED\ETHANOL").V
alue/Aspen.Tree.FindNode("\Data\Streams\FEED\Output\MOLEFLOW\MIXED\ETHANOL").
Value);
        %Constraints Vector
        c(1,1) = Purity;
        c(1,2) = Recovery;
        Cons = (c>0).*c; %Only accept unsatisfied constrainst: c > 0
        CAPEX = EthanolCAPEX(); %Fixed Distillation Cost
        OPEX = EthanolOPEX()/5; %Operating Distillation Cost
        Obj  = [CAPEX, OPEX]; %Objectives Function. CAPEX vs OPEX
    else
        Obj  = [2e7, 2e7]; %Penalty
        Cons = [1, 1];
    end
end
```

Fig. 5 Objective function of ethanol column.

Table 2. Solution for optimization of ethanol column.

No. of iterations	Decision variable			IACO$_R$ algorithm	
	Stage number	Reflux ratio	Feed stage	OPEX (USD/year)	CAPEX (USD)
1	37	2.01	21	347,210	2,671,588
2	17	2.21	10	375,030	2,585,850
3	18	1.19	13	270,730	2,437,852
4	16	1.21	10	274,250	2,265,694
5	14	1.92	8	349,660	2,064,574
6	14	2.05	8	362,460	2,008,680
7	13	1.82	8	339,880	1,863,914
8	13	1.68	7	326,140	1,860,714
9	13	1.02	7	257,450	1,859,208
10	13	1.01	7	254,500	1,857,693

iteration (number of stages, the reflux ratio, and the feed stage). The constraint was better than 99 percent ethanol purification, as shown in Table 2.

3.1.2 C2: Effect of design of minimum heat exchanger area on heat exchanger network design

In this problem, the objective was to determine the minimum area of the 3 heat exchangers, as shown in Fig. 6. Aspen Plus and MATLAB were used to perform the optimization to identify the minimum heat exchanger area of each unit. These exchangers were responsible for heating a cold stream with a heat capacity flow rate of 100,000 kW/°F from 100 to 500°F. The overall coefficients of heat exchangers 1, 2, and 3 were 120, 80, and 40 kW/ft^2°F, respectively. The minimum area of the total three heat exchangers (7,050.7 ft^2) was allocated to heat exchangers 1, 2, and 3 with 584.9, 1,419.3, and 5,046.5 ft^2, respectively. The outlet temperatures of heat exchangers 1, and 2 were 182.5 and 298.1 °F, respectively.

Shown in Fig. 7 is the MATLAB code for the problem about heat exchanger area, where the variable Area1 represents the heat exchange area of the 1st heat exchanger, the variable Area2 represents the heat exchange area of the 2nd heat exchanger, and the variable Area3 represents the area. Heat exchange of the third heat exchanger, the two variables are Area1 and Area2, obtained from the calculation with the algorithm will be calculated as the variable X. To apply it to Aspen, we will use the command Aspen.Tree.FindNode receives the values of both variables to design the third heat exchanger and in the problem has constraint values set as variables. Temp1 and Temp1 are used to represent the temperature values of the first and second heat exchangers. In the answer we need is the value from the variable Obj that stores the variable Tot_Area that is the value of the heat exchange area of the three heat exchangers and the constraint value.

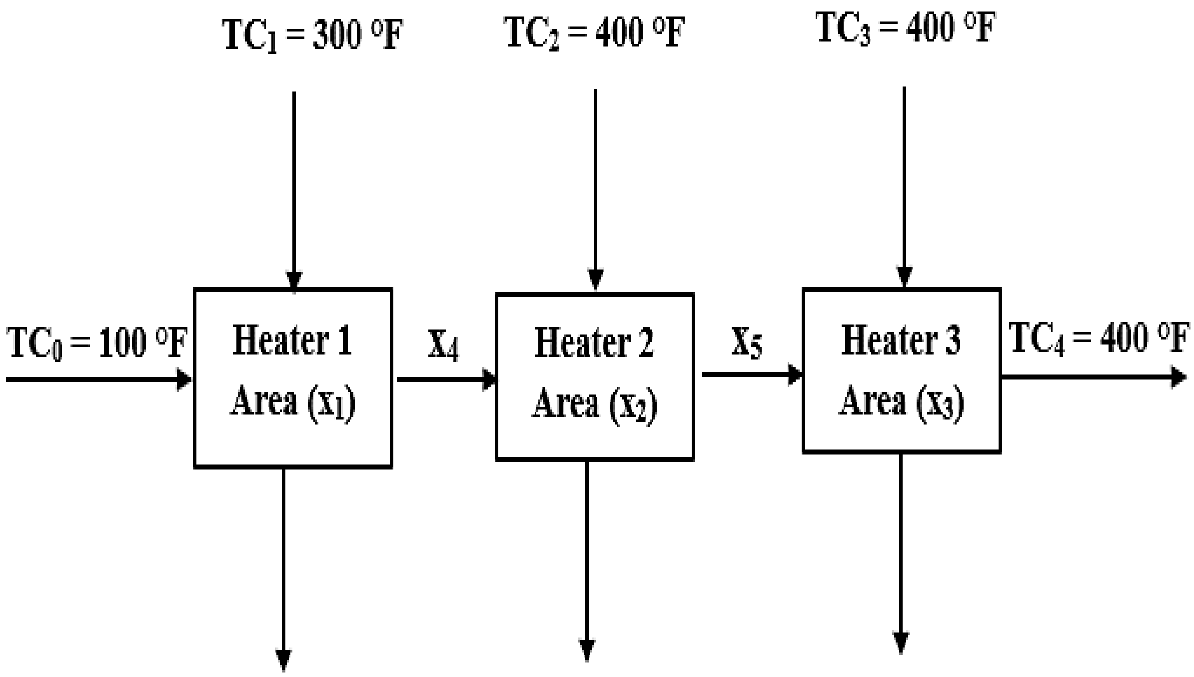


Fig. 6 Heat exchanger network.

```
    function Obj = ObjectiveFun_ChemBM1(X)
    global Aspen
    Obj =[];
    Area1 = X(1); % Heat exchanger area 1
    Area2 = X(2); % Heat exchanger area 2
    Aspen.Tree.FindNode('\Data\Flowsheeting
Options\Calculator\BM1\Input\FVN_INIT_VAL\AREA1').Value = Area1;
    Aspen.Tree.FindNode('\Data\Flowsheeting
Options\Calculator\BM1\Input\FVN_INIT_VAL\AREA2').Value = Area2;
        Aspen.Reinit;           % Reinit simulation
        Aspen.Engine.Run2(1);    %Run the simulation. (1) ---> Matlab isnt
busy; (0) Matlab is Busy;
        while Aspen.Engine.IsRunning == 1 % 1 --> If Aspen is running; 0 --->
If Aspen stop.
            pause(0.5);
        end
    Conv = Aspen.Tree.FindNode('\Data\Results Summary\Run-
Status\Output\PER_ERROR').Value; %Convergence Assessment
        if Conv == 0
            % Area3 : Heat exchanger area 3 (Constraint)
            Area3 = Aspen.Tree.FindNode('\Data\Flowsheeting
Options\Calculator\BM1\Output\WRITE_VAL\3').Value;
            % The objective is to find the minimum area of three heat
exchangers
            Tot_Area = Area1+Area2+Area3;
            % Temp : Temp out put of HX1 and HX2 (Constraints)
            Temp1 = Aspen.Tree.FindNode('\Data\Flowsheeting
Options\Calculator\BM1\Output\WRITE_VAL\4').Value;
            Temp2 = Aspen.Tree.FindNode('\Data\Flowsheeting
Options\Calculator\BM1\Output\WRITE_VAL\5').Value;
            %Constraints Vector
            c(1,1) = Area3;
            c(1,2) = Temp1;
            c(1,3) = Temp2;
            Obj  = [Tot_Area, c]; %Objectives Function
        else
            Obj  = [62083, 9999, 299, 399];
        end
    end
```

Fig. 7 Objective function of heat exchanger network design.

4. Conclusions

Functions of benchmark problems were used to validate the performance of the proposed method. However, sometimes there were errors due to computer performance or the MATLAB programs that could result in a suboptimal solution. Therefore, this article proposed a new improved algorithm ($IACO_R$). The improved algorithm's success was associated with a weighted solution (ω) that improved the algorithm's efficiency when dealing with complicated solutions. Furthermore, it increased the possibility of reaching the optimal result by improving convergence to an exact value. All the results from the 7 benchmarks, 4 chemical engineering problems, and 19 benchmark testing functions, showed that this enhanced method could determine a better result than the standard algorithm and could achieve the actual optimal solution with reasonable error from exact solution (lower than 5%). However, even though the improved algorithm had excellent efficiency at reaching the optimal value, it was unsuitable for achieving high accuracy for high dimensional problems, such as benchmark B2 running with 20 dimensions and chemical engineering problem C2 with complex constraints. The results showed that there was still a high percentage of error, though not as much as for the standard ACO_R. Nonetheless, this enhanced method was superior to the standard algorithm.

Acknowledgements

This work was supported by master's degree study plan A, Faculty of Engineering Kasetsart University, Bangkok, Thailand that provided funding to support this project under research project code 65/15/CHEM/M.Eng.

Appendix

Codes are available at https://drive.google.com/drive/folders/1EjIoCrW4qdiIal2dO_w547v Omag XU4Pz?usp=drivelink

Conflict of Interest

There is no conflict of interest.

Supporting Information

Not applicable.

References

[1] K. Iqbal, Fundamental Engineering Optimization Methods Second Edition, 1st ed, 2013, https://bookboon.com/en/fundamental-engineering-optimization-methodsebook?mediaTy pe=ebook.

[2] O. Bozorg-Haddad, M. Solgi, H. A. Loáiciga, Meta-heuristic and evolutionary algorithms for engineering optimization, John Wiley & Sons, 2017, doi: 10.1002/9781119387053.

[3] M. Dorigo, Ant colony optimization, *Scholarpedia*, 2007, **2**, 1461, doi: 10.4249/scholarpedia.1461.

[4] M. Dorigo, M. Birattari, T. Stutzle, Ant colony optimization, *IEEE Computational Intelligence Magazine*, 2006, **1**, 28-39, doi: 10.1109/MCI.2006.329691.

[5] S. Sukpancharoen and T.R. Srinophakun, Application of the ant colony algorithm to optimize the synthesis of heat-integrated distillation sequences for 6-components (C4-C7), *International Journal of Mechanical Engineering and Technology*, 2018, **9**, 514-530, https://iaeme.com/Home/issue /IJMET?Volume=9&Issue=4.

[6] T. Naraattcha, N. Tongprik, and T.R. Srinophakun, Biogeography-Based Optimization by Using Crossover Operator for Chemical Engineering Problems, *Kasetsart Engineering Journal*, 2020, **109**, 13-22, https://tci-thaijo.org/index.php/kuengj.

[7] P. Suwajittanont, K. Peantham, T. R. Srinophakun, Enhanced Krill Herd algorithm for chemical engineering problems, Kasetsart Engineering Journal, 2020, **33**, 1-12, https://tci-thaijo.org/index.php/kuengj.

[8] W. Wongrat, T. Srinophakun, P. Srinophakun, Modified genetic algorithm for nonlinear data reconciliation, *Computers & Chemical Engineering*, 2005, **29**, 1059-1067, doi: 10.1016/j.compchemeng.2004.11.005.

[9] J. Thunyawart, T. Srinophakun, W. Henwatthana, Simulation of mass exchange networks using modified genetic algorithms, *Korean Journal of Chemical Engineering*, 2011, **28**, 332-341, doi: 10.1007/s11814-010-0360-y.

[10] T. Wasanapradit, N. Mukdasanit, N. Chaiyaratana, T. Srinophakun, Solving mixed-integer nonlinear programming problems using improved genetic algorithms, *Korean Journal of Chemical Engineering*, 2011, **28**, 32-40, doi: 10.1007/s11814-010-0323-3.

[11] R. Junsittiwate, T. R. Srinophakun, S. Sukpancharoen, Multi-objective atom search optimization of biodiesel production from palm empty fruit bunch pyrolysis, *Heliyon*, 2022, **8**, e09280, doi: 10.1016/j.heliyon.2022.e09280.

[12] M. Chanthasuwannasin, B. Kottititum, T. Srinophakun, A mixed coding scheme of a particle swarm optimization and a hybrid genetic algorithm with sequential quadratic programming for mixed integer nonlinear programming in common chemical engineering practice, *Chemical Engineering Communications*, 2017, **204**, 840-851, doi: 10.1080/00986445.2017.1294583.

[13] H. A. Vedaie, A. Farzan, Maximum likelihood constrained Gaussian kernel probability density function estimation using continuous ACO, *Optik*, 2016, **127**, 2354-2359, doi: 10.1016/j.ijleo.2015.11.088.

[14] A. Kaveh, S. Talatahari, An improved ant colony optimization for constrained engineering design problems, *Engineering Computations*, 2010, **27**, 155-182, doi: 10.1108/02644401011008577.

[15] C. Janta-in, K. Wiriyalapsakul, and T.R. Srinophakun, Application of teaching-learning-based optimization algorithm in designing 100 kW power plant using pyrolysis of oil palm empty fruit bunches, *Agriculture and Natural Resources*, 2023, **57**, 87-98, https://li01.tci-thaijo.org/index.php/anres/article/view/258256.

[16] S. Sukpancharoen, T. R. Srinophakun, P. Aungkulanon, Grey wolf optimizer (GWO) with multi-objective optimization for biodiesel production from waste cooking oil using central composite design (CCD), *International Journal of Mechanical Engineering and Robotics Research*, 2020, 1219-1225, doi: 10.18178/ijmerr.9.8.1219-1225.