Journal of Advanced Research in Applied Sciences and Engineering Technology

Journal homepage:
https://semarakilmu.com.my/journals/index.php/applied_sciences_eng_tech/index
ISSN: 2462-1943

Cubature Kalman Optimizer: A Novel Metaheuristic Algorithm for Solving Numerical Optimization Problems

Zulkifli Musa[1,3], Zuwairie Ibrahim[2,*], Mohd Ibrahim Shapiai[3], Yusei Tsuboi[4]

[1] Faculty of Electrical and Electronics Engineering Technology, Universiti Malaysia Pahang, 26600 Pekan, Malaysia
[2] Faculty of Manufacturing, Universiti Malaysia Pahang, 26600 Pekan, Malaysia
[3] Malaysia-Japan International Institute of Technology, Universiti Teknologi Malaysia, 54100 Kuala Lumpur, Malaysia
[4] SICK K.K. 1-32-2 Honcho, Nakanoku, Tokyo, Japan

ARTICLE INFO

Article history:
Received 23 May 2023
Received in revised form 26 September 2023
Accepted 7 October 2023
Available online 20 October 2023

Keywords:
Optimization; Metaheuristic; CKF; local search neighborhood

ABSTRACT

This study introduces a new single-agent metaheuristic algorithm, named cubature Kalman optimizer (CKO). The CKO is inspired by the estimation ability of the cubature Kalman filter (CKF). In control system, the CKF algorithm is used to estimate the true value of a hidden quantity from an observation signal that contain an uncertainty. As an optimizer, the CKO agent works as individual CKF to estimate an optimal or a near-optimal solution. The agent performs four main tasks: solution prediction, measurement prediction, and solution update phases, which are adopted from the CKF. The proposed CKO is validated on CEC 2014 test suite on 30 benchmark functions. To further validate the performance, the proposed CKO is compared with well-known algorithms, including single-agent finite impulse response optimizer (SAFIRO), single-solution simulated Kalman filter (ssSKF), simulated Kalman filter (SKF), asynchronous simulated Kalman filter (ASKF), particle swarm optimization algorithm (PSO), genetic algorithm (GA), grey wolf optimization algorithm (GWO), and black hole algorithm (BH). Friedman's test for multiple algorithm comparison with 5% of significant level shows that the CKO offers better performance than the benchmark algorithms.

1. Introduction

Optimization is seen in many fields such as engineering, social science, economics, and business. It is a process of achieving an optimal solution to the problem. The optimal solution can be either a minimum or a maximum solution. In general, optimization methods can be divided into exact methods and approximate methods [1]. The exact methods may not be suitable for some complex optimization problems. Thus, approximate methods are the other option to solve these problems. Approximation algorithms and heuristic algorithms are subcomponents of approximate methods. Heuristic algorithms can be further classified into two classes: problem specific heuristics and metaheuristics [1]. Problem-specific heuristics are problems-dependent algorithms whereas

* *Corresponding author.*
E-mail address: zuwairie@ump.edu.my

https://doi.org/10.37934/araset.33.1.333355

metaheuristics are more general algorithms and can be used to solve various types of optimization problems with minimum modification.

Metaheuristic algorithms have gained huge popularity and attracted researcher's attention because of its flexibility and ability in solving large scale and variety of optimization problems. These algorithms have iterative and stochastic behavior. Metaheuristic algorithms can be classified into five source of inspiration which are evolution algorithms, swarm intelligence algorithms, physics inspired algorithms, human and animal lifestyle algorithms, and estimation-based algorithms as shown in Figure 1. The first category, "Evolutionary Algorithms," represents the algorithms that are developed based on biological reproduction and evolution. The most popular evolution-inspired technique is the genetic algorithm (GA), which simulates Darwinian evolution [2] The GA have also been applied to solve engineering problems search as light-shelves and solar flat plate collector [3,4]. Other algorithms are differential evolution (DE), evolution strategy (ES), and Bayesian evolutionary algorithm (BEA) [5-7]. The second category is "Swarm Intelligence algorithms," which mimics the social behavior of groups of animals. The most popular algorithm is particle swarm optimization (PSO) that inspired by the social behavior of bird flocking and gray wolf optimization (GWO) inspired by grey wolves [8,9] Other algorithms are rat swarm optimizer, and Chameleon swarm algorithm [10,11]. The third category is "Physics-Inspired Algorithms" which imitate the physical rules of the universe. The most popular algorithms are the black hole (BH) algorithm based on the observable fact of black hole phenomena [12]. Other algorithms are sine cosine algorithm (SCA) and atomic orbital search [13,14]. The fourth category is "Human and Animal Lifestyle Algorithms". Some of the most popular algorithms are colliding bodies optimization (CBO), interior search algorithm (ISA), and mine blast algorithm (MBA) [15-17]. In the last category, "Estimation-based Algorithms" represents the algorithms that are developed based on estimation system. Some of the algorithms inspired by the estimation system are listed in Table 1. The heuristic Kalman algorithm (HKA) is the first algorithm inspired by Kalman filter [18]. Other algorithms are simulated Kalman filter (SKF), single solution simulated Kalman filter (ssSKF), asynchronous simulated Kalman filter (ASKF), and single agent finite impulse response optimizer (SAFIRO) [19-22].

The estimation-based algorithms are not very popular and lacking in number compared to others type under metaheuristic algorithms. Hence, it is always interesting and beneficial to discover the source of inspiration by looking away from nature. As stated by Wolpert and Macready [23], there is no optimization algorithm that is better than other algorithms in solving all optimization problems. Thus, there is room for exploring and developing new and effective optimization algorithms for solving several types of problems. However, the challenge in this field is how to get a good source of inspiration from the existing knowledge either to improve the existing algorithms or develop a new algorithm.

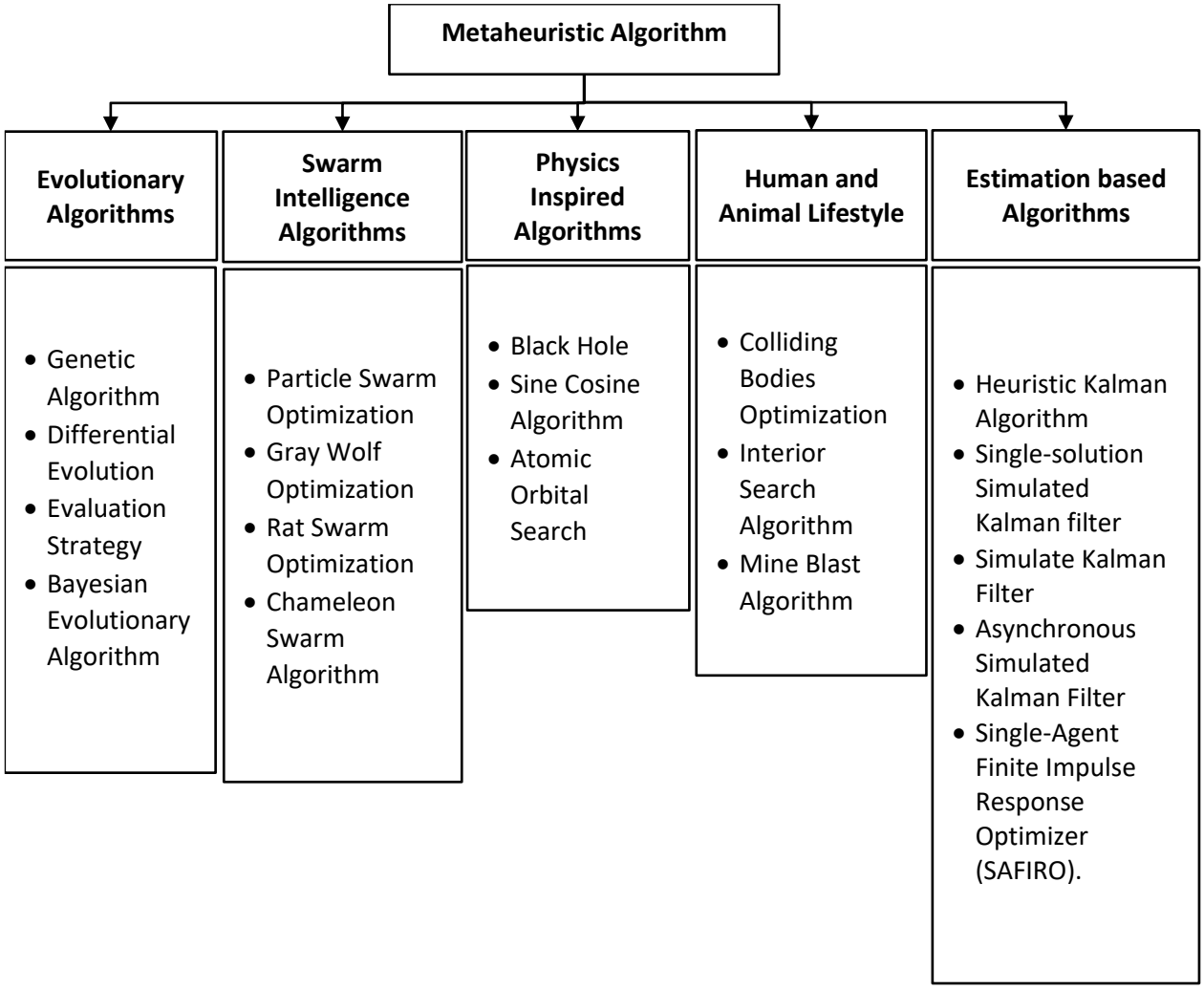


Fig. 1. Categorization of meta-heuristic algorithms

Table 1
Metaheuristic algorithm inspired by estimation system

Algorithm	Inspiration	Number of solutions		Year of proposal
		Individual-based	Population-based	
Heuristic Kalman Algorithm (HKA) [18]	Kalman filter (KF)		/	2009
Simulated Kalman Filter (SKF) algorithm [19-21]	Kalman filter (KF)		/	2016
Single-agent Finite Impulse Response Optimizer (SAFIRO) algorithm [22]	Unbiased finite impulse response (UFIR) filter	/		2018

Therefore, in this paper, a new metaheuristic optimization algorithm inspired by the estimation ability of the cubature Kalman filter (CKF) called cubature Kalman optimizer (CKO) is proposed. The CKO works with only one agent to find the best solution for solving a numerical optimization algorithm. Similar to the concept of the estimation system, the CKO algorithm adopt four-step from CKF: (1) solution prediction step, (2) simulated measurement (simulate observation signal), (3) measurement prediction step, and (4) solution update.

The rest of the paper is organized as follows: section 2 presents the inspiration source and section 3 presents step-by-step framework of the proposed CKO algorithm. In section 4, the empirical

evaluation is carried out and in section 5, simulation results are compared with other metaheuristic algorithms. Finally, the summary and conclusions of the study are presented in section 6.

2. The Cubature Kalman Filter (CKF) Algorithm

In the control system, an estimation system is used to estimate the true state value of a hidden quantity from an observation signal that contains process and measurement uncertainties. The estimation process is done by minimizing the uncertainty in the observation signal. The overview of the (upper area) 'real system' and (lower area) 'CKF estimation' is shown in Figure 2. In this real system, the actual value of the state variable $x_{act}(t)$ of the process is unknown. Therefore, the sensor is used to measure the state value. However, the observation signal $z(t)$, is not present the actual state value because it usually contains the process noise $Q(t)$ and measurement noise $R(t)$. As the solution, the CKF is used to estimate the actual value by minimizing the error of measurement. The CKF estimator is a closed-loop system developed with the feedback of the previous state updated. The CKF has a complete estimation process: (1) state prediction step (obtain $x_{k|k-1}$ and $P_{k|k-1}$), (2) measurement prediction step (obtain $z_{k|k-1}$, $Pzz_{k|k-1}$, and $Pxz_{k|k-1}$), and (3) state update step (obtain $x_{k|k}$ and $P_{k|k}$), to obtain the hidden quantities from the observation signal. A cubature transformation technique (CTT) is embedded in both state and measurement prediction stages. The CTT process included (1) generate cubature point, (2) propagated cubature point, and (3) mean cubature point. The CTT able to increase stability and accuracy by capturing the mean and error accurately [24].

Firstly, in the state prediction step, the process starts with generating two cubature points $[X_1, X_2]$ closest to the previous state's position $x_{k-1|k-1}$. The first cubature point X_1, is given the positive weight of the square root of the error, $\sqrt{P_{k-1|k-1}}$, while the second cubature point X_2 is given the negative weight of the square root of error, $-\sqrt{P_{k-1|k-1}}$. The process continues by propagating cubature points to a new position $[X_1^*, X_2^*]$ by using the state prediction function $f(.,.)$. Lastly, the predicted state position, $x_{k|k-1}$, is produced by calculating the mean value of propagated cubature point.

Secondly, the measurement prediction step shows a similar process as the state prediction step, except for the use of the square root of the predicted error $\sqrt{P_{k|k-1}}$ as the weight of cubature points generation and the measurement prediction function $h(.,.)$ as the propagation of the cubature points.

Finally, in the state update step, the estimated state is updated by using all amounts of predicted state value and some amount of innovation, where the percentage of innovation amount is determined by estimation gain.

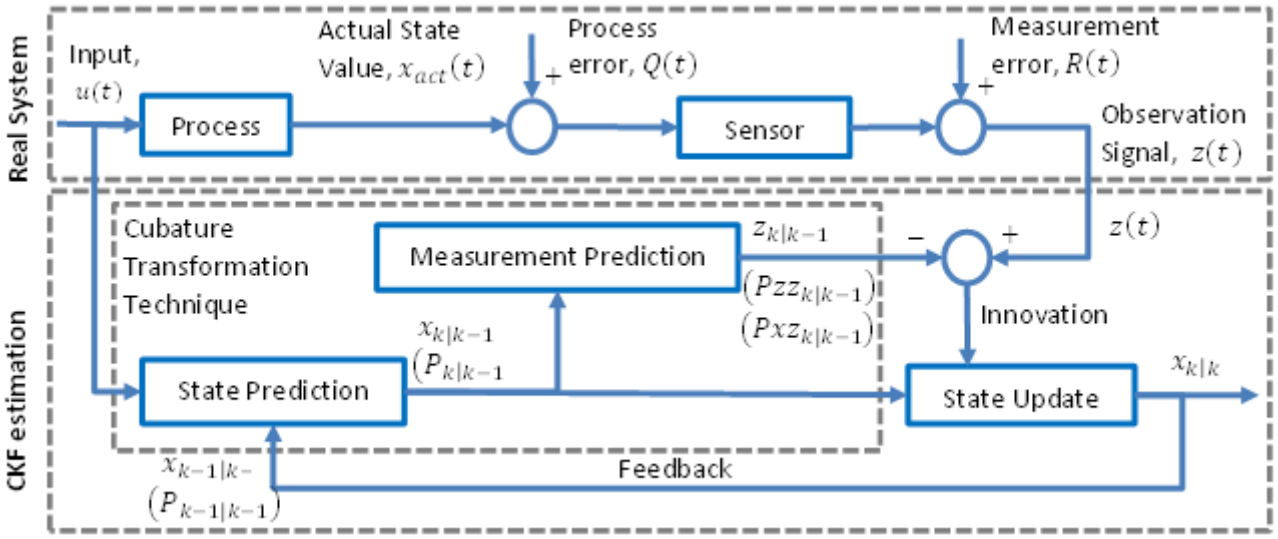


Fig. 2. The 'real system' and 'CKF estimation'

The detailed process of the scalar model CKF is derived as follows:

2.1 State Prediction

In the state prediction step, the predicted state variable, $x_{k|k-1}$, is acquired through the following equations:

$$X_{j_{k-1|k-1}} = x_{k-1|k-1} + \left[\sqrt{P_{k-1|k-1}} \quad -\sqrt{P_{k-1|k-1}}\right] \quad j = 1, 2 \tag{1}$$

$$X^*_{j\ k|k-1} = f\ (X_{j_{k-1|k-1}}, u(k-1)) \tag{2}$$

$$x_{k|k-1} = \frac{1}{2} \sum_{j=1}^{2} \left(X^*_{j\ k|k-1}\right) \tag{3}$$

where the generated two cubature points are represented by $X_{j_{k-1|k-1}} = [X_{1_{k-1|k-1}}, X_{2_{k-1|k-1}}]$ and the propagated two cubature points are represented by $X^*_{j\ k|k-1} = [X^*_{1\ k|k-1}, X^*_{2\ k|k-1}]$. The $\sqrt{P_{k-1|k-1}}$ is the positive cubature point weight and $-\sqrt{P_{k-1|k-1}}$ is the negative cubature point weight. The predicted state variable, $x_{k|k-1}$, is acquire by calculating the mean of two propagated cubature points.

The predicted state error $P_{k|k-1}$, is generated by implementing Eq. (4), as follows:

$$P_{k|k-1} = ((X^*_{1\ k|k-1} - x_{k|k-1}) \times (X^*_{2\ k|k-1} - x_{k|k-1})) \ + Q_k \tag{4}$$

where the $(X^*_{1\ k|k-1} - x_{k|k-1})$ is used to collect the square root of state error, $\sqrt{P}$ for the first cubature point $(j = 1)$, while $(X^*_{2\ k|k-1} - x_{k|k-1})$ is used to collect the square root of state error, $\sqrt{P}$, for the second cubature point $(j = 2)$. Then the predicted state error, $P_{k|k-1}$, is computed by multiplying between two square root state errors $(P = \sqrt{P} \times \sqrt{P})$ along with system noise, Q_k.

2.2 Measurement Prediction

In the measurement prediction step, the predicted measurement variable, $z_{k|k-1}$, is acquired through the following equations:

$$X_{j_{k|k-1}} = x_{k|k-1} + \left[\sqrt{P_{k|k-1}} \quad -\sqrt{P_{k|k-1}}\right] \quad j = 1, 2 \tag{5}$$

$$Z_{j_{k|k-1}} = h\ (X_{j_{k|k-1}}, u_{k-1}) \tag{6}$$

$$z_{k|k-1} = \frac{1}{2} \sum_{j=1}^{2} \left(Z_{j_{k|k-1}}\right) \tag{7}$$

where the generated cubature point and the propagated cubature point are represented by $X_{j_{k|k-1}} = [X_{1_{k|k-1}}, X_{2_{k|k-1}}]$ $X_{j_{k|k-1}} = \left[X_{1_{k|k-1}}, X_{2_{k|k-1}}\right]$ and $Z_{j_{k|k-1}} = \left[Z_{1_{k|k-1}}, Z_{2_{k|k-1}}\right]$, respectively. The $\sqrt{P_{k|k-1}}$ is the positive cubature point weight and $-\sqrt{P_{k|k-1}}$ is the negative cubature point weight. The predicted measurement variable, $z_{k|k-1}$, is acquired by calculating the mean of two propagated cubature points.

The measurement error, $Pzz_{k|k-1}$, and cross-error, $Pxz_{k|k-1}$, are generated by using the following equation:

$$Pzz_{k|k-1} = ((Z_{1_{k|k-1}} - z_{k|k-1}) \times (Z_{2_{k|k-1}} - z_{k|k-1})) + R_k \tag{8}$$

$$Pxz_{k|k-1} = ((X^*_{1\,k|k-1} - x_{k|k-1}) \times (Z_{2_{k|k-1}} - z_{k|k-1})) \tag{9}$$

where $(Z_{1_{k|k-1}} - z_{k|k-1})$ is used to collect the square root of measurement error, $\sqrt{Pzz}$, for the first cubature point $(j = 1)$, while $(Z_{2_{k|k-1}} - z_{k|k-1})$ is used to collect the square root of measurement error, $\sqrt{Pzz}$, for the second cubature point $(j = 2)$. Then the measurement error, $Pzz_{k|k-1}$, is computed by multiplying between two square root measurement errors $(Pzz = \sqrt{Pzz} \times \sqrt{Pzz})$ along with measurement noise, R_k. Meanwhile, the cross error, $Pxz_{k|k-1}$ is calculated by multiplying the square root of state error for the first cubature point $(j = 1)$, and the square root measurement error for the second cubature point $(j = 2)$: $(Pxz = \sqrt{Pxx} \times \sqrt{Pzz})$. After that, the gain, W_k, is computed using Eq. (10):

$$W_k = \ Pxz_{k|k-1}/Pzz_{k|k-1} \tag{10}$$

2.3 State Estimation

Lastly, Eq. (11) and Eq. (12) are used to update the estimated value for the state variable, $x_{k|k}$, and their corresponding error, $P_{k|k}$:

$$x_{k|k} = x_{k|k-1} + W_k\ (z(t) - z_{k|k-1}) \tag{11}$$

$$P_{k|k} = P_{k|k-1} - W_k\ Pzz_{k|k-1} \tag{12}$$

3. The Cubature Kalman Optimizer (CKO) Algorithm

Optimization deals with the problem of minimizing or maximizing objective functions given a defined domain or set of its constraints. Mathematically, a function minimization optimization problem can be written as the following equation:

$$\begin{array}{lll} Minimize: & f(x) & \\ Subject\ to: & g\_j\ (x)\ \leq 0 & j = 1,2 \dots, J \\ & h_k(x) = 0 & k = 1,2, \dots, K \\ & x_{iL}\ \leq x_i\ \leq\ x_{iU} & i = 1,2, \dots, I \end{array} \quad (13)$$

where the function $f(x)$ represents the objective (or goal) function, $g_j(x)$ is an inequality constraint and $h_k(x)$ is an equality constraint function. The x vector represents the I design variables that are modified to obtain the optimum solution. The searchable design space is defined by the upper and lower bounds, x_{iL} and x_{iU}, of the design variables. In general, the objective and constraint functions can be either linear or non-linear functions. Also, the functions can be either explicit or implicit functions.

The cubature Kalman optimizer (CKO) is a single-agent metaheuristic algorithm based on CKF framework, where one agent is employed to estimate the global minima or maxima. An agent in CKO holds the estimated solution position in the search space.

In CKO, an agent, $x_d(t)$, can be represented as in Eq. (14):

$$x(t) = \{x_1(t), x_2(t), \dots, x_d(t), \dots, x_D(t)\} \qquad d = 1,2, \dots, D \quad (14)$$

where a solution of the d^{th} dimension at t^{th} iteration is denoted by $x_d(t)$, the highest dimensional number is denoted by D.

The overall process of the CKO algorithm is divided into six main phases: (1) initialization, (2) fitness evaluation and update $X\_best\_so\_far$, (3) solution prediction, (4) simulate measurement, (5) measurement prediction, and (6) solution update as depicted in Figure 3. The first two processes are similar to most metaheuristic algorithms. In contrast, the solution prediction, simulated measurement, measurement prediction, and solution update phases are four dedicated operations in the proposed optimizer. The purpose of the simulated measurement phase is to simulate the actual measurement output in a real estimation process, while the solution prediction, measurement prediction, and solution update phases are adopted from the CKF.

The CKO algorithm starts with random initialization of the solution, $x_d(0)$ and its error, $P_d(0)$. After that, an agent is evaluated in the calculation of fitness. Then, according to the type of problem, the $X\_best\_so\_far_d$ is firstly updated. The $X\_best\_so\_far_d$ which holds the best solution so far, is updated only if $x_d(t)$ has obtained a better solution. For minimization problems, $X\_best\_so\_far_d$ is updated if the fitness of $x_d(t)$ is less than the fitness of $X\_best\_so\_far_d$. Meanwhile for the maximization problem, $X\_best\_so\_far_d$ is updated if the fitness of $x_d(t)$ is greater than the fitness of $X\_best\_so\_far_d$.

In the solution predicted step of CKO, the predicted solution, $xp_d(t)$ are generated based on the CTT using the following steps: (1) generate two cubature points of the solution $\{T1^1(t), T1^2(t)\}$, (2) propagate cubature point in the local neighborhoods search area $\{U1^1(t), U1^2(t)\}$, (3) generate a predicted solution by calculating the mean of the cubature point. Then, the predicted error, $Pp_d(t)$, is calculated.

In the simulated measurement step, the actual measurement, $z_d(t)$, is simulated around the predicted solution to introduce uncertainty, where the locus between the predicted solution and the best-so-far solution $|xp_d(t) - X\_best\_so\_far_d(t)|$ is the magnitude of measurement. Once the solution converges, the measurement magnitude is progressively decreased. The transition magnitude of measurement from high to low indicates the change from exploration stage to exploitation stage.

In the measurement prediction of CKO, the process is quite similar to the solution prediction step. The predicted measurement, $zp_d(t)$, is generated also based on CTT following these steps: (1) generate two cubature points of the solution, $\{T2^1(t), T2^2(t)\}$ (2) propagate cubature point in the local neighborhoods search area, $\{U2^1(t), U2^2(t)\}$ (3) generate predicted measurement by calculating the mean of the cubature point. Then, the innovation error, $Pzz_d(t)$, and cross-error, $Pxz_d(t)$, are calculated. The gain, $W_d(t)$, is calculated based on the ratio of these $Pzz_d(t)$ and $Pxz_d(t)$.

Finally, CKO performs the solution update step to update the solution $x_d(t+1)$, and its corresponding error $P_d(t+1)$. The estimated solution, $x_d(t+1)$ is updated based on the predicted solution and innovation, which is the difference between simulated measurement and predicted measurement.

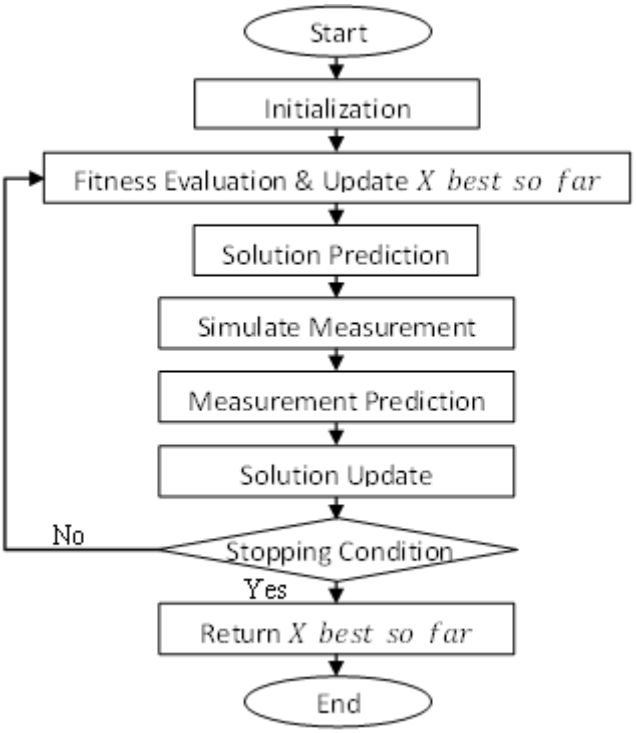


Fig. 3. The CKO flowchart

Note that, in the CKF, the cubature point propagations mentioned in Eq. (2) and Eq. (6) are done by using the state prediction function and measurement prediction function. However, in CKO, the local neighborhood term is used to propagate the cubature point in local search, where the radius, δ, of local search are updated in every iteration. This radius is large at earlier iterations to promote exploration stage. Meanwhile, when the iteration increases, the radius is scale down to favor exploitation stage. An adaptive value, β in local neighborhood formulation is employed to control how fast the radius, δ, is be reduced.

The detailed process of the CKO is given as follows:

3.1 Initialization Phase

The CKO algorithm starts with the random initialization of its agent, $x_d(0)$, within the search space as Eq. (15), where $\underline{x}_d$ is the lower limit and $\overline{x}_d$ is the upper limit of the search space in the dth dimension. Additionally, the initial value of the solution error, $P_d(0) \in [0,1]$, is generated using a random value as Eq. (16).

$$x_d(0) = randn_d + [\cup (\underline{x}_d , \overline{x}_d)] \tag{15}$$

$$P_d(0) = randn_d \tag{16}$$

3.2 Fitness Evaluation and X_best_so_far$_d$ Update Phase

The iteration begins with the fitness calculation of the solution. The fitness of the solution, $x_d(t)$ is compared to the fitness of $X\_best\_so\_far_d$ whereby $X\_best\_so\_far_d$ will be updated if the better solution ($x_d(t) < X\_best\_so\_far_d(t)$ for minimization problems, or $x_d(t) > X\_best\_so\_far_d(t)$ for maximization problem) is found.

3.3 Solution Prediction Phase

At first, the δ is determined by using the following Eq. (17):

$$\delta = e^{-\beta \times \frac{t}{tMax}} \times \frac{\underline{x}_d - \overline{x}_d}{2} \tag{17}$$

where t its current iteration and $tMax$ is maximum number of iterations. The predicted solution candidate, $xp_a(t)$, is determined by the following equations:

$$T1_d^j(t) = x_d(t) + \left[\sqrt{P_d(t)} \quad -\sqrt{P_d(t)}\right] \quad d = 1,2,\dots,D \text{ and } j = 1,2 \tag{18}$$

$$U1_d^j(t) = T1_d^j(t) + rand_d(\mathrm{U}\,[-\delta , \delta]) \tag{19}$$

$$xp_d(t) = \frac{1}{2} \sum_{j=1}^{2} U1_d^j(t) \tag{20}$$

where $T1_d^j(t)$ is the generated cubature point (two pints), jth is the cubature point, and $U1_d^j(t)$ is the propagation of both cubature point randomly in the search space by using a random element, $rand_d \in [0,1]$.Once the predicted solution candidate is calculated, the solution error, $Pp_d(t)$, need to be predicted by using Eq. (21):

$$Pp_d(t) = (\,(U1_d^1(t) - xp_d(t)) \times (U1_d^2(t) - xp_d(t))\,) + randn_d \tag{21}$$

where $randn_d \in [0,1]$ is used to present the system error.

3.4 Simulated Measurement Phase

The measurement step acts as feedback to the estimation process. The measurement of the agent is simulated based on the following Eq. (22):

$$z_d(t) = (xp_d(t) + sin(rand_d \times 2\pi) \times |xp_d(t) - X\_best\_so\_far_d(t)|) \quad (22)$$

where the simulated measurement value for the agent, $z_d(t)$, may take any random position in a locus $|xp_d(t) - X\_best\_so\_far_d(t)|$. A random element, $rand_d \in [0,1]$, in $sin(rand_d \times 2\pi)$ is responsible for the stochastic aspect of the CKO algorithm.

3.5 Measurement Prediction Phase

At first, the predicted measurement vector, $zp_d(t)$ is determined by the following equations:

$$T2_d^j(t) = xp_d(t) + \left[\sqrt{Pp_d(t)} \quad -\sqrt{Pp_d(t)}\right] \quad d = 1,2,\dots,D \text{ and } j = 1,2 \quad (23)$$

$$U2_d^j(t) = T2_d^j(t) + rand_d(\mathrm{U}\,[-\delta, \delta]) \quad (24)$$

$$zp_d(t) = \frac{1}{2}\sum_{j=1}^{2} U2_d^j(t) \quad (25)$$

where $T2_d^j(t)$ is the generated cubature point (two points) and jth is cubature points. The $U2_d^j(t)$ is the propagation of both cubature point randomly in the search space by using a random element, $rand_d \in [0,1]$. Once the predicted measurement, $zp_d(t)$ is calculated, the measurement error, $Pzz_d(t)$, and the cross-error, $Pxz_d(t)$ need to be estimated for gain calculation. These two types of error can be obtained by using Eq. (26) and Eq. (27), respectively:

$$Pzz_d(t) = ((U2_d^1(t) - zp_d(t)) \times (U2_d^2(t) - zp_d(t))) + randn_a \quad (26)$$

$$Pxz_d(t) = ((U1_d^1(t) - xp_d(t)) \times (U2_d^2(t) - zp_d(t))) \quad (27)$$

where $randn_d \in [0,1]$ is known as measurement noise. Then, the gain, $W_d(t)$, can be calculated using Eq. (28):

$$W_d(t) = Pxz_d(t)/Pzz_d(t) \quad (28)$$

3.6 Solution Update Phase

Finally, the solution, $x_d(t+1)$ and solution error, $P_d(t+1)$ are updated by the following equations:

$$x_d(t+1) = xp_d(t) + W_d(t)\big(z_d(t) - zp_d(t)\big) \quad (29)$$

$$P_d(t+1) = Pp_d(t) - W_d(t)Pzz_d(t) \quad (30)$$

This process is iteratively updated until the stopping condition is fulfilled.

4. Experimental Setup

The performance of CKO is evaluated using CEC 2014 benchmark suite, that contain with 3 unimodal functions, 13 simple multimodal functions, 6 hybrid functions, and 8 composition functions [25]. Then, the performance of the CKO is compared with other well-established optimization algorithms such as the SAFIRO, ssSKF, SKF, ASKF, PSO, GA, GWO and BH. The stopping condition is set at 1,000,000 for the number of function evaluations. The complexity of the benchmark functions is set at 50 dimensions, and the experiments are conducted 50 times. The initialization and parameter settings for all of the tested algorithms are listed in Table 2. All algorithms are implemented in MATLAB. The MATLAB codes for the CEC 2014 benchmark suite can be downloaded from http://www.ntu.edu.sg/home/EPNSugan/index_files/CEC2014.

It should be noted that the performance and convergence of these metaheuristic methods completely depend on the internal parameters of the algorithms. The CKO has only one parameter, β, where the appropriate value of β is considered to be 12. In Table 2, the key parameters of the selected methods are also presented. These values are recommended by the authors in the original papers. The references for each method are presented in the third column of Table 2.

Table 2
Parameter setting of the selected algorithms

Year	Algorithm	Reference	Parameter	Parameter Value
2022	Cubature Kalman Optimizer (CKO)	Current study	Number of agents	1
			coefficient, β	12
2017	Single agent Finite Impulse Response Optimizer (SAFIRO)	Ab Rahman *et al.,* [22]	Number of agents	1
			parameters (N)	4
			coefficient, β	5
2018	Single solution Simulate Kalman Filter (ssSKF)	Abdul Azizi *et al.,* [20]	Number of agents	1
			coefficient, β	5
2016	Simulate Kalman Filter (SKF)	Ibrahim *et al.,* [19]	Number of agents	100
2018	Asynchronous Simulate Kalman Filter (ASKF)	Aziz *et al.,* [21]	Number of agents	100
2019	Particle Swarm Optimizer (PSO)	Eberhart and Shi [26]	Number of agents	100
			Initial inertia weight, w_1	0.9
			Final inertia weight, w_2	0.5
			Cognitive acceleration factor, c_1	2
			Social acceleration factor, c_2	2
2004	Genetic Algorithm (GA)	Haupt and Haupt [27]	Population Size	100
			Crossover probability, *Pc*	0.5
			Mutation probability, *Pm*	0.2
2014	Grey Wolf Optimizer (GWO)	Mirjalili [9]	Number of agents	100
			Adaptive parameter, *a*	decreases from 2 to 0
			Coefficient vector, *C*	rand [0,2]
2013	Black Hole (BH)	Hatamlou [12]	Number of stars	100

A Friedman statistical test is then carried out to rank the algorithm based on the difference's performance between the algorithms at 5% significant level. In order to decide which algorithms are significantly difference from each other, the Holm post hoc procedure is chosen. These analyses are performed using the KEEL software [28].

5. Result and Discussion

5.1 Statistical Analysis

The mean values (mean) and standard deviation (std.) achieved by all tested algorithms for unimodal, simple multimodal, hybrid, and composition benchmark functions as shown in Table 3 until Table 6, respectively. The numbers written in **bold** indicate the best mean value obtained for the corresponding objective function among all tested algorithms. Meanwhile, Table 7 presents the ranking of algorithm performance using the Friedman test and Table 8 presents the significance level using Holm post hoc test.

5.1.1 Unimodal faction (Fn1 to Fn3)

Unimodal functions are related to rotation problems. According to Mirjalili [9], unimodal functions are correlated with exploitation benchmarking. As seen in Table 3, SAFIRO records the highest-ranking algorithm by producing the best solution in Fn1 and Fn3, while CKO records as a second-rank algorithm by producing the best solution in Fn2. Although CKO gets the second rank behind SAFIRO in Fn1 and Fn3, CKO is still very close to the optimal solution (near-optimal solution), especially for Fn1. The F1 is known as a difficult problem because it involved a quadratic ill-conditioned property [25]. Thus, the result for Fn3 indicates that CKO has an excellent performance in exploiting the optimum solution, where CKO was able to converge near the ideal solution of 300.

Table 3
Comparison of different methods in unimodal function

Fn		CKO	SAFIRO	ssSKF	SKF	ASKF	PSO	GA	GWO	BH
1	mean	1.79E+6	**4.49E+5**	4.92E+6	4.75E+6	3.76E+6	4.35E+7	3.41E+8	6.06E+7	4.36E+6
	Std.	5.81E+5	1.56E+5	1.26E+6	1.67E+6	1.44E+6	3.45E+7	8.25E+7	3.21E+7	9.39E+5
2	mean	**5.86E+3**	5.88E+3	1.30E+7	3.32E+7	1.71E+7	1.14E+7	2.3E+10	5.75E+9	1.14E+5
	Std.	5075.3	6156.4	1.45E+6	1.30E+8	4.69E+7	6.72E+7	3.80E+9	2.95E+9	1.19E+5
3	mean	3.02E+2	**3.00E+2**	3.66E+2	1.72E+4	1.59E+4	9.93E+3	6.11E+4	5.08E+4	1.14E+4
	Std.	3.9112	9.57E-5	11.959	9185.2	6403	9628.3	12227	11252	2150.1

5.1.2 Simple multimodal faction (Fn4 to Fn16)

As seen in Table 4, CKO clearly shows superior performance by producing the best solution in handling the majority of the evaluated simple multimodal functions. The CKO algorithm was managed to outperform other optimization algorithms in Fn8, Fn9, Fn10, Fn11, Fn13, Fn14, and Fn16. Meanwhile, for Fn5 and Fn7, CKO produced the best solution value as other SAFIRO algorithms. Although the CKO algorithm was outperformed by the SAFIRO algorithms in Fn4, Fn6 and Fn15, the produced outcomes were still very close to the optimal solution (near-optimal solution). Most of the simple multimodal functions are related to shifting and rotation problems. These functions are suitable for exploration benchmarking of an algorithm [9]. Hence, these results proved that apart from being great in exploitation, CKO is also very good in exploration, especially in Fn7, Fn8, Fn12, Fn13 and Fn14, where CKO successfully acquired a value that is very near to the ideal fitness of 700, 800, 1200, 1300 and 1400, respectively.

5.1.3 Hybrid faction (Fn17 to Fn22)

In hybrid functions, the variables are randomly divided into several subcomponents, where different basic functions (known as *N*) are used for different subcomponents [25]. The hybrid function is a combination of several multimodal functions (Fn19, Fn21, and Fn22), or it can be a combination of unimodal functions with simple multimodal functions (Fn17, Fn18, and Fn20). This consideration makes the function more complicated to solve. The readings in Table 5, show that SAFIRO shows the highest rank in the hybrid function, leading to three out of six functions. Although CKO only lead Fn20, the CKO manage to get the second rank for other hybrid functions (except Fn17, Fn18, and Fn21) and was still very close to the best solution of the first-rank algorithm.

Table 4
Comparison of different methods in simple multimodal function

Fn		CKO	SAFIRO	ssSKF	SKF	ASKF	PSO	GA	GWO	BH
4	mean	5.03E+2	**4.94E+2**	5.03E+2	5.23E+2	5.29E+2	1.06E+3	3.11E+3	9.59E+2	5.73E+2
	Std.	39.481	17.862	22.194	41.813	38.409	277.16	705.25	338.28	46.244
5	mean	**5.20E+2**	**5.20E+2**	5.21E+2	5.20E+2	5.20E+2	5.21E+2	5.21E+2	5.21E+2	5.20E+2
	Std.	0.00016	9.93E-6	0.02683	0.01016	0.00934	0.05942	0.05737	0.03547	0.03096
6	mean	6.19E+2	**6.16E+2**	6.19E+2	6.33E+2	6.31E+2	6.32E+2	6.56E+2	6.28E+2	6.57E+2
	Std.	3.9787	4.5968	4.0806	3.8908	4.981	5.2396	2.5233	3.831	4.7812
7	mean	**7.00E+2**	**7.00E+2**	7.01E+2	7.00E+2	7.00E+2	7.00E+2	9.33E+2	7.40E+2	7.00E+2
	Std.	0.00915	0.00788	0.01313	0.19049	0.13329	0.03346	37.768	22.884	0.07424
8	mean	**8.04E+2**	9.74E+2	9.78E+2	8.07E+2	8.07E+2	8.59E+2	1.07E+3	9.73E+2	9.22E+2
	Std.	1.9436	39.117	41.158	2.6967	3.1555	12.203	19.965	29.905	14.224
9	mean	**1.05E+3**	1.09E+3	1.09E+3	1.06E+3	1.06E+3	1.05E+3	1.40E+3	1.09E+3	1.22E+3
	Std.	29.508	48.567	41.377	37.868	35.547	29.129	32.909	29.112	46.976
10	mean	**1.23E+3**	5.88E+3	5.84E+3	1.35E+3	1.35E+3	1.64E+3	6.29E+3	6.20E+3	3.03E+3
	Std.	165.01	780.86	707.67	202.35	168.24	227.67	494.54	836.22	434.97
11	mean	**5.65E+3**	6.19E+3	6.40E+3	6.18E+3	6.15E+3	1.23E+4	1.28E+4	6.24E+3	8.11E+3
	Std.	712.34	799.4	1023	686.18	783.8	2163.6	428.18	843.1	987.62
12	mean	**1.20E+3**	1.20E+3	1.20E+3	1.20E+3	1.20E+3	1.20E+3	1.20E+3	1.20E+3	1.20E+3
	Std.	0.05156	0.04402	0.37471	0.09492	0.08893	0.40237	0.34351	1.5175	0.23441
13	mean	**1.30E+3**	1.30E+3	1.30E+3	1.30E+3	1.30E+3	1.30E+3	1.30E+3	1.30E+3	1.30E+3
	Std.	0.09737	0.14115	0.10022	0.09047	0.06953	0.10881	0.38416	0.08391	0.03733
14	mean	**1.40E+3**	1.40E+3	1.40E+3	1.40E+3	1.40E+3	1.40E+3	1.46E+3	1.41E+3	1.40E+3
	Std.	0.24405	0.36392	0.29441	0.10574	0.06568	0.09850	9.7036	10.178	0.01409
15	mean	1.51E+3	**1.51E+3**	1.53E+3	1.56E+3	1.55E+3	1.53E+3	3.82E+4	2.00E+3	1.77E+3
	Std.	2.6185	2.7525	3.1778	20.536	17.527	7.9526	26070	653.12	49.778
16	mean	**1.62E+3**	1.62E+3	1.62E+3	1.62E+3	1.62E+3	1.62E+3	1.62E+3	1.62E+3	1.62E+3
	Std.	0.78473	0.80952	0.56302	0.81281	0.71454	0.68879	0.4513	0.97739	0.60571

Table 5
Comparison of different methods in hybrid functions

Fn		CKO	SAFIRO	ssSKF	SKF	ASKF	PSO	GA	GWO	BH
17	mean	1.53E+5	**2.65E+4**	3.26E+5	8.43E+5	8.35E+5	2.47E+6	1.60E+7	3.39E+6	5.53E+5
	Std.	78418	11610	1.48E+5	4.58E+5	4.81E+5	2.59E+6	7.60E+6	3.00E+6	1.94E+5
18	mean	3.38E+3	4.31E+3	3.74E+5	4.71E+6	8.85E+6	1.21E+4	5.16E+6	2.59E+7	**2.40E+3**
	Std.	962.31	1471	64934	1.57E+7	3.75E+7	39429	2.63E+6	6.12E+7	250.46
19	mean	1.94E+3	**1.92E+3**	1.92E+3	1.95E+3	1.95E+3	1.96E+3	2.01E+3	1.97E+3	1.95E+3
	Std.	24.107	8.9355	9.9872	29.843	29.804	32.777	14.764	27.303	31.868
20	mean	**2.24E+3**	2.44E+3	2.47E+3	3.26E+4	3.01E+4	6.84E+3	3.20E+4	1.53E+4	7.66E+3
	Std.	62.991	81.363	104.36	13113	11072	2981.9	13541	6565	2144.4
21	mean	1.14E+5	**3.76E+4**	2.25E+5	1.16E+6	8.74E+5	6.02E+5	5.06E+6	1.60E+6	4.37E+5
	Std.	59822	18089	1.24E+5	6.04E+5	3.42E+5	6.47E+5	2.57E+6	1.42E+6	1.22E+5
22	mean	2.98E+3	2.83E+3	**2.81E+3**	3.41E+3	3.44E+3	3.42E+3	3.56E+3	2.89E+3	3.73E+3
	Std.	238.31	277.37	282.2	336.82	313.88	435.87	278.65	282.98	333.65

5.1.4 Composition faction (Fn23 to Fn30)

In composite functions, the capability of exploration and exploitation of an algorithm can be benchmarked concurrently due to having many local optima contained in the test functions [9]. Table 6 shows, CKO and ssSKF record the highest-ranking algorithm in the current domain with performances of two out of the eight functions. The performance of the CKO algorithm in Fn23 and Fn26 obtained the best solution compared to the other algorithms. Although, CKO outperforms by other algorithms in, the CKO still very closed to the best solution of the first rank algorithm.

Table 6
Comparison of different methods in composition function

Fn		CKO	SAFIRO	ssSKF	SKF	ASKF	PSO	GA	GWO	BH
23	mean	**2.64E+3**	2.64E+3	2.65E+3	2.65E+3	2.65E+3	2.66E+3	2.72E+3	2.71E+3	2.65E+3
	Std.	0.01850	0.1023	1.0797	2.1333	1.5916	5.6322	21.107	28.104	0.46051
24	mean	2.66E+3	2.68E+3	2.68E+3	2.66E+3	2.67E+3	2.67E+3	2.78E+3	**2.60E+3**	2.67E+3
	Std.	6.6821	5.1387	4.7791	5.3597	5.2331	8.2314	9.645	0	8.1356
25	mean	2.71E+3	2.71E+3	**2.71E+3**	2.73E+3	2.73E+3	2.73E+3	2.76E+3	2.72E+3	2.75E+3
	Std.	2.5899	2.6363	1.9487	4.1366	4.431	4.2891	9.6616	7.4823	9.7738
26	mean	**2.70E+3**	2.78E+3	2.71E+3	2.79E+3	2.78E+3	**2.70E+3**	2.70E+3	2.78E+3	2.80E+3
	Std.	0.10365	37.58	19.84	45.326	38.754	0.09038	0.57343	40.184	19.721
27	mean	3.54E+3	**3.44E+3**	3.51E+3	3.87E+3	3.89E+3	3.88E+3	4.49E+3	3.71E+3	4.64E+3
	Std.	94.982	160.03	92.329	123.2	112.44	161.89	78.061	107.55	265.11
28	mean	4.71E+3	4.86E+3	**4.59E+3**	7.00E+3	7.06E+3	9.70E+3	6.35E+3	4.71E+3	1.12E+4
	Std.	488.32	665.87	289.32	1102.4	990.41	1732.4	477.94	419.26	1148.3
29	mean	1.05E+4	3.03E+4	8.13E+4	**6.76E+3**	9.64E+3	2.08E+4	6.85E+6	2.18E+6	1.03E+4
	Std.	3298.7	10180	21333	6195.2	33237	85171	3.86E+6	5.02E+6	1178.1
30	mean	1.84E+4	3.61E+4	4.67E+4	1.97E+4	**1.81E+4**	1.68E+5	1.82E+5	1.02E+5	5.68E+4
	Std.	3052.3	7411.1	10560	3646.1	2709.9	80774	76669	54161	3977.1

5.1.5 Friedman and post hoc test

The Friedman test ranks is used to analyse the performance of the CKO algorithm against eight other algorithms based on mean fitness values over 50 runs across 30 benchmark functions. The average rank for each algorithm is calculated, with lower values indicating better performance. Table 7 and Figure 4 reveal that CKO outperforms SAFIRO, ASKF, ssSKF, SKF, PSO, BH, GWO, and GA, ranking

first. The Friedman statistic considers reduction performance with a chi-square value of 86.555556 and 8 degrees of freedom.

Table 7
Average ranking of the algorithm

Algorithm	Average Ranking
CKO	1.9500
SAFIRO	3.5000
ASKF	4.5500
ssSKF	4.7000
SKF	4.8333
PSO	5.6333
BH	5.9333
GWO	6.2833
GA	7.6167

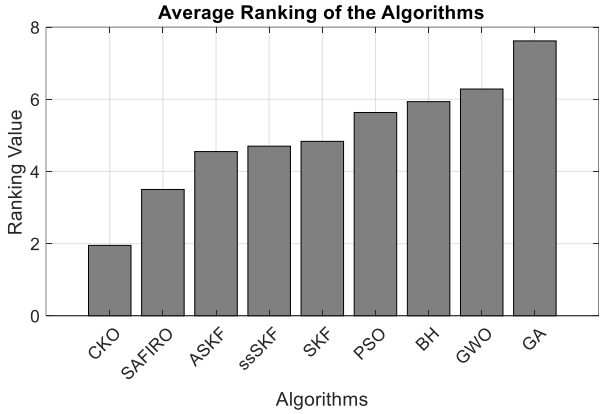


Fig. 4. Average ranking of the algorithm

Based on the Friedman test performed, significant differences are observed between the algorithms. Hence, the null hypothesis is rejected, and further analysis with a post hoc test [29] (using Holm's method) is performed to determine whether the CKO algorithm is better than the other algorithm or vice versa. The result of the Holm post hoc test with significance level $\alpha = 0.05$ are tabulated in Table 8 shows that the proposed CKO algorithm performed significantly better than the other eight algorithms (without an unadjusted p-value).

Table 8
Holm post hoc result of β value for α = 0.05

i	Comparison	z	P	Holm
8	CKO vs GA	8.013877	0	0.00625
7	CKO vs GWO	6.128259	0	0.007143
6	CKO vs BH	5.633284	0	0.008333
5	CKO vs PSO	5.20902	0	0.01
4	CKO vs SKF	4.077649	0.000045	0.0125
3	CKO vs ssSKF	3.889087	0.000101	0.016667
2	CKO vs ASKF	3.676955	0.000236	0.025
1	CKO vs SAFIRO	2.192031	0.028377	0.05

5.2 Convergence Behaviour

Convergence curves were generated to observe the ability of CKO and other algorithms to reach an optimal or a near-optimal solution in the optimization process. In this case, each convergence curve shows the mean fitness of the best solution against 10,000 iterations over 51 runs. Statistically, CKO demonstrates a very good result by leading in 14 out of 30 functions in solving the CEC2014 Benchmark test Suite. Among the functions that CKO rank first, Fn2, Fn10, Fn20, and Fn23 functions were selected to visualize the convergence curve of unimodal, simple multimodal, hybrid and compositions, respectively.

5.2.1 Convergence curve and boxplot (Fn2, Fn10, Fn20, Fn23)

Figure 5 shows that the mean fitness value of CKO (dashed black line) changes gradually at the start of the iteration, then plateaus until the end. This suggests that the search agent transitions from exploration to exploitation before stopping the process when it reaches the maximum iteration to find the best solution. This CKO's convergence pattern falls under the moderate category, striking a balance between exploration and exploitation. The CKO algorithm yielded the best results for Fn2, Fn10, Fn20, and Fn23 with mean fitness values of 5855, 1233, 2242, and 2644, respectively.

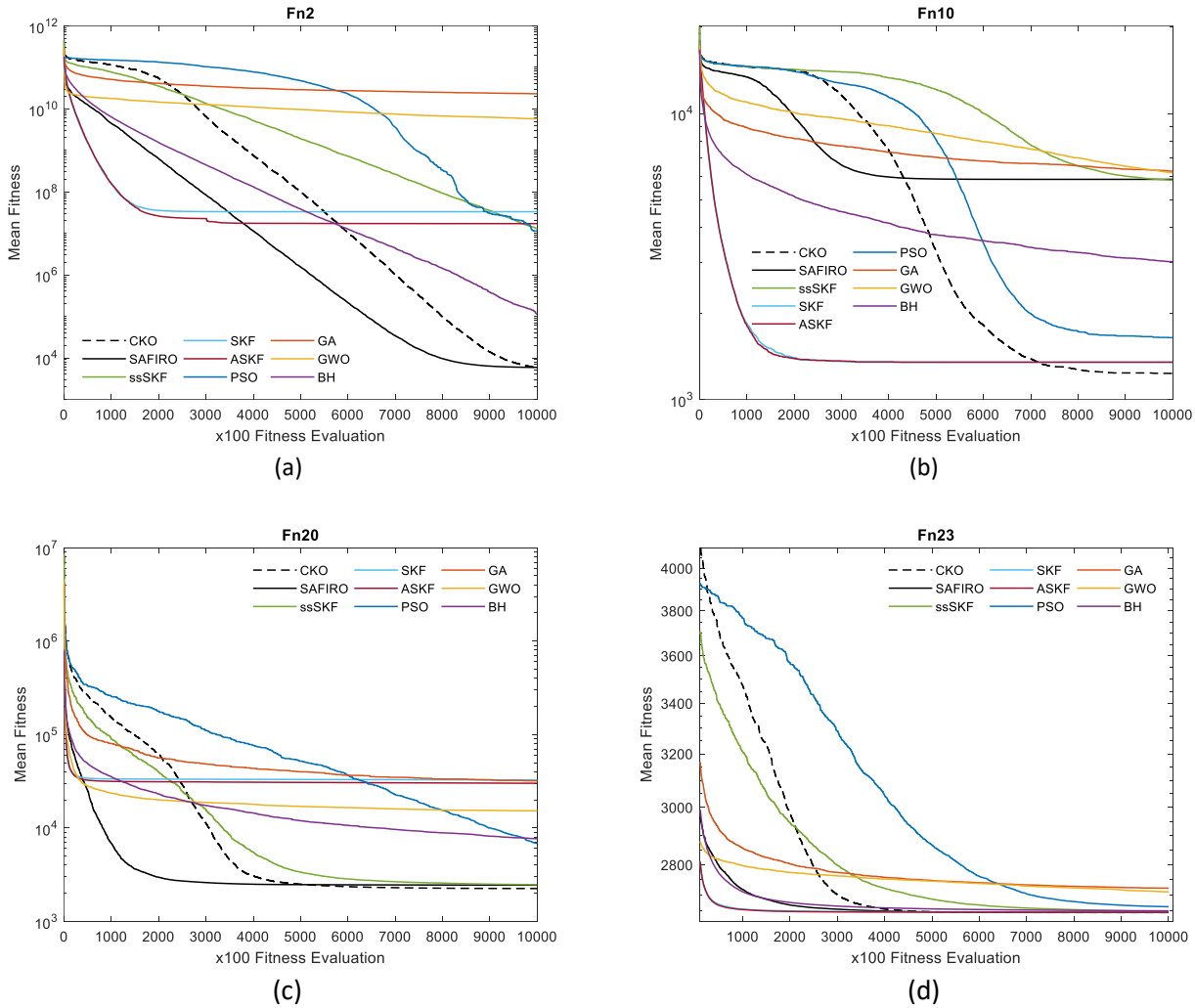


Fig. 5. Convergence curve result comparison for benchmark function (a) Unimodal (Fn2) (b) Simple multimodal (Fn10) (c) Hybrid (Fn20) (d) Composition (Fn23)

Generally, CKO still performs competitively compared to SAFIRO, ssSKF, SKF, ASKF, PSO, GA, GWO, and BH algorithms. Based on the boxplot pattern in Figure 6, the CKO algorithm has very good and consistent performance depicted by the lowest point and small derivation from the median value in solving the same benchmark problem over 50 runs.

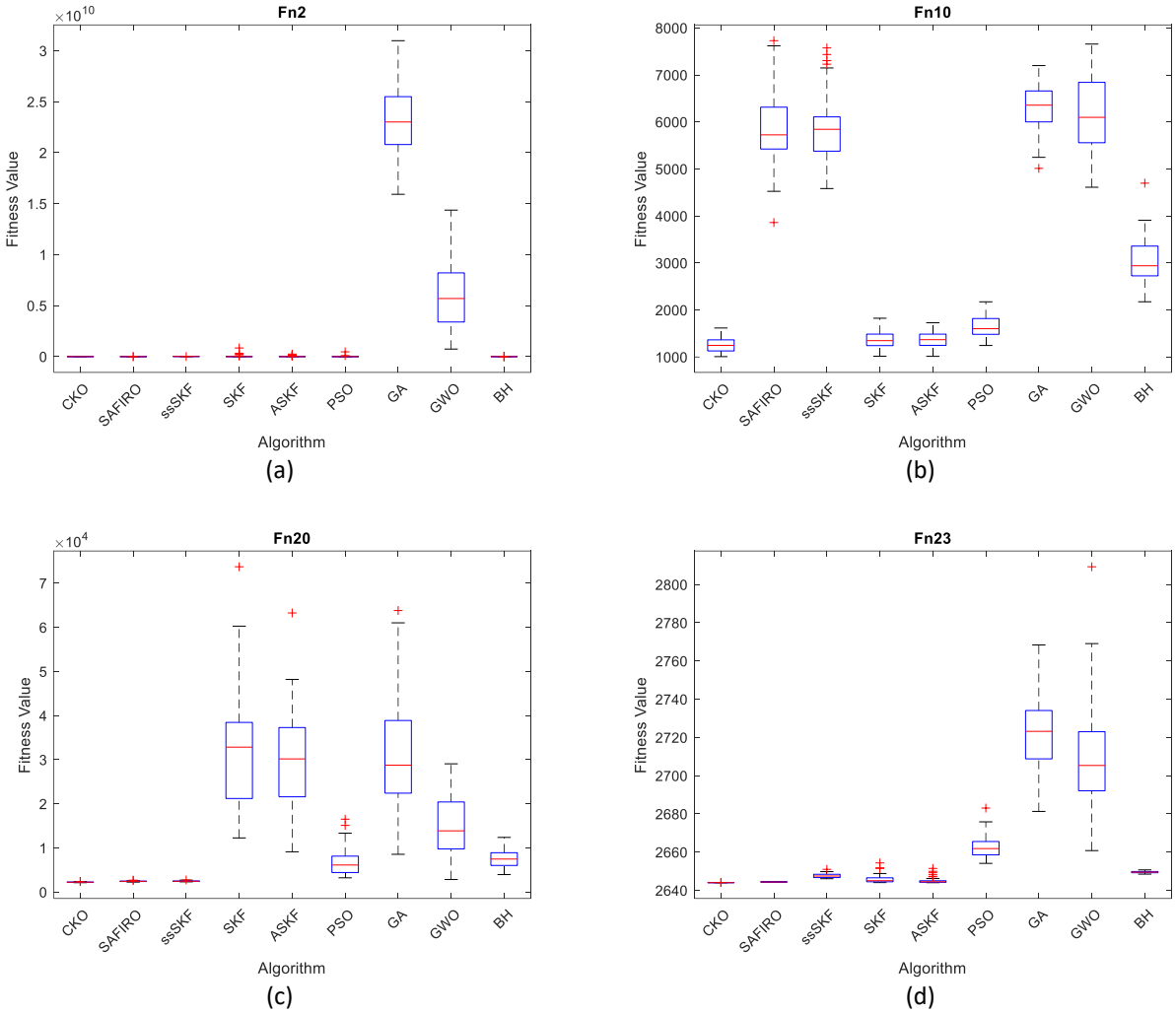
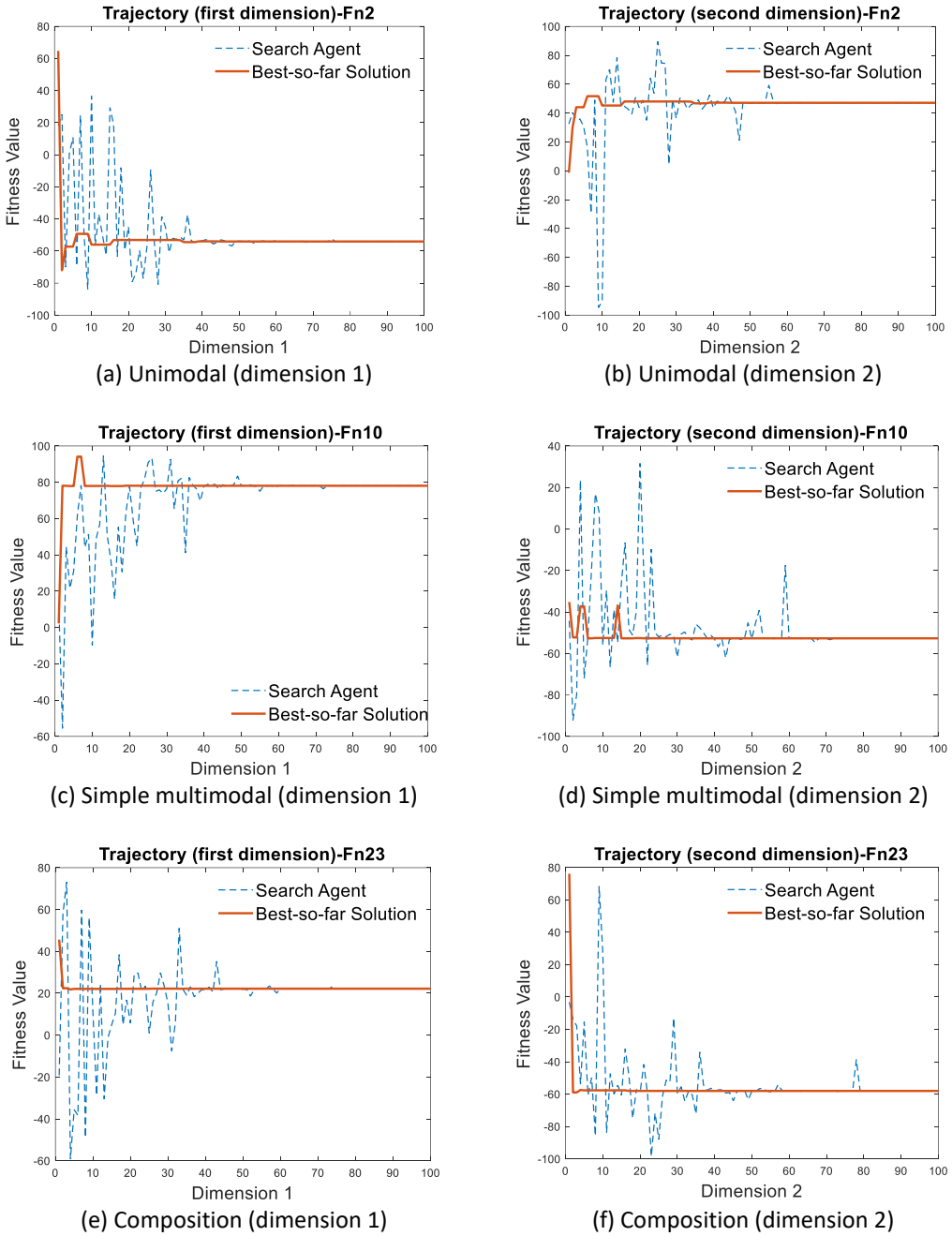


Fig. 6. Boxplot result comparison for benchmark function (a) Unimodal (Fn2) (b) Simple multimodal (Fn10) (c) Hybrid (Fn20) (d) Composition (Fn23)

5.2.2 The trajectory of CKO agent (Fn2, Fn10, Fn23)

Figure 7 illustrates CKO's search agent trajectory for one and two-dimensional solutions, with each dimension generating two graphs showing the agent's solution against the best-so-far solution. CKO's search agent displayed an exploration process during the initial phase of optimization with noticeable movement changes, followed by slow changes that confirmed the exploitation process. Finally, the plateau graph at the end of the iteration indicated CKO's search agent found a near-optimal solution before reaching the maximum iteration.



(a) Unimodal (dimension 1) (b) Unimodal (dimension 2)

(c) Simple multimodal (dimension 1) (d) Simple multimodal (dimension 2)

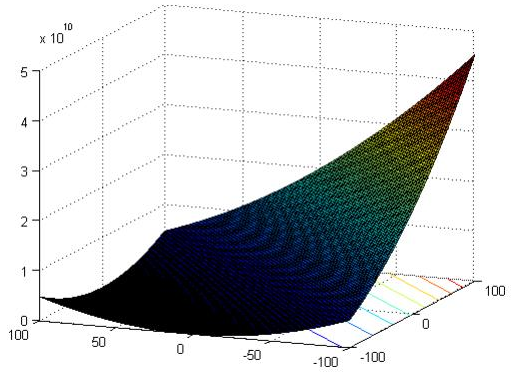
(e) Composition (dimension 1) (f) Composition (dimension 2)

Fig. 7. Experiment results comparison for Fn2, Fn10, and Fn23

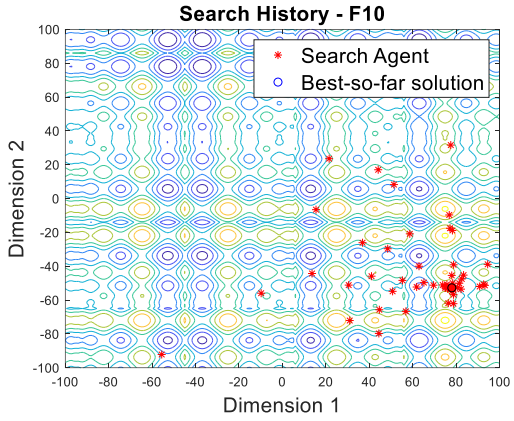
5.2.3 Search history of CKO agent (Fn2, Fn10, Fn23)

Next, the search history of the agent is also traced to observe the capability of CKO's agent in exploring and exploiting the search space, in finding the best solution. Mobilities of CKO's agent are marked and plotted on the contour map for two-dimensional functions of the selected unimodal, multimodal, and composition problems as illustrated in Figure 8. The star (*) symbol indicates the locations visited by the agent which represents the solution for each iteration. The circle (º) in each figure represents the final best solution ($X\_best\_so\_far$). It is shown that CKO's agent can adequately explore promising areas of the search space and then exploit the best solution. As the iteration increases, the agent moves towards the best-so-far solution. This is due to the reduction in the local neighbourhood radius, δ, as previously mentioned in Eq. (17).

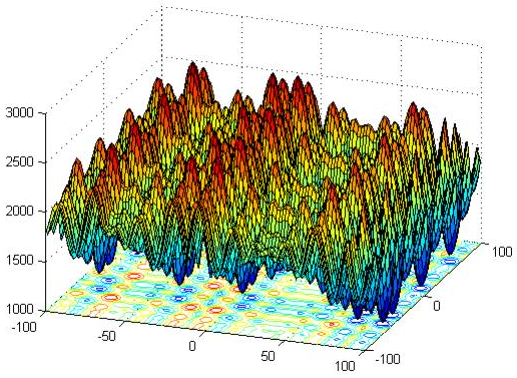
(a) Unimodal (Fn2)



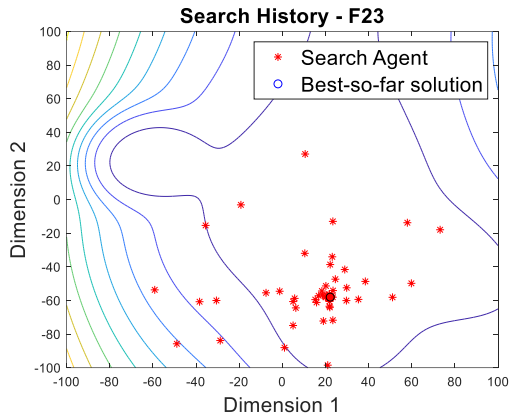
(b) 3-*D* map for 2-*D* functions of Fn2



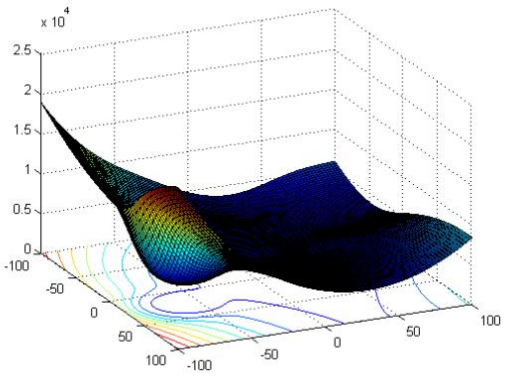
(c) Simple multimodal (Fn10)



(d) 3-*D* map for 2-*D* functions of Fn10



(e) Composition (Fn23)



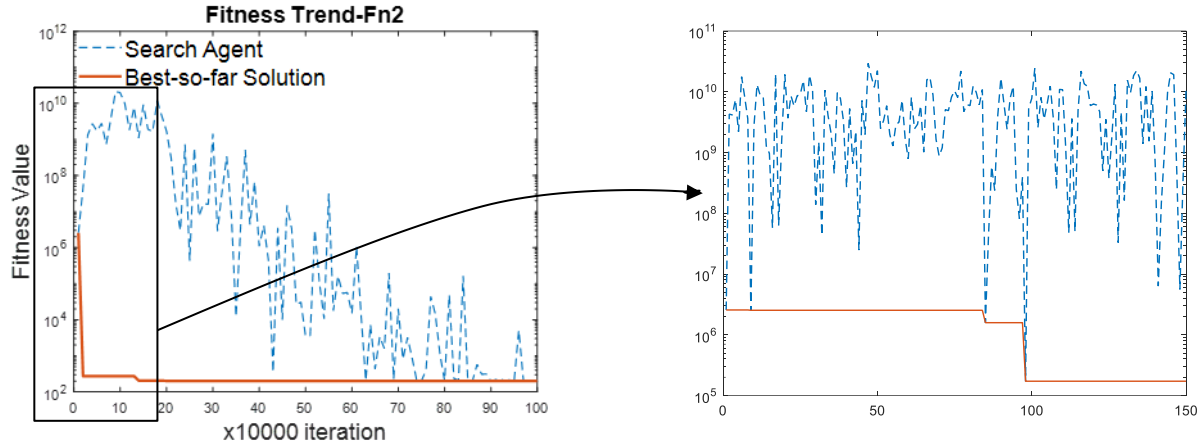
(f) 3-*D* map for 2-*D* functions of Fn23

Fig. 8. Search history of the CKO's agent during optimization of the benchmark function: Fn2, Fn10 and Fn23

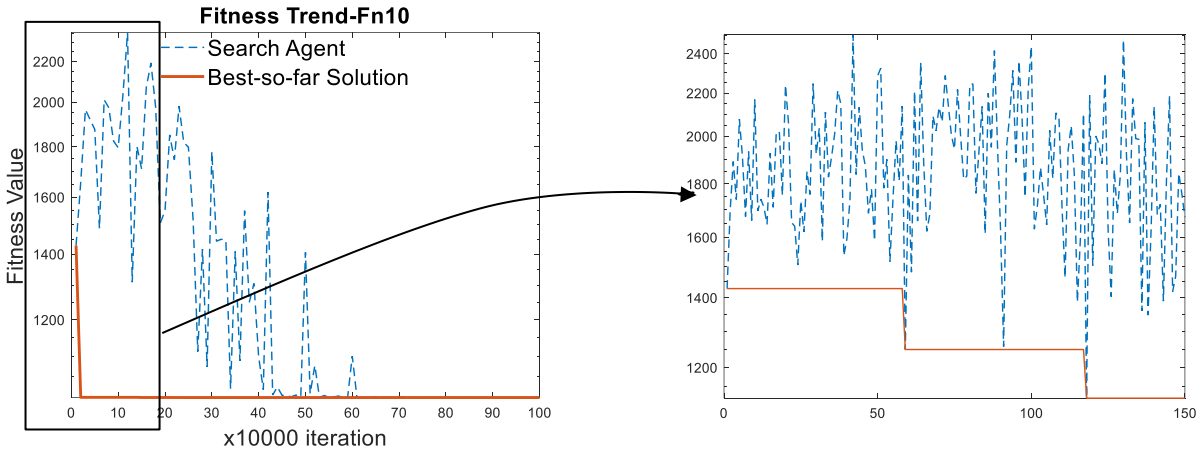
5.2.4 Fitness trends of CKO agent (Fn2, Fn10, Fn23)

Lastly, the fitness trend is generated to observe the pattern of the agent's solution against the best solution so far ($X\_best\_so\_far$) over 1 run. Based on Figure 9, it can be concluded that CKO's agent went through sufficient exploration at the beginning of the search and decline slowly to have better exploitation around the improved best-so-far solution for unimodal, simple multimodal, and

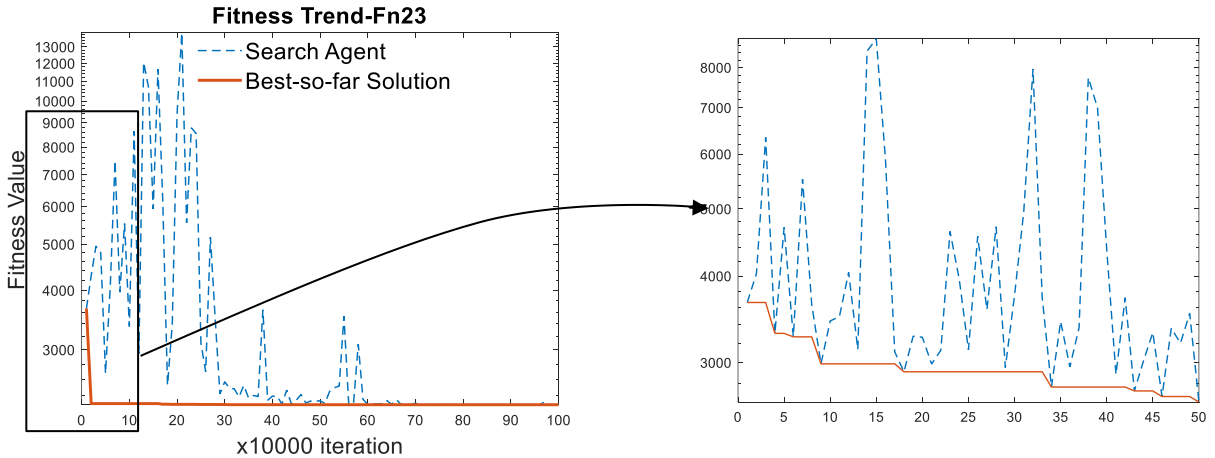
composition functions. This pattern ensures that the CKO algorithm eventually converges to the near-optimal solution in the search area.



(a) Unimodal (Fn2)



(b) Simple multimodal (Fn10)



(c) Composition (Fn23)

Fig. 9. Fitness trend of the CKO's agent during optimization of the benchmark function: Fn2, Fn10 and Fn23

Overall, CKO performs very well in solving simple multimodal functions and demonstrates a highly competitive performance in solving unimodal, hybrid, and composition functions. In the simple multimodal functions, the CKO manage to achieve first rank for 10 functions out of 13 functions. Although CKO only lead four functions out of 17 functions in unimodal, hybrid, and composition function, CKO is still very close to the optimal solution (near-optimal solution). The CKO shows a good ability to reach the optimal and near-optimal solution with a better number of best mean fitness values compared to other algorithms. These results prove that the CKO algorithm manages to iteratively estimate the optimal and near-optimal solution for varieties of optimization problems (unimodal, multimodal, hybrid, and composition functions) with different complexities. Despite working with a single agent, CKO is capable to outperform population-based algorithms under the same number of function evaluations.

6. Conclusions

This paper introduces a new single-agent metaheuristic optimization algorithm inspired by the estimation ability of CKF, named the Cubature Kalman Optimizer (CKO) algorithm. The CKO algorithm aims to find an estimate of an optimal or a near-optimal solution for an optimization problem. The proposed CKO algorithm has six steps: (1) initialization, (2) fitness evaluation and update $X\_best\_so\_far$, (3) solution prediction, (4) simulate measurement, (5) measurement prediction, and (6) solution update. The first three processes are similar to most metaheuristic algorithms. The solution prediction, simulated measurement (simulate observation signal), measurement prediction and solution update are adopted from the CKF. To evaluate the performance of the CKO algorithm, the CEC 2014 Benchmark Test Suite has been applied. Experimental results indicate that CKO is capable to converge to an optimal and a near-optimal solution and significantly outperforms well-known algorithms such as SAFIRO, ssSKF, SKF, ASKF, PSO, GA, GWO, and BH in solving the CEC 2014 benchmark problems. It is important to notice that the proposed CKO algorithm contributes to new knowledge and provides a platform for other researchers to explore, modify or hybrid this algorithm with other algorithms to produce a better metaheuristic algorithm for solving optimization problems. The CKO algorithm can also be used by other researchers to solve optimization problems in various fields.

Acknowledgement

This research was funded by a grant from TNB Research Sdn. Bhd. and Universiti Teknologi Malaysia (Grant R.K130000.7643.4C425).

References

[1] Talbi, El-Ghazali. *Metaheuristics: from design to implementation*. John Wiley & Sons, 2009.
[2] Holland, John H. "Genetic algorithms and adaptation." *Adaptive control of ill-defined systems* (1984): 317-333. https://doi.org/10.1007/978-1-4684-8941-5_21
[3] Bahdad, Ali Ahmed Salem, and Sharifah Fairuz Syed Fadzil. "Design Optimization for Light-Shelves with Regard to Daylighting Performance Improvements in The Tropics." *Journal of Advanced Research in Fluid Mechanics and Thermal Sciences* 100, no. 3 (2022): 35-50. https://doi.org/10.37934/arfmts.100.3.3550
[4] Kumar, Jayadeep, and James Varghese. "Area Optimization of a Flat Plate Collector for Meeting the Air Conditioning Load of an Office Building Using a Solar Vapor Absorption System." *Journal of Advanced Research in Fluid Mechanics and Thermal Sciences* 92, no. 1 (2022): 134-148. https://doi.org/10.37934/arfmts.92.1.134148
[5] Kenneth, Storn R., and Rainer M. Storn. "Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces." *Journal of Global Optimization* 11, no. 4 (1997): 341-359. https://doi.org/10.1023/A:1008202821328
[6] Rechenberg, Ingo. "Evolutionsstrategien." In *Simulationsmethoden in der Medizin und Biologie: Workshop,*

Hannover, 29. Sept.–1. Okt. 1977, pp. 83-114. Springer Berlin Heidelberg, 1978. https://doi.org/10.1007/978-3-642-81283-5_8

[7] Nakib, Amir, Bernard Thibault, and Patrick Siarry. "Bayesian based metaheuristic for large scale continuous optimization." In *2015 IEEE International Parallel and Distributed Processing Symposium Workshop*, pp. 314-322. IEEE, 2015. https://doi.org/10.1109/IPDPSW.2015.150

[8] Okwu, Modestus O., Lagouge K. Tartibu, Modestus O. Okwu, and Lagouge K. Tartibu. "Particle swarm optimisation." *Metaheuristic Optimization: Nature-Inspired Algorithms Swarm and Computational Intelligence, Theory and Applications* (2021): 5-13. https://doi.org/10.1007/978-3-030-61111-8_2

[9] Mirjalili, Seyedali, Seyed Mohammad Mirjalili, and Andrew Lewis. "Grey wolf optimizer." *Advances in engineering software* 69 (2014): 46-61. https://doi.org/10.1016/j.advengsoft.2013.12.007

[10] Dhiman, Gaurav, Meenakshi Garg, Atulya Nagar, Vijay Kumar, and Mohammad Dehghani. "A novel algorithm for global optimization: rat swarm optimizer." *Journal of Ambient Intelligence and Humanized Computing* 12 (2021): 8457-8482. https://doi.org/10.1007/s12652-020-02580-0

[11] Braik, Malik Shehadeh. "Chameleon Swarm Algorithm: A bio-inspired optimizer for solving engineering design problems." *Expert Systems with Applications* 174 (2021): 114685. https://doi.org/10.1016/j.eswa.2021.114685

[12] Hatamlou, Abdolreza. "Black hole: A new heuristic optimization approach for data clustering." *Information sciences* 222 (2013): 175-184. https://doi.org/10.1016/j.ins.2012.08.023

[13] Mirjalili, Seyedali. "SCA: a sine cosine algorithm for solving optimization problems." *Knowledge-based systems* 96 (2016): 120-133. https://doi.org/10.1016/j.knosys.2015.12.022

[14] Azizi, Mahdi, Siamak Talatahari, and Agathoklis Giaralis. "Optimization of engineering design problems using atomic orbital search algorithm." *IEEE Access* 9 (2021): 102497-102519. https://doi.org/10.1109/ACCESS.2021.3096726

[15] Kaveh, Ali, and Vahid Reza Mahdavi. "Colliding bodies optimization: a novel meta-heuristic method." *Computers & Structures* 139 (2014): 18-27. https://doi.org/10.1016/j.compstruc.2014.04.005

[16] Gandomi, Amir H. "Interior search algorithm (ISA): a novel approach for global optimization." *ISA transactions* 53, no. 4 (2014): 1168-1183. https://doi.org/10.1016/j.isatra.2014.03.018

[17] Sadollah, Ali, Ardeshir Bahreininejad, Hadi Eskandar, and Mohd Hamdi. "Mine blast algorithm: A new population based algorithm for solving constrained engineering optimization problems." *Applied Soft Computing* 13, no. 5 (2013): 2592-2612. https://doi.org/10.1016/j.asoc.2012.11.026

[18] Toscano, Rosario, and Patrick Lyonnet. "Heuristic Kalman algorithm for solving optimization problems." *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 39, no. 5 (2009): 1231-1244. https://doi.org/10.1109/TSMCB.2009.2014777

[19] Ibrahim, Zuwairie, Nor Hidayati Abdul Aziz, Nor Azlina Ab Aziz, Saifudin Razali, and Mohd Saberi Mohamad. "Simulated Kalman filter: a novel estimation-based metaheuristic optimization algorithm." *Advanced Science Letters* 22, no. 10 (2016): 2941-2946. https://doi.org/10.1166/asl.2016.7083

[20] Abdul Aziz, Nor Hidayati, Zuwairie Ibrahim, Nor Azlina Ab Aziz, Mohd Saberi Mohamad, and Junzo Watada. "Single-solution Simulated Kalman Filter algorithm for global optimisation problems." *Sādhanā* 43 (2018): 1-15. https://doi.org/10.1007/s12046-018-0888-9

[21] Aziz, Nor Azlina Ab, Zuwairie Ibrahim, NH Abdul Aziz, and Tasiransurini Ab Rahman. "Asynchronous simulated Kalman filter optimization algorithm." *International Journal of Engineering and Technology (UAE)* 7, no. 4.27 (2018): 44-49. https://doi.org/10.14419/ijet.v7i4.27.22478

[22] Ab Rahman, Tasiransurini, Zuwairie Ibrahim, Nor Azlina Ab Aziz, Shunyi Zhao, and Nor Hidayati Abdul Aziz. "Single-agent finite impulse response optimizer for numerical optimization problems." *IEEE Access* 6 (2018): 9358-9374. https://doi.org/10.1109/ACCESS.2017.2777894

[23] Wolpert, David H., and William G. Macready. *No free lunch theorems for search*. Vol. 10. Technical Report SFI-TR-95-02-010, Santa Fe Institute, 1995.

[24] Ito, Kazufumi, and Kaiqi Xiong. "Gaussian filters for nonlinear filtering problems." *IEEE transactions on automatic control* 45, no. 5 (2000): 910-927. https://doi.org/10.1109/9.855552

[25] Liang, Jing J., Bo Y. Qu, and Ponnuthurai N. Suganthan. "Problem definitions and evaluation criteria for the CEC 2014 special session and competition on single objective real-parameter numerical optimization." *Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore* 635, no. 2 (2013).

[26] Eberhart, Russ C., and Yuhui Shi. "Comparing inertia weights and constriction factors in particle swarm optimization." In *Proceedings of the 2000 congress on evolutionary computation. CEC00 (Cat. No. 00TH8512)*, vol. 1, pp. 84-88. IEEE, 2000.

[27] Haupt, Randy L., and Sue Ellen Haupt. *Practical genetic algorithms*. John Wiley & Sons, 2004. https://doi.org/10.1002/0471671746

[28] Alcalá-Fdez, Jesús, Luciano Sanchez, Salvador Garcia, Maria Jose del Jesus, Sebastian Ventura, Josep Maria Garrell,

José Otero et al. "KEEL: a software tool to assess evolutionary algorithms for data mining problems." *Soft Computing* 13 (2009): 307-318. https://doi.org/10.1007/s00500-008-0323-y

[29] Derrac, Joaquín, Salvador García, Daniel Molina, and Francisco Herrera. "A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms." *Swarm and Evolutionary Computation* 1, no. 1 (2011): 3-18. https://doi.org/10.1016/j.swevo.2011.02.002