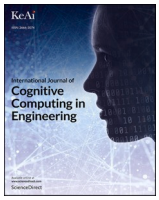
International Journal of Cognitive Computing in Engineering 5 (2024) 140–152

Contents lists available at ScienceDirect

International Journal of Cognitive Computing in Engineering

journal homepage: www.keaipublishing.com/en/journals/international-journal-of-cognitive-computing-in-engineering/



Multi-Agent cubature Kalman optimizer: A novel metaheuristic algorithm for solving numerical optimization problems

Zulkifli Musa [a,c], Zuwairie Ibrahim [b,*], Mohd Ibrahim Shapiai [c]

[a] Faculty of Electrical and Electronics Engineering Technology, Universiti Malaysia Pahang, Pekan 26600, Malaysia
[b] Faculty of Manufacturing, Universiti Malaysia Pahang, Pekan 26600, Malaysia
[c] Malaysia-Japan International Institute of Technology, Universiti Teknologi Malaysia, Kuala Lumpur 54100, Malaysia

ARTICLE INFO

Keywords:
Optimization
Metaheuristic
CKF
Local search neighborhood

ABSTRACT

Optimization problems arise in diverse fields such as engineering, economics, and industry. Metaheuristic algorithms, including the Simulated Kalman Filter (SKF), have been developed to solve these problems. SKF, inspired by the Kalman Filter (KF) in control engineering, requires three parameters (initial error covariance $P(0)$, measurement noise Q, and process noise R). However, studies have yet to focus on tuning these parameters. Furthermore, no significant improvement is shown by the parameter-less SKF (with randomized $P(0)$, Q, and R). Randomly choosing values between 0 and 1 may lead to too small values. As an estimator, KF raises concerns with excessively small Q and R values, which can introduce numerical stability issues and result in unreliable outcomes. Tuning parameters for SKF is a challenging and time-consuming task. The Multi-Agent Cubature Kalman Filter (MACKO), inspired by the Cubature Kalman filter (CKF), was introduced in this work. The nature of the Cubature Kalman filter (CKF) allows the use of small values for parameters $P(0)$, Q, and R. In the MACKO algorithm, Cubature Transformation Techniques (CTT) are employed. CTT can use small values for parameters $P(0)$, Q, and R, so CKF was developed to overcome KF and other estimation algorithms. Moreover, in CTT, the term local neighborhoods is used to propagate the cubature point in local search, where the radius, δ, of local search is updated in every iteration to balance between the exploration and exploitation processes. MACKO is evaluated on the CEC 2014 benchmark suite with 30 optimization problems, and its performance is compared with nine existing metaheuristic algorithms. Simulation results demonstrate that MACKO is superior, outperforming the benchmark algorithms, as indicated by Friedman's test with a 5 % significance level.

1. Introduction

Optimization methodologies play an afflictive role in tackling complex real-world design challenges, encountering heightened intricacies attributable to factors such as discontinuities, inadequate data, dynamicity, and uncertainty. Conventional optimization techniques grounded in mathematical principles often grapple with escalating complexity, resulting in exponential time complexities and struggles to identify optimal solutions. In response to these constraints, the effectiveness of metaheuristic optimization algorithms has attracted considerable attention. These algorithms present derivative-free methodologies characterized by straightforward structures proficient in navigating and circumventing local optima. Metaheuristic classifications encompass distinctions such as local search versus global search, single objective versus multi-objective, and population-based versus trajectory-based, rooted in their sources of inspiration.

A relatively recent addition to the metaheuristic algorithm landscape involves those inspired by estimation algorithms. In this paradigm, agents or optimizers are problem solvers tasked with discovering optimal or near-optimal solutions. One noteworthy algorithm, the Simulated Kalman Filter (SKF), emerged in 2016 as a population-based metaheuristic inspired by the Kalman Filter's capabilities. While the original SKF employs constant values for its three parameters, the parameter-less version randomizes $P(0)$, Q, and R. However, this approach yielded little improvement, with both versions exhibiting comparable performances on the CEC 2014 benchmark test suite. The random selection of values between 0 and 1, an essential aspect of parameter-less SKF, is discouraged due to potential issues with numerical stability, impeding the algorithm's capacity to identify optimal solutions.

* Corresponding author.
E-mail address: zuwairie@ump.edu.my (Z. Ibrahim).

https://doi.org/10.1016/j.ijcce.2024.03.003
Received 1 March 2023; Received in revised form 2 February 2024; Accepted 7 March 2024
Available online 8 March 2024

To address these challenges, the study explores the potential of using the Cubature Kalman Filter (CKF) as a guiding strategy for metaheuristic algorithms. CKF demonstrates proficiency in handling small (0), Q, and R values, making it a suitable source of inspiration for new optimizers. In this context, a novel metaheuristic optimization algorithm inspired by CKF, named the Multi-Agent Cubature Kalman Optimizer (MACKO), is introduced. MACKO incorporates a single parameter, the adaptive coefficient value β, governing the rate at which the local search radius, δ, diminishes. Experimental results indicate that MACKO significantly surpasses SAFIRO, TLBO, ASKF, ssSKF, PSO, BH, GWO, and GA.

The paper is organized into seven sections: Section II provides an overview of metaheuristic algorithms; Section III explores the motivation behind the proposed algorithm; Section IV elucidates the MACKO algorithm; Sections V and VI present a practical evaluation and comparison with other algorithms; and Section VII concludes, summarizing the results.

2. Related works

In the domain of metaheuristics, Fig 1 presents a classification into five categories based on their sources of inspiration. The first category encompasses Evolution algorithms, featuring well-established approaches such as the Genetic Algorithm (GA) (Holland, 1984), Bayesian evolutionary algorithm (BEA) (Nakib et al., 2015), Synergistic fibroblast optimization (Dhivyaprabha et al., 2018), Physarum-inspired computational modal (Chen et al., 2019), and Barnacles mating Optimizer (BMO) (Sulaiman et al., 2020). GA, a prominent algorithm in this category, utilizes selection, crossover, and mutation to replace the weakest solution in each generation, evolving solutions based on the best solutions from particles and the overall swarm.

The second category involves Swarm intelligence algorithms, including Particle Swarm Optimization (PSO) (Kennedy & Eberhart, 1995), Salp Swarm Algorithm (SSA) (Mirjalili et al., 2017), Harris Hawk optimization (HHO) (Heidari et al., 2019), Rat Swarm Optimizer (Dhiman et al., 2021), and Chameleon Swarm Algorithm (Braik, 2021).

Physics-inspired algorithms constitute the third category, encompassing Black Hole (BH) (Hatamlou, 2013), Multi-Verse Optimizer (Mirjalili et al., 2016), Sine Cosine Algorithm (SCA) (Mirjalili, 2016), Henry Gas Solubility Optimization (Hashim et al., 2019), and Atomic Orbital Search (Azizi et al., 2021). BH within this category emulates the gravitational power of black holes and their ability to attract nearby objects.

The fourth category involves algorithms inspired by human and animal lifestyles, featuring Teaching Learning Based Optimization (TLBO) (Rao et al., 2012), Mine Blast Algorithm (MBA) (Sadollah et al., 2013), Colliding Bodies Optimization (CBO) (Kaveh & Mahdavi, 2014), Interior Search Algorithm (ISA) (Gandomi, 2014), and Gaining-Sharing Knowledge Based Algorithm (Mohamed et al., 2020). Hybrid forms, such as hybrid GA and Optimization Hidden Neurons (Yan, 2020), hybrid GA and Reinforcement Learning (Koksal et al., 2021), hybrid BMO and Artificial Neural Network (Mustaffa & Sulaiman, 2023), hybrid PSO and Reinforcement Learning (Koksal et al., 2021), and hybrid SSA and Pulse Couple Neural Networks (Kavita et al., 2022), combine several algorithms.

The fifth category, estimation-based, includes Heuristic Kalman Algorithm (HKA) (Toscano & Lyonnet, 2009), Simulated Kalman Filter (SKF) (Ibrahim et al., 2016), Single Solution Simulated Kalman Filter (ssSKF) (Abdul Aziz et al., 2018), Asynchronous Simulated Kalman Filter (ASKF) (Nor et al., 2018), and Single Agent Finite Impulse Response Optimizer (SAFIRO) (Ab Rahman et al., 2018).. The ssSKF and ASKF are variants of SKF, incorporating trajectory-based and asynchronous iteration strategies.

HKA, a population-based algorithm inspired by the Kalman Filter (KF), is distinguished by its Gaussian distribution, which is pivotal for exploration behavior. Exploration is executed by adjusting distribution parameters to converge toward a near-optimal solution with minimal variance. The operational aspects of HKA encompass the probability density function (pdf), the measurement process, and the Kalman estimator. Furthermore, HKA involves tuning three parameters: the number of points used, the number of best candidates, and the slowdown coefficient, making it user-friendly for non-experts (Toscano & Lyonnet, 2012).

In contrast to HKA, SKF employs a distributed-free setting where solutions are randomly initialized within the search space in a uniform distribution. Despite this difference, both HKA and SKF are population-based metaheuristics inspired by the Kalman Filter's (KF) capabilities. While HKA adopts two steps of KF, which consist of the measurement and estimation steps, SKF closely aligns with the KF algorithm, going through the prediction, measurement, and estimation steps in each iteration.

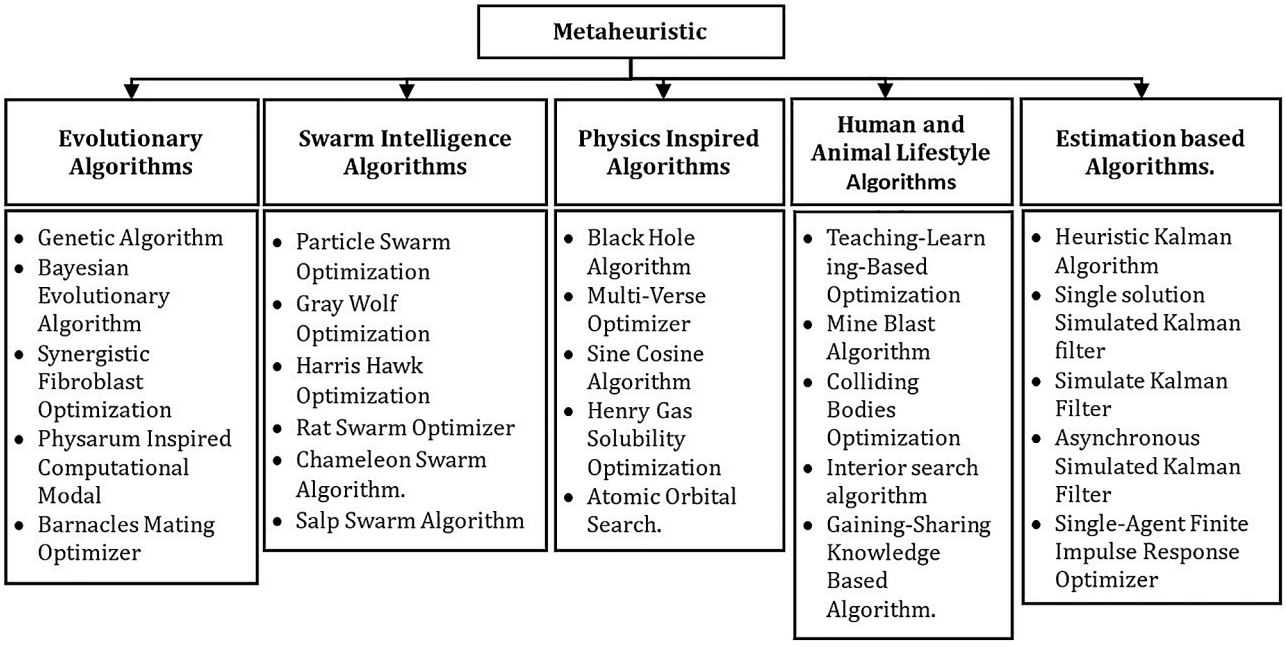


Fig. 1. Categorization of meta-heuristic algorithms.

Deviating from the KF estimator framework, the Finite Impulse Response (FIR) filter framework inspires the development of a single-solution metaheuristic optimization algorithm called the Single-Agent Finite Impulse Response Optimizer (SAFIRO). SAFIRO adopts the two steps of the UFIR framework, encompassing the measurement and estimation steps. In the measurement step, SAFIRO integrates a random mutation of the best-so-far solution and a shrinking local neighborhood method to seek a solution. As a single solution-based approach, SAFIRO operates with a single agent to find an optimal or near-optimal solution. Exploration in SAFIRO is driven by a random mutation of the best-so-far solution, while exploitation is encouraged by a shrinking local neighborhood.

HKA, SKF, and SAFIRO generally employ distinct search strategies based on their inspiration source frameworks and the number of parameters, as outlined in Table 1. Regarding the number of parameters used, HKA and SKF utilize three. HKA's three parameters include the number of points used, the number of best candidates, and the slowdown coefficient. In comparison, SKF's three parameters consist of the initial error covariance $P(0)$, measurement noise Q, and process noise R. Conversely, SAFIRO employs only two parameters in its implementation: horizontal length and coefficient value. These three algorithms demonstrate highly competitive results with a vital exploitation phase compared to other well-known metaheuristic algorithms. However, parameter tuning for HKA, SKF, and SAFIRO remains challenging and time-consuming.

Consequently, numerous studies have been conducted on these estimation-based metaheuristic algorithms, including combinations or modifications. Some examples include modified HKA for multi-objective optimization problems (Robert et al., 2018), modified SKF for binary optimization problems (Z.M. Yusof et al., 2016), modified SKF for combinatorial optimization problems (Z.M. Yusof et al., 2016), hybrid HKA and particle filter (Duong & Raghavan, 2018), hybrid SKF and PSO (Muhammad et al., 2016), and hybrid SKF and GSA (Muhammad et al., 2017). These estimation-based algorithms and their hybrid versions have also been applied to real-world problems such as job shop scheduling environments (Zhang et al., 2021), life prediction of lithium-ion batteries (Duong & Raghavan, 2018), adaptive beamforming in wireless cellular communication (Rahman et al., 2018), airport gate allocation problems (Abdul Aziz et al., 2018), and PCB drill path optimization (Aziz et al., 2017).

3. The Cubature Kalman Filter (CKF) algorithm

In control systems, the Cubature Kalman Filter (CKF) stands as an estimation system utilized for determining the actual state value by reducing the uncertainty of the observation signal. Fig 2 visually depicts the real system (upper region) alongside the CKF estimation (lower region). In the actual system, a sensor captures the measured state value. However, this measurement, denoted as $z(t)$, does not precisely reflect the actual state value, $x\_act\ (t)$, as it encompasses both process noise, $Q(t)$, and measurement noise, $R(t)$. Here, t signifies time in this real system. To address this, the CKF estimates the actual value by minimizing the measurement error, employing a three-step process: state prediction, measurement prediction, and state update.

Firstly, the state prediction step predicts the state position, $x_{k|k-1}$, and predicts the system's error covariance, $P_{k|k-1}$. In this step, two cubature points, $X_{j_{k-1|k-1}}$, are generated closest to the previous state position, $x_{k-1|k-1}$. Subsequently, the cubature points are propagated, $X_j^* k|k-1$, utilizing the state prediction function f and determining the mean of the propagated cubature points as the state-predicted position.

Secondly, the measurement prediction step anticipates the measurement position, $z_{k|k-1}$, measurement error, $Pzz_{k|k-1}$, and cross-error, $Pxz_{k|k-1}$ of the system. In this step, two cubature points, $X_{j_{k|k-1}}$, generated again closest to the state-predicted position, $x_{k|k-1}$. The cubature points are then propagated, $Z_{j_{k|k-1}}$, using the measurement prediction function h, and the mean of the propagated cubature points determines the measurement-predicted position, $z_{k|k-1}$.

Finally, in the state update step, the state position, $x_{k|k}$, and the corresponding error, $P_{k|k}$, are updated based on the predicted state position, $x_{k|k-1}$, predicted measurement position, $z_{k|k-1}$, and observation position, $z(t)$. At each iteration k, the system noise, Q_k and measurement noise, R_k, play a role in the state and measurement prediction steps.

It is important to note that the processes encompassing the generation, propagation, and computation of mean cubature points collectively constitute the Cubature Transformation Technique (CTT). Incorporating CTT into algorithms enhances both stability and accuracy, precisely capturing the mean and error aspects (Liang et al., 2014). In the realm of Cubature Kalman Filter (CKF) estimation, the variable k signifies the iteration process, where each iteration involves '$k-1|k-1$' as the preceding estimation stage, '$k|k-1$' as the prediction stage, and '$k|k$' as the update estimation stage.

The detailed process of the CKF estimator unfolds as follows:

3.1. State prediction

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

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

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

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

where the $X_{j_{k-1|k-1}}$ in (1) generated two cubature points and the $X_j^* k|k-1$ in (2) propagated two cubature points. Two cubature points indicate by $j=1$ (first cubature point) and $j=2$ (second cubature point). In (1), $\sqrt{P_{k-1|k-1}}$ is the positive weight, and $-\sqrt{P_{k-1|k-1}}$ is the negative weight. The predicted state variable, $x_{k|k-1}$ is acquired by calculating the mean

Table 1
Metaheuristic algorithm inspired by estimation system.

Optimization Algorithm	Inspiration Source	Number of solution: Individual / Population	Initialization	Search Strategy	Para. Num.	Year
Heuristic Kalman Algorithm (HKA)	Kalman filter (KF)	Population Based	Gaussian distribution	(1) Prediction (2) Measurement (3) Estimation	3	2009
Simulated Kalman Filter (SKF) algorithm	Kalman filter (KF)	Population Based	Random distribution	(1) Prediction (2) Measurement (3) Estimation	3	2016
Single-agent Finite Impulse Response Optimizer (SAFIRO) algorithm	Unbiased finite impulse response (UFIR) filter	Individual Based	Random distribution	(1) Measurement (2) Estimation	2	2018

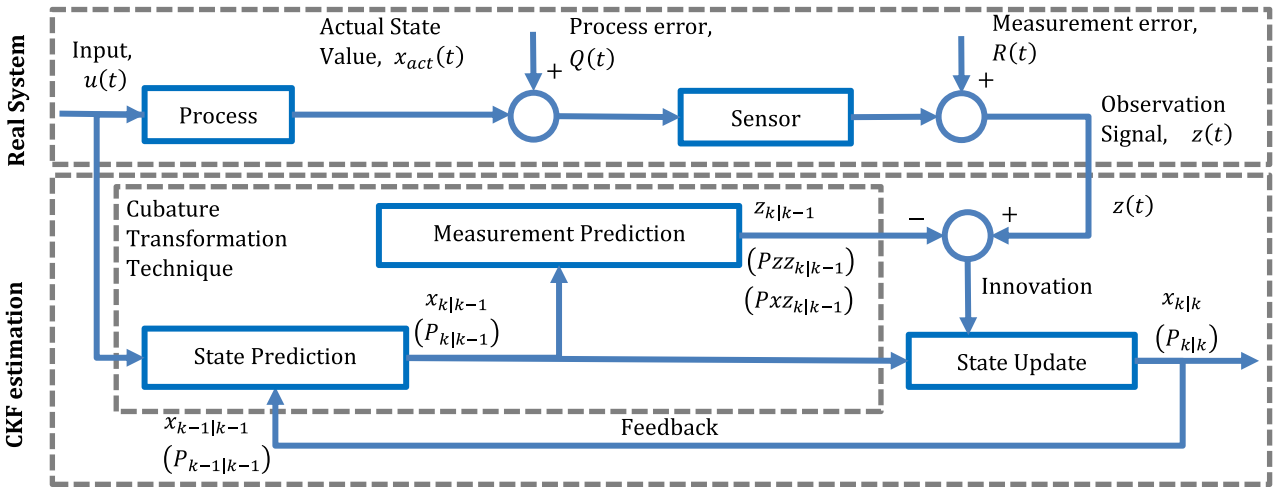


Fig. 2. The 'real system' and 'CKF estimation'.

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

$$P_{k|k-1} = \left(\left(X_1^* k|k-1 - x_{k|k-1}\right) \times \left(X_2^* k|k-1 - x_{k|k-1}\right)\right) + Q_k \tag{4}$$

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 (at $j = 1$), 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 (at $j = 2$). Then the predicted state error, $P_{k|k-1}$, is then computed by multiplying between two square root state errors ($P = \sqrt{P} \times \sqrt{P}$) along with system noise, Q_k.

3.2. Measurement prediction

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

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

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

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

where the $X_{j_{k|k-1}}$ in (5) is generated cubature points and $Z_{j_{k|k-1}}$ in (6) is the propagated cubature point. 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:

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

$$Pxz_{k|k-1} = \left(\left(X_{1k|k-1}^* - x_{k|k-1}\right) \times \left(Z_{2_k|k-1} - z_{k|k-1}\right)\right) \tag{9}$$

where $(Z_{1k|k-1} - z_{k|k-1})$ is used to collect the square root of measurement error, $\sqrt{Pzz}$ for the first cubature point (at $j = 1$), while $(Z_{2k|k-1} - z_{k|k-1})$ is used to collect the square root of measurement error, $\sqrt{Pzz}$, for the second cubature point (at $j = 2$). 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 and the square root measurement error for the second cubature point. After that, the gain, W_k is computed using (10):

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

3.3. State estimation

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

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

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

4. The Multi-Agent Cubature Kalman Optimizer (MACKO) algorithm

Optimization involves determining an objective function's minimum or maximum value within specified constraints. Mathematically, a minimization optimization problem can be represented as (13), where $f(x)$ is the objective function, $g_j(x)$ is an inequality constraint function, and $h_k(x)$ is an equality constraint function. In this context, the vector X represents design variables adjusted to achieve the optimal solution. The design space's extent is defined by upper limit x_{iU} and lower limit x_{iL} of design variables. The objective and constraint functions can be linear or non-linear, explicit or implicit.

$$Minimize: f(x) Subject\ to: g_j(x)\ \leq 0 j = 1, 2\ldots, J \tag{13}$$

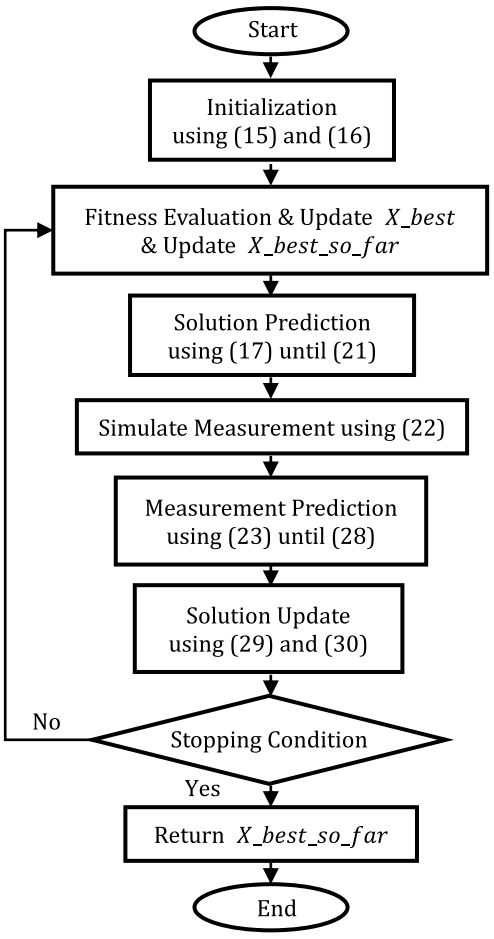
$$h_k(x) = 0 k = 1, 2, \ldots, K$$

$$x_{iL}\ \leq x_i\ \leq\ x_{iU}\ i = 1, 2, \ldots, I$$

The Multi-Agent Cubature Kalman Optimizer (MACKO) is a metaheuristic algorithm utilizing a population of agents to estimate the global minimum or maximum. Each agent, denoted as $x_i^d(t)$ in (14), holds a solution position in the search space for the i^{th} agent, d^{th} dimension, at the t^{th} iteration. The MACKO algorithm encompasses six primary phases: (1) Initialization, (2) Fitness evaluation and *X_best_so_far* update, (3) Solution prediction, (4) Simulated measurement, (5) Measurement prediction, and (6) Solution update, as illustrated in Fig 3(a).

$$x(t) = \left\{x_1^d(t), x_2^d(t), \ldots,\ x_i^d(t), \ldots, x_N^D(t)\right\} \tag{14}$$

The first two phases involve common practices in most metaheuristic algorithms. The algorithm initializes solutions, $x_i^d(0)$, and their errors,



Pseudocode: MACKO Algorithm

Initialize solution distribution: $x_i^d(0)$ using (15)
Initialize solution error: $P_i^d(0)$ using (16)
Initial the best solution: $X\_best\_so\_far = infinity$
for $t = 1 : max\_iteration$
  Fitness calculation: *fitness = calculate_fitness* $(x_i^d(0))$
  Sorting and update X_best:
  [sorted_fitness, idx] = sort (fitness)
  $X\_best = x_i^d(idk, :)$
  if $X\_best < X\_best\_so\_far$
    $X\_best\_so\_far = X\_best$
  end
  Update radius: $\delta(t)$ using (17)
  for $i = 1 : max\_agent$
    Generate cubature points: $T1_{i,j}^d(t)$ using (18)
    Propagate cubature points: $U1_{i,j}^d(t)$ using (19)
    Generate predicted solution: $xp_i^d(t)$ using (20)
    Generate predicted error: $Pp_i^d(t)$ using (21)
  end
  Simulate measurement: $z_i^d(t)$ using (22)
  for $i = 1 : max\_agent$
    Generate cubature points: $T2_{i,j}^d(t)$ using (23)
    Propagate cubature points: $U2_{i,j}^d(t)$ using (24)
    Generate predicted measurement: $zp_i^d(t)$ using (25)
    Calculate measurement error: $Pzz_i^d(t)$ using (26)
    Calculate cross error: $Pxz_i^d(t)$ using (27)
  end
  Calculate Gain: $W_i^d(t)$ using (28)
  Calculate estimated solution: $x_i^d(t+1)$ using (29)
  Calculate estimated error: $P_i^d(t+1)$ using (30)
end
return $X\_best\_so\_far$

(b)

Fig. 3. The MACKO algorithm process flow, (a) flowchart and (b) pseudocode.

$P_i^d(0)$, with random values. The fitness of an agent is calculated, and $X\_best^d(t)$ is updated based on the problem type. The $X\_best\_so\_far^d$ is updated only if it results in a better solution. If it's a minimization problem, $X\_best\_so\_far^d$ is updated if $X\_best^d(t)$ has lower fitness. For maximization problems, $X\_best\_so\_far^d$ is updated if $X\_best^d(t)$ has higher fitness.

The remaining phases (3 until 6) are adapted from the Cubature Kalman Filter (CKF). Like CKF, the MACKO algorithm employs the CTT in the solution and measurement prediction stages. The solution prediction phase generates the predicted solution, $xp_i^d(t)$, using CTT, involving the creation of two cubature points, propagating these points in the local neighbourhoods search area, and then determining the predicted solution by taking the mean value of cubature points. The predicted error, $Pp_i^d(t)$, is also calculated in this phase. Then, the simulated measurement phase introduces simulated measurements $z(t)$ into CKF. The measurement prediction phase generates the predicted measurement value, $zp_i^d(t)$, using CTT, similar to the solution prediction phase. The innovation, $Pzz_i^d(t)$, cross-error, $Pxz_i^d(t)$, and gain, $W_i^d(t)$, are also calculated. The $W_i^d(t)$ is determined based on the ratio between $Pzz_i^d(t)$ and $Pxz_i^d(t)$. The last step, the solution update phase, involves updating the solution $x_i^d(t+1)$ and its error $P_i^d(t+1)$ by combining the predicted solution with discrepancies between simulated and predicted measurements. These five phases (2 until 6) are iteratively repeated until a stopping condition is met. MACKO returns *X_best_so_far* as the final result.

Notably, in CKF, process noise, $Q(t)$, and measurement noise, $R(t)$, may have values that are too small or too large. In MACKO, both values are randomly set between 0 and 1, making MACKO a parameter-less algorithm. Using CTT in phases (3) and (5) helps prevent divergence and dimensional issues associated with excessively small values. In CKF, cubature point propagation is carried out using the state prediction functions, f, and measurement prediction functions, h. In contrast, MACKO propagates cubature points using the local neighborhood term in the local search area. The local search radius, δ, is dynamically adjusted in each iteration, starting large and gradually decreasing. An adjustable parameter controls the reduction speed, β.

The detailed process of MACKO is outlined as follows:

4.1. Initialization

The MACKO begins with the random initialization of values to its agent, $x_i^d(0)$, within the search space as (15), where $\underline{x}_d$ is the lower limit, $\overline{x}_d$ is the upper limit of the search space in the dth dimension, and ith is the number of the agent. Additionally, the initial value of the solution error, $P_i^d(0) \in [0,1]$, is generated using a random value as (16).

$$x_i^d(0) = randn_i^d + \left[\cup \left(\underline{x}_d,\ \overline{x}_d \right) \right] \tag{15}$$

$$P_i^d(0) = randn_i^d \tag{16}$$

4.2. Fitness evaluation, update

The iteration starts with determining the fitness of each solution, $x_i^d(t)$. Then, the $X\_best^d(t)$ is updated according to the type of problem. The fitness of the best solution, $X\_best^d(t)$ is compared to the fitness of $X\_best\_so\_far^d(t)$ whereby $X\_best\_so\_far^d(t)$ will be updated if the better solution ($X\_best^d(t)<X\_best\_so\_far^d(t)$ for minimization problems, or $X\_best^d(t)>X\_best\_so\_far^d(t)$ for maximization problem) is found.

4.3. Solution prediction

At first, the δ in (17) is determined, where t is the current iteration, and $tMax$ is the maximum number of iterations.

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

The following equations determine the predicted solution candidate, $xp_i^d(t)$, where $T1_{i,j}^d(t)$ in (18) is the generated cubature point, $U1_{i,j}^d(t)$ in (19) is the propagation of both cubature points randomly in the search space by using a random element, $randn_i^d \in [0,1]$.

$$T1_{i,j}^d(t) = x_i^d(t) + \left[\sqrt{P_i^d(t)} \quad -\sqrt{P_i^d(t)} \right] \tag{18}$$

$$U1_{i,j}^d(t) = T1_{i,j}^d(t) + randn_i^d(\mathrm{U}\,[-\delta,\ \delta]) \tag{19}$$

$$xp_i^d(t) = \frac{1}{2}\sum_{j=1}^{2} U1_{i,j}^d(t) \tag{20}$$

Two cubature point involved in these step is indicated by $j = 1$ (first point) and $j = 2$ (second point). Once the predicted solution candidate is calculated, the solution error, $Pp_i^d(t)$, must be predicted using (21), where $randn_i^d \in [0,1]$ is used to present the system error.

$$Pp_i^d(t) = \left(\left(U1_{i,1}^d(t) - xp_d(t)\right) \times \left(U1_{i,2}^d(t) - xp_d(t)\right)\right) + randn_i^d \tag{21}$$

4.4. Simulated measurement

The measurement step performs the role of feedback of estimation process. The measurement of each solution is simulated based on the following (22):

$$z_i^d(t) = \left(xp_i^d(t) + sin\left(randn_i^d \times 2\pi\right) \times \left|xp_i^d(t) - X\_best\_so\_far^d(t)\right|\right) \tag{22}$$

where the simulated measurement value for the agent, $z_i^d(t)$ may take any random position in a locus $|xp_i^d(t) - X\_best\_so\_far^d(t)|$. A random element, $randn_i^d \in [0,1]$ in $sin(randn_i^d \times 2\pi)$ is responsible for the stochastic aspect of the MACKO algorithm.

4.5. Measurement prediction

The predicted measurement vector, $zp_1^d(t)$ for predicted solution, $xp_1^d(t)$ is determined by the following equations:

$$T2_{i,j}^d(t) = xp_i^d(t) + \left[\sqrt{Pp_i^d(t)} \quad -\sqrt{Pp_i^d(t)} \right] \tag{23}$$

$$U2_{i,j}^d(t) = T2_{i,j}^d(t) + randn_i^d(\mathrm{U}\,[-\delta,\ \delta]) \tag{24}$$

$$zp_i^d(t) = \frac{1}{2}\sum_{j=1}^{2} U2_{i,j}^d(t) \tag{25}$$

where $T2_{i,j}^d(t)$ in (23) are the generated cubature point and $U2_{i,j}^d(t)$ in (24) is the propagation of both cubature point randomly in the search space by using a random element, $randn_i^d \in [0,1]$. Once the predicted measurement, $zp_i^d(t)$ is calculated, the measurement error, $Pzz_i^d(t)$ and the cross-error, $Pxz_i^d(t)$ need to be estimated for gain calculation. These two types of error can be obtained by using (26) and (27), respectively:

$$Pzz_i^d(t) = \left(\left(U2_{i,1}^d(t) - zp_i^d(t)\right) \times \left(U2_{i,2}^d(t) - zp_i^d(t)\right)\right) + randn_i^d \tag{26}$$

$$Pxz_i^d(t) = \left(\left(U1_{i,1}^d(t) - xp_i^d(t)\right) \times \left(U2_{i,2}^d(t) - zp_i^d(t)\right)\right) \tag{27}$$

where $randn_i^d \in [0,1]$ is measurement noise. Then, the gain, $W_i^d(t)$ can be calculated using (28):

$$W_i^d(t) = Pxz_i^d(t)/Pzz_i^d(t) \tag{28}$$

Finally, the solution, $x_i^d(t+1)$ and solution error, $P_i^d(t+1)$ are updated by the following equations:

$$x_i^d(t+1) = xp_i^d(t) + W_i^d(t)\left(z_i^d(t) - zp_i^d(t)\right) \tag{29}$$

$$P_i^d(t+1) = Pp_i^d(t) - W_i^d(t)Pzz_i^d(t) \tag{30}$$

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

Towards the end of this section, the pseudo-code for MACKO is presented in Fig 3(b), consisting of 31 lines. Lines (1), (2), and (3) represent initialization, while lines (5) to (11) handle fitness evaluation and update *X_best_so_far*. Lines (12) to (18) focus on solution prediction, line (19) involves simulated measurement, lines (20) to (27) cover measurement prediction, and lines (28) to (29) address solution prediction. The iterative process, repeating until the stopping condition is met, is indicated by lines (4) and (30). Finally, line (31) returns *X_best_so_far* as the final result.

5. Experiment setup

The MACKO algorithm underwent comprehensive testing and comparison against various algorithms, including SAFIRO, ssSKF, SKF, ASKF, PSO, GA, GWO, BH, and TLBO. The evaluation utilized the CEC 2014 benchmark suite (Liang et al., 2014), comprising 30 single-objective test functions that emulate real-world optimization problems. These functions are categorized into unimodal, simple multimodal, hybrid, and composition groups, designed to assess exploration capability, exploitation capability, complex problem-solving capability, and the balance between exploration and exploitation, respectively.

Initially, the test results were assessed across 50 trials, involving 1 million function evaluations, with complexity set to 50 dimensions. Parameters for all tested algorithms, including the sole parameter for MACKO ($\beta = 12$), were configured as per Table 2. The MATLAB code for the CEC 2014 benchmark suite is accessible at http://www.ntu.edu.sg/home/EPNSugan/index_files/CEC2014. The evaluation employed the Friedman test to rank algorithms based on performance differences at a 5 % significance level. The Holm post hoc procedure determined significant differences between algorithms. Friedman and Holm's analyses were executed using the KEEL software (Alcalá-Fdez et al., 2009). In the second experiment, a single trial evaluated MACKO with 500 function evaluations and two dimensions, aiming to investigate the behavior of its agents. The exploration and exploitation capability of each MACKO agent were scrutinized through the trajectory, search history, and fitness trend analysis.

6. Result and discussion

This section provides a practical evaluation and comparison with other algorithms.

Table 2
Parameter setting of the selected algorithms.

Reference	Algorithm	Parameter	Parameter Value
This work	Multi-Agent Cubature Kalman Optimizer	Number of agents coefficient, β	4 12
(Ab Rahman et al., 2018)	Single agent Finite Impulse Response Optimizer	Number of agents parameters (**N**) coefficient, β	1 4 5
(Abdul Aziz et al., 2018)	Single solution Simulate Kalman Filter	Number of agents coefficient, β	1 5
(Ibrahim et al., 2016)	Simulate Kalman Filter	Number of agents Error Covariance, P Process error, Q Measurement error, R	100 1000 0.5 0.5
(Nor et al., 2018)	Asynchronous Simulate Kalman Filter	Number of agents Error Covariance, P Process error, Q Measurement error, R	100 1000 0.5 0.5
(Eberhart & Shi, 2000)	Particle Swarm Optimizer	Number of agents Initial inertia weight, w_1 Final inertia weight, w_2 Cognitive acceleration, c_1 Social acceleration, c_2	100 0.9 0.5 2 2
(Ahn, 2006)	Genetic Algorithm	Population Size Crossover probability, Pc Mutation probability, Pm	100 0.5 0.2
(Mirjalili et al., 2014)	Grey Wolf Optimizer	Number of agents Adaptive parameter, a Coefficient vector, C	100 decreases [2–0] rand [0,2]
(Hatamlou, 2013)	Black Hole	Number of stars	100
(Rao et al., 2012)	Teaching Learning Based Optimization	Number of agents	50

6.1. Statistical analysis

Table 3 presents all algorithms' mean and standard deviation (std.) across unimodal, simple multimodal, hybrid, and composition benchmark functions. The optimal mean value for each objective function is denoted in **bold** among the tested algorithms. The performance ranking, determined through the Friedman test, is outlined in Table 4, while Table 5 provides the significance levels via the Holm post hoc test.

6.1.1. Exploitation verification (Fn1 to Fn3)

Unimodal test functions serve as a means to scrutinize the exploitation capability of an optimization algorithm (Mirjalili et al., 2014). In this investigation, aimed at assessing MACKO's proficiency in exploiting promising regions, three unimodal benchmark functions (Fn1, Fn2, and Fn3) were addressed, and the outcomes were juxtaposed with those of nine chosen optimization methods, as detailed in Table 3. The findings reveal that SAFIRO surpasses all other algorithms by yielding the optimal solution in Fn1, Fn2, and Fn3. Despite MACKO securing the second position after SAFIRO in Fn3 and the third position behind SAFIRO and TLBO in Fn1 and Fn2, it consistently generates solutions closely aligned with the optimal solution (near-optimal solution), particularly for Fn2 and Fn3. Notably, the results for Fn1 underscore MACKO's exceptional performance in exploiting optimal solutions, akin to the proficiency demonstrated by ssSKF, SKF, ASKF, and BH algorithms. Fn1, recognized for its quadratic ill-conditioned property (Liang et al., 2014), presents a challenging problem; nevertheless, MACKO exhibits a comparative level to SAFIRO and TLBO.

6.1.2. Exploration verification (Fn4 to Fn16)

Simple multimodal functions evaluate the MACKO exploration capability (Mirjalili et al., 2014) by minimising 13 multimodal functions (Fn4 to Fn16). The results in Table 3 (simple multimodal functions: Fn4 to Fn16) show that MACKO exhibits superior performance in handling most of these functions. In the Fn5, Fn6, Fn7, and Fn15 cases, SAFIRO showed excellent performance, and for Fn4, TLBO outperformed, delivering the optimal solution. Despite MACKO securing the second position behind SAFIRO and TLBO in these scenarios, its solutions closely approached the optimal outcomes. This commendable explorative capability of MACKO is attributed to its effective updating strategy.

6.1.3. Solving complex problem verification (Fn17 to Fn22)

In hybrid functions, variables are randomly partitioned into various subcomponents, and different basic functions are applied to each subcomponent (Liang et al., 2014). The six hybrid functions (Fn17 to Fn22) involve the fusion of several multimodal functions (Fn19, Fn21, and Fn22) or the fusion of unimodal and simple multimodal functions (Fn17, Fn18, and Fn20). This fusion increases the complexity of the functions, testing an algorithm's capability to solve intricate optimization problems. Table 3 (hybrid functions: Fn17 to Fn22) shows that, with a lead in three of the six functions, SAFIRO has the highest ranking among hybrid functions. In the meantime, the leading algorithms in Fn18 and Fn22 are ssSKF and BH, respectively. Despite only taking the lead in Fn20, MACKO regularly yields results that are extremely near to the optimal solution generated by the top-ranking algorithm, particularly for Fn18, Fn19, and Fn22. The result achieved by MACKO demonstrates how well it can handle challenging optimization challenges.

6.1.4. Balance exploration and exploitation verification (Fn23 to Fn30)

In the composition function, the global optimum is the local optimum with the smallest bias value, combining the characteristics of the sub-functions and preserving both global and local optima. This function is more intricate to solve than others, serving as a test for an algorithm's ability to balance exploration and exploitation due to multiple local optima simultaneously. Table 3 (composition function: Fn23 to Fn30) shows that TLBO is the industry leader in this area, scoring well in three of the eight functions (Fn23, Fn25, and Fn30). In contrast, GWO, PSO, SAFIRO, ssSKF, and SKF each score well in one function (Fn24, Fn26, Fn27, Fn28, and Fn29, respectively). Although others may have outperformed MACKO, it constantly generates solutions that are extremely near the best solution given by the top-ranking algorithm. The result achieved by MACKO demonstrates its ability to strike a balance between exploration and exploitation when searching for the global optimum of composition functions.

6.1.5. Friedman and post HOC test

The Friedman test compared MACKO's performance to the other eight algorithms, obtaining a chi-square value of 87.561818 with 9 degrees of freedom. The algorithms were rated based on their average fitness value, calculated from 50 runs across all 30 benchmark functions. The average rank was calculated for each method, with a lower rank indicating higher performance. As shown in Table 4, MACKO took first place, followed by SAFIRO, TLBO, ASKF, ssSKF, SKF, PSO, BH, GWO, and GA.

The Friedman test results revealed considerable differences in the performance of the algorithms. As a result, the null hypothesis was rejected, necessitating the execution of a post hoc test to determine which algorithm, MACKO or the others, performed better. The Holm post hoc test (Derrac et al., 2011), as shown in Table 5, demonstrates that the MACKO algorithm outperforms the other eight algorithms significantly, with a p-value less than the significance level of $= 0.05$.

Table 3
Comparison of different methods in solving CEC2014 test functions.

Fn		MACKO	SAFIRO	ssSKF	SKF	ASKF	PSO	GA	GWO	BH	TLBO
1	mean	2.180E+06	**4.493E+05**	4.915E+06	4.747E+06	3.760E+06	4.346E+07	3.405E+08	6.055E+07	4.362E+06	6.5475e+05
	Std.	6.605E+05	1.558E+05	1.257E+06	1.665E+06	1.440E+06	3.447E+07	8.251E+07	3.205E+07	9.394E+05	4.0339e+05
2	mean	6.893E+03	**5.876E+03**	1.296E+07	3.317E+07	1.714E+07	1.140E+07	2.304E+10	5.752E+09	1.138E+05	5.9083+03
	Std.	6230	6156.4	1.45E+06	1.30E+08	4.69E+07	6.72E+07	3.80E+09	2.95E+09	1.19E+05	7629.6
3	mean	3.027E+02	**3.000E+02**	3.663E+02	1.721E+04	1.591E+04	9.934E+03	6.113E+04	5.081E+04	1.142E+04	3.4558+02
	Std.	5.5977	9.57E-05	11.959	9185.2	6403	9628.3	12227	11252	2150.1	58.236
4	mean	5.036E+02	4.941E+02	5.028E+02	5.228E+02	5.285E+02	1.062E+03	3.114E+03	9.587E+02	5.731E+02	**4.9285+02**
	Std.	41.896	17.862	22.194	41.813	38.409	277.16	705.25	338.28	46.244	37.887
5	mean	520.00081	**520.00007**	521.13000	520.00580	520.00500	521.06000	520.99700	521.11000	520.01622	521.1100
	Std.	0.000127	9.93E-06	0.02683	0.010161	0.00934	0.059419	0.057366	0.035471	0.030956	0.03773
6	mean	6.182E+02	**6.158E+02**	6.187E+02	6.328E+02	6.310E+02	6.315E+02	6.556E+02	6.276E+02	6.567E+02	6.3787+02
	Std.	3.9018	4.5968	4.0806	3.8908	4.981	5.2396	2.5233	3.831	4.7812	4.0499
7	mean	700.0114	**700.0085**	701.1300	700.1500	700.1000	700.0200	933.3570	740.0900	700.1400	700.1200
	Std.	0.011804	0.007879	0.013132	0.19049	0.13329	0.03346	37.768	22.884	0.074243	0.25406
8	mean	**8.027E+02**	9.737E+02	9.775E+02	8.069E+02	8.073E+02	8.587E+02	1.070E+03	9.726E+02	9.219E+02	9.8169+02
	Std.	1.7197	39.117	41.158	2.6967	3.1555	12.203	19.965	29.905	14.224	23.881
9	mean	**1.043E+03**	1.088E+03	1.091E+03	1.063E+03	1.064E+03	1.052E+03	1.399E+03	1.087E+03	1.217E+03	1.091+03
	Std.	24.517	48.567	41.377	37.868	35.547	29.129	32.909	29.112	46.976	27.573
10	mean	**1.209E+03**	5.881E+03	5.839E+03	1.346E+03	1.350E+03	1.644E+03	6.286E+03	6.204E+03	3.025E+03	5.2322+03
	Std.	154.91	780.86	707.67	202.35	168.24	227.67	494.54	836.22	434.97	782.48
11	mean	**5.635E+03**	6.185E+03	6.392E+03	6.183E+03	6.146E+03	1.230E+04	1.283E+04	6.235E+03	8.108E+03	1.0753+04
	Std.	748.89	799.4	1023	686.18	783.8	2163.6	428.18	843.1	987.62	2939.4
12	mean	**1200.017**	1200.400	1200.900	1200.300	1200.200	1202.600	1202.200	1202.000	1200.700	1203.2
	Std.	0.047762	0.044021	0.37471	0.094916	0.088933	0.40237	0.34351	1.5175	0.23441	0.25254
13	mean	**1300.498**	1300.923	1300.566	1300.563	1300.549	1300.605	1302.876	1300.550	1300.551	1300.6
	Std.	0.12659	0.14115	0.10022	0.09047	0.069533	0.10881	0.38416	0.08391	0.037326	0.092926
14	mean	**1400.11**	1400.7822	1400.40	1400.32	1400.31	1400.34	1462.7360	1407.90	1400.27	1400.3
	Std.	0.19785	0.36392	0.29441	0.10574	0.065683	0.098502	9.7036	10.178	0.014087	0.10132
15	mean	1510.80	**1510.50**	1531.80	1556.7	1547.00	1528.70	3816.7440	2000.50	1775.00	1575.7
	Std.	3.1799	2.7525	3.1778	20.536	17.527	7.9526	26070	653.12	49.778	47.094
16	mean	**1619.00**	1620.60	1620.70	1619.10	1618.90	1621.60	1.622E+03	1619.40	1621.50	1620.3
	Std.	0.93463	0.80952	0.56302	0.81281	0.71454	0.68879	0.4513	0.97739	0.60571	0.59231
17	mean	1.870E+05	**2.650E+04**	3.261E+05	8.430E+05	8.352E+05	2.474E+06	1.605E+07	3.393E+06	5.529E+05	1.7323e+05
	Std.	1.06E+05	11610	1.48E+05	4.58E+05	4.81E+05	2.59E+06	7.60E+06	3.00E+06	1.94E+05	3.1364e+05
18	mean	3.440E+03	4.313E+03	3.744E+05	4.708E+06	8.853E+06	1.208E+04	5.163E+06	2.585E+07	**2.397E+03**	3.5865+03
	Std.	864.82	1471	64934	1.57E+07	3.75E+07	39429	2.63E+06	6.12E+07	250.46	1446.5
19	mean	1.937E+03	**1.920E+03**	1.923E+03	1.954E+03	1.947E+03	1.958E+03	2.005E+03	1.973E+03	1.954E+03	1.922+03
	Std.	18.457	8.9355	9.9872	29.843	29.804	32.777	14.764	27.303	31.868	6.9578
20	mean	**2.244E+03**	2.441E+03	2.468E+03	3.256E+04	3.005E+04	6.837E+03	3.197E+04	1.525E+04	7.661E+03	2.321+03
	Std.	65.612	81.363	104.36	13113	11072	2981.9	13541	6565	2144.4	108.87
21	mean	1.532E+05	**3.764E+04**	2.251E+05	1.160E+06	8.744E+05	6.020E+05	5.062E+06	1.604E+06	4.369E+05	9.8795+04
	Std.	76510	18089	1.24E+05	6.04E+05	3.42E+05	6.47E+05	2.57E+06	1.42E+06	1.22E+05	66050
22	mean	3.036E+03	2.830E+03	**2.811E+03**	3.405E+03	3.444E+03	3.420E+03	3.556E+03	2.889E+03	3.728E+03	3.0072+03
	Std.	318.52	277.37	282.2	336.82	313.88	435.87	278.65	282.98	333.65	264.46
23	mean	2644.0404	2644.4013	2647.6080	2645.5876	2645.0928	2662.4991	2722.5740	2710.8020	2649.5255	**2644.0045**
	Std.	0.015833	0.1023	1.0797	2.1333	1.5916	5.6322	21.107	28.104	0.46051	7.8391e-10
24	mean	2659.4468	2676.8444	2676.8340	2664.3281	2664.4939	2670.4950	2776.1120	**2600.0000**	2666.5915	2600.0019
	Std.	5.1581	5.1387	4.7791	5.3597	5.2331	8.2314	9.645	0	8.1356	0.00027962
25	mean	2.714E+03	2.712E+03	2.711E+03	2.730E+03	2.731E+03	2.730E+03	2.759E+03	2.724E+03	2.748E+03	**2.700+03**
	Std.	2.5428	2.6363	1.9487	4.1366	4.431	4.2891	9.6616	7.4823	9.7738	3.7499e-13
26	mean	2700.5317	2782.9000	2704.5000	2787.8000	2782.3000	**2700.5020**	2702.5000	2780.1000	2792.2000	2759.0345
	Std.	0.1146	37.58	19.84	45.326	38.754	0.090378	0.57343	40.184	19.721	49.461
27	mean	3.528E+03	**3.443E+03**	3.509E+03	3.866E+03	3.890E+03	3.883E+03	4.494E+03	3.707E+03	4.640E+03	4.1172+03
	Std.	88.84	160.03	92.329	123.2	112.44	161.89	78.061	107.55	265.11	98.643
28	mean	4.684E+03	4.858E+03	**4.590E+03**	7.000E+03	7.056E+03	9.699E+03	6.350E+03	4.712E+03	1.119E+04	5.2792+03
	Std.	283.99	665.87	289.32	1102.4	990.41	1732.4	477.94	419.26	1148.3	657.42
29	mean	8.570E+03	3.028E+04	8.129E+04	**6.761E+03**	9.639E+03	2.079E+04	6.853E+06	2.180E+06	1.030E+04	4.6376e+07
	Std.	1791.4	10180	21333	6195.2	33237	85171	3.86E+06	5.02E+06	1178.1	2.8356e+07
30	mean	1.807E+04	3.613E+04	4.671E+04	1.970E+04	1.811E+04	1.678E+05	1.824E+05	1.021E+05	5.684E+04	**1.6079+04**
	Std.	2975.2	7411.1	10,560	3646.1	2709.9	80774	76669	54161	3977.1	2742.2

Table 4
Average ranking of the algorithm.

Algorithm	MACKO	SAFIRO	TLBO	ASKF	ssSKF	SKF	PSO	BH	GWO	GA
Ranking	2.1833	3.8833	4.7167	5.1667	5.3833	5.4333	6.2333	6.7	6.8833	8.4167

6.2. Convergence behavior

Statistically, MACKO effectively solves the CEC2014 Benchmark test Suite, taking first place in 11 out of 30 functions. The following experiment uses a convergence curve and boxplot to visualize MACKO and other benchmark algorithm performance from each type of benchmark test suite (Fn2, Fn10, Fn20, and Fn26).

6.2.1. Convergence capability and boxplot (Fn2, Fn10, Fn20 and Fn26)

Instead of settling for a local optimum too quickly, a skilled

Table 5
Holm post hoc result of β value for $\alpha = 0.05$.

i	Comparison	z	P	Holm
9	MACKO vs GA	7.973707	0	0.005556
8	MACKO vs GWO	6.01226	0	0.00625
7	MACKO vs BH	5.777739	0	0.007143
6	MACKO vs PSO	5.180777	0	0.008333
5	MACKO vs SKF	4.157414	0.000032	0.01
4	MACKO vs ssSKF	4.093454	0.000042	0.0125
3	MACKO vs ASKF	3.816293	0.000135	0.016667
2	MACKO vs TLBO	3.240651	0.001193	0.025
1	MACKO vs SAFIRO	2.174647	0.029657	0.05

optimization algorithm must converge towards a global optimum. Convergence curve in Fig 4 (left column) indicate that the average fitness value (represented by the black dashed line) of the MACKO algorithm varies gradually at first, stabilizes after a given number of iterations, and stays constant until the method is completed. The graphical representation of the test function demonstrates MACKO's effectiveness in searching the search space with balanced between exploration and exploitation phase. Primarily, when addressing Fn10 and Fn26, MACKO competes effectively against other algorithms with more exploration periods. Meanwhile, tackling Fn2 MACKO with more exploration period shows comparable to SAFIRO and TLBO, although MACKO is below their ranking. Similarly, in the context of Fn26, MACKO with more exploration period shows comparable to PSO, although MACKO is below their ranking. Meanwhile, the boxplot pattern shown in Fig 4 (right column) displays the consistent performance of MACKO, as indicated by its low minimum value and small deviation from the median value when solving the same benchmark problem over 50 runs. In general, MACKO remains a strong competitor compared to SAFIRO, ssSKF, SKF, ASKF, PSO, GA, GWO, BH, and TLBO.

6.2.2. The trajectory of MACKO agent

The trajectory of the MACKO algorithm's agent, plotted for a single dimension over the iterations, is presented to validate its exploration and exploitation capabilities. As depicted in Fig 5, two graphs (one-dimensional and two-dimensional) are generated for the unimodal function (Fn2), simple multimodal function (Fn10), and composition function (Fn26) to illustrate how the search agent's solution compares to the best-so-far solution (*X_best_so_far*) throughout the process. The blue dot-line traces the path of the search agent, while the red solid-line represents the best solution found thus far.

In the initial phase of the iterations, the search agent trajectory exhibits extensive exploration in both dimensions, encompassing practically the whole region. As the iterations progress, the search agent's trajectory modifications slow, indicating a move toward exploitation. The plateau at the conclusion of the iterations indicates that the MACKO algorithm's agent found a solution close to optimal before hitting the maximum iteration limit. Throughout this process, the best solution is updated anytime a better one is found.

This well-handled continuum of exploration and exploitation corresponds to the sufficient or ideal exploration period for the algorithm, as was previously mentioned. MACKO excels in unimodal, simple multimodal, and composition functions because of its skilful navigation of the solution space, early identification of interesting regions, and subsequent strategic exploitation of these regions. The discovery-exploitation shift seen in the convergence patterns highlights how adaptable MACKO is when taking on a range of engineering optimization problems and maintaining a competitive edge.

6.2.3. Search history of MACKO agent

Subsequently, the proficiency of the MACKO algorithm in exploring and exploiting the search space to identify optimal solutions is assessed by examining its agent's search history. The agent's mobility is visualized on contour maps for selected two-dimensional functions, encompassing unimodal (Fn2), simple multimodal (Fn10), and composition problems (Fn26), presented in Fig 6.

The agent's traversed locations are denoted by black stars (*), while a red circle (o) represents the ultimate best solution. An analysis of the search history for Fn2, illustrated in Fig 6(a), unveils the MACKO algorithm's excellence in exploitation. In this context, MACKO manoeuvres the search space, adeptly exploring promising regions and exploiting them to attain optimal solutions. This same trend is evident in the search history for Fn10, as depicted in Fig 6(b), highlighting MACKO's robust exploration capabilities. Once again, the algorithm effectively explores promising areas and exploits them to achieve optimal solutions. Transitioning to the search history for Fn26, as presented in Fig 6(c), it becomes apparent that the MACKO algorithm strikes a harmonious balance between exploration and exploitation.

MACKO efficiently explores promising areas within the search space in this scenario while exploiting them to reach optimal solutions. This consistent performance across diverse test functions underscores the adaptability of the MACKO algorithm in addressing engineering optimization challenges. MACKO demonstrates a resilient exploration phase, skillfully avoiding local optima, as particularly evident in Fig 7 (b). This success is attributed to the efficacy of the updated strategy employed, which involves the strategic shrinking of local neighbourhoods, as elucidated in (17).

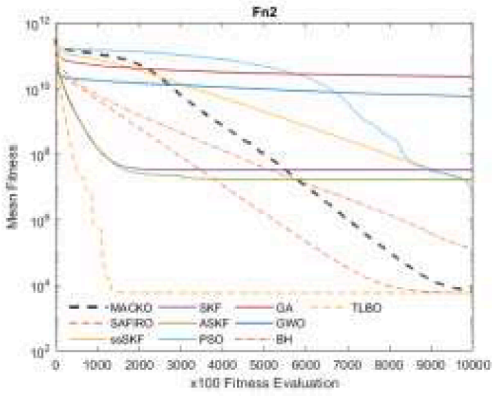
6.2.4. Fitness trends of MACKO agent

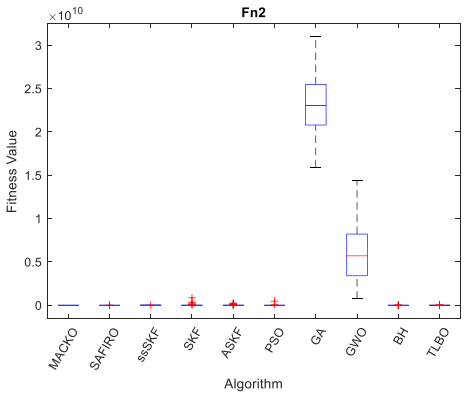
Lastly, a comparison of MACKO's agent's performance in a single run with the best solution discovered thus far (*X_best_so_far*) can be seen in the fitness trend graph. The agents belonging to MACKO, denoted by the blue dot line in Fig 7, demonstrate excellent performance. When the best-so-far solution improves across unimodal, simple multimodal, and composite functions, they begin exploring and progressively move to exploitation. The sub-figure in Fig 7 shows how each agent works simultaneously to independently calculate the fitness value in the early iterations (exploration phase) and then share it with other agents. A diminishing local neighbourhood with a synchronized search strategy within the CKF framework highlights how well the MACKO algorithm conducts a strong exploration phase and is essential for avoiding local traps.

Overall, MACKO proved to have an optimal or enough exploration period due to the CKF update strategy associated with the shrinking local neighbourhood. The MACKO is a practical algorithm for solving simple multimodal functions and demonstrates strong competitiveness in solving unimodal, hybrid, and composition functions. MACKO successfully led eight out of 13 functions in simple multimodal and one out of six functions in hybrid. MACKO obtains second and third ranks for the remaining 17 functions in the unimodal and composition functions. Despite coming in second and third place, MACKO was nearly equal to the first-rank algorithm. MACKO outperforms nine other population-based algorithms, including the recent estimation base algorithm, even with the same number of function evaluations. Furthermore, MACKO uses only one parameter compared to different estimation-based algorithms, making tuning easy.

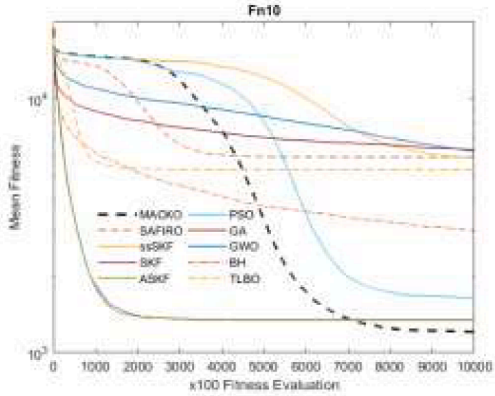
7. Conclusions

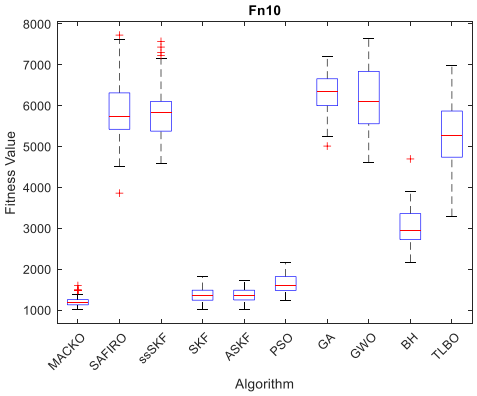
This study presents the Multi-Agent Cubature Kalman Optimizer (MACKO) as a new optimization technique motivated by CKF's estimation capabilities. The goal of MACKO is to find optimal or nearly optimal solutions. This work is limited to solving a single-objective optimization problem within the 30 benchmark functions in the CEC 2014 benchmark suite. The MACKO algorithm comprises six fundamental steps: initialization, fitness evaluation, *X_best_so_far* update, solution prediction, simulated measurement, measurement prediction, and solution update. The initial three steps share similarities with other metaheuristic algorithms, whereas the latter steps draw inspiration from the Cubature Kalman Filter (CKF) framework. A distinctive feature of



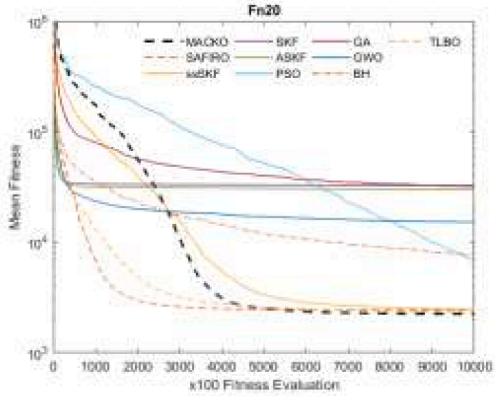


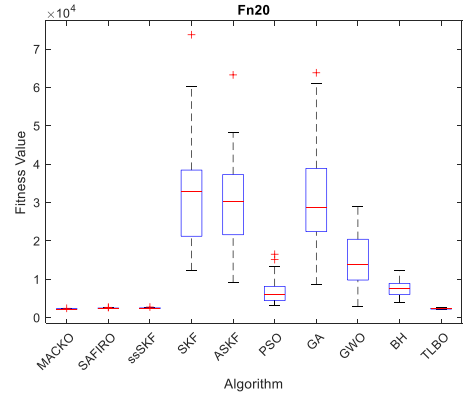
(a) Unimodal (Fn2)



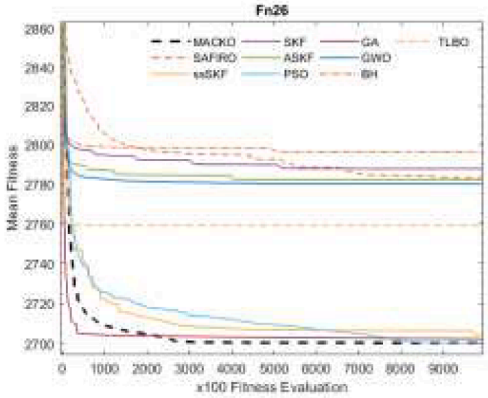


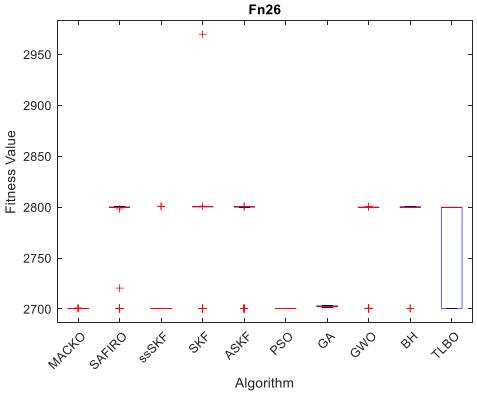
(b) Simple multimodal (Fn10)





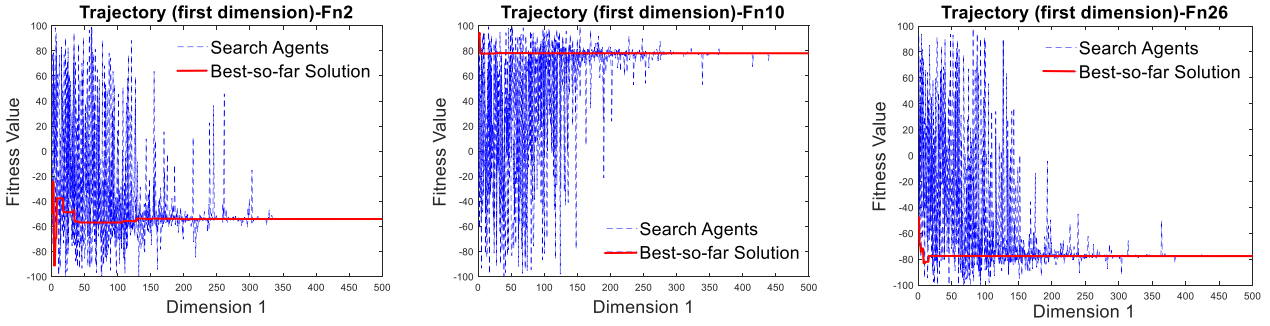
(c) Hybrid (Fn20)



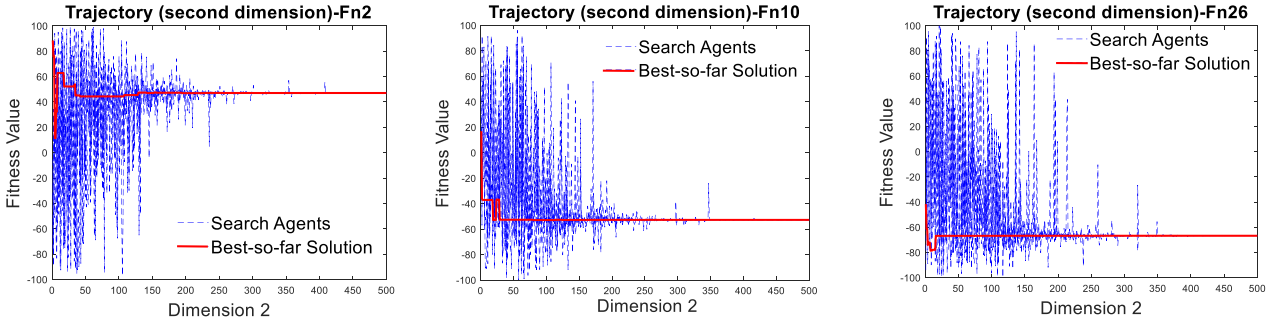


(d) Composition (Fn26)

Fig. 4. Convergence curve and Boxplot comparison for Fn2, Fn10, Fn20 and Fn26;.

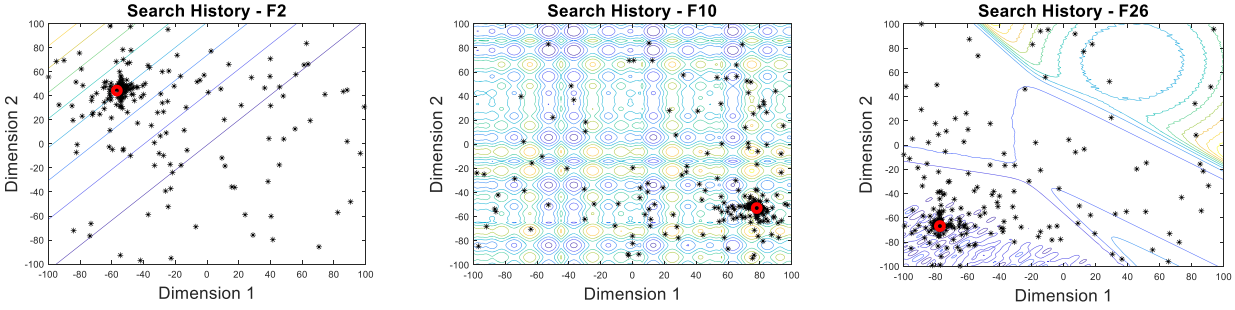


(a) Unimodal (dimension 1) (c) Simple multimodal (dimension 1) (e) Composition (dimension 1)



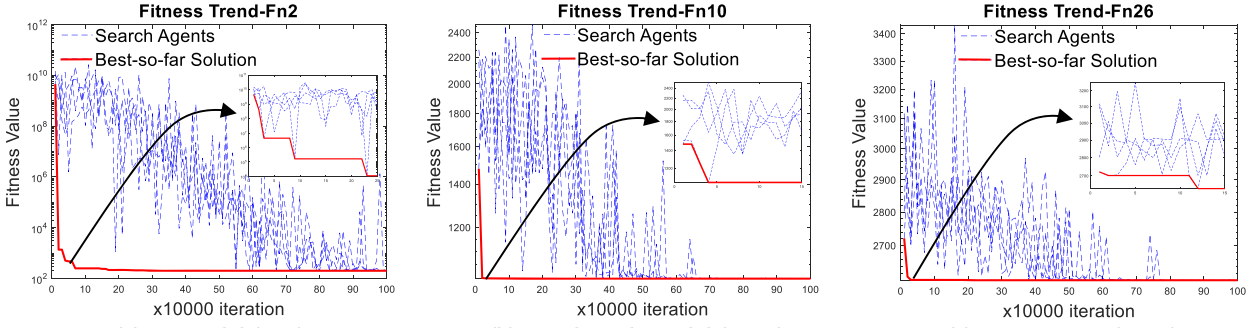
(b) Unimodal (dimension 2) (d) Simple multimodal (dimension 2) (f) Composition (dimension 2)

Fig. 5. Trajectory for Fn2, Fn10, and Fn26.



(a) Unimodal (Fn2) (b) Simple multimodal (Fn10) (c) Composition (Fn26)

Fig. 6. Search history of the MACKO's agent for Fn2, Fn10 and Fn26.



(a) Unimodal (Fn2) (b) Simple multimodal (Fn10) (c) Composition (Fn26)

Fig. 7. Fitness trend of the MACKO's agent for Fn2, Fn10 and Fn26.

MACKO is its reliance on a singular parameter. Meanwhile, the CTT, containing the local neighborhood, provides an optimal or sufficient exploration period. The performance of MACKO is then analyzed using the Friedman test and Post Hoc Holm test through four types of experiments within the CEC 2014 Benchmark Test Suite over nine benchmark algorithms: SAFIRO, ssSKF, SKF, ASKF, PSO, GA, GWO, BH, and TLBO. The result shows that MACKO ranks first and significantly outperforms all benchmark algorithms. Subsequently, the behaviors of

MACKO agents are investigated using convergence curves, trajectories, search histories, and fitness trends. Based on the obtained results and findings, it is evident and can be concluded that MACKO exhibits an optimal or sufficient exploration period compared to other benchmark algorithms. The significance of the MACKO algorithm lies in its ability to provide fresh insights and avenues for further exploration and advancement. Researchers can build upon, modify, or integrate it with other algorithms to devise more effective optimization solutions. Moreover, the MACKO algorithm can find applications in solving optimization problems across diverse fields. MACKO can be enhanced by adapting it for multi-objective optimization problems, binary optimization problems, and combinatorial optimization problems. Fig. 3

Future works

For future work, the searching strategies of MACKO algorithm are investigate and incorporation to escape from local optima. Furthermore, the applicability of MACKO's algorithm in addressing real engineering optimization problems, such as the tuning of parameters in proportional-integral-derivative (PID) controllers and the resolution of printed circuit board (PCB) routing problems, will be tested.

Author contributions

The task of individual authors: Zulkifli Musa conducted on develop methodology, software, data curation and writing (original draft preparation); Mohd Ibrahim Shapiai (Supervision) conducted visualization and investigation. and writing (editing); Zuwairie Ibrahim conducted conceptualization, validation, and writing (reviewing); all authors had approved the final version.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

Holland, J. H. (1984). Genetic algorithms and adaptation. In *Adaptive control of ill-defined systems, 16* pp. 317–333). Boston, MA: Springer US.

Nakib, A., Thibault, B., & Siarry, P. (2015). Bayesian based metaheuristic for large scale continuous optimization. In *, 2015. Proc. - 2015 IEEE 29th Int. Parallel Distrib. Process. Symp. Work. IPDPSW* (pp. 314–322). https://doi.org/10.1109/IPDPSW.2015.150

Dhivyaprabha, T. T., Subashini, P., & Krishnaveni, M. (2018). Synergistic fibroblast optimization: A novel nature-inspired computing algorithm. *Frontiers of Information Technology & Electronic Engineering, 19*(7). https://doi.org/10.1631/FITEE.1601553

Chen, X., Liu, Y., Li, X., Wang, Z., Wang, S., & Gao, C. (2019). A new evolutionary multiobjective model for traveling salesman problem. *IEEE access : practical innovations, open solutions, 7*. https://doi.org/10.1109/ACCESS.2019.2917838

Sulaiman, M. H., Mustaffa, Z., Saari, M. M., & Daniyal, H. (Jan. 2020). Barnacles mating optimizer: A new bio-inspired algorithm for solving engineering optimization problems. *Engineering Applications of Artificial Intelligence, 87*(September 2019), Article 103330. https://doi.org/10.1016/j.engappai.2019.103330

Kennedy, J., & Eberhart, R. (1995). Particle swarm optimisation. In *Proc. IEEE Int. Conf. Neural Network* (pp. 1972–1978). https://doi.org/10.1007/978-3-030-61111-8_2

Mirjalili, S., Gandomi, A. H., Mirjalili, S. Z., Saremi, S., Faris, H., & Mirjalili, S. M. (Dec. 2017). Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems. *Advances in Engineering Software, 114*, 163–191. https://doi.org/10.1016/j.advengsoft.2017.07.002

Heidari, A. A., Mirjalili, S., Faris, H., Aljarah, I., Mafarja, M., & Chen, H. (2019). Harris hawks optimization: Algorithm and applications. *Future Generation Computer Systems, 97*. https://doi.org/10.1016/j.future.2019.02.028

Dhiman, G., Garg, M., Nagar, A., Kumar, V., & Dehghani, M. (2021). A novel algorithm for global optimization: Rat swarm optimizer. *Journal of Ambient Intelligence and Humanized Computing, 12*(8). https://doi.org/10.1007/s12652-020-02580-0

Braik, M. S. (2021). Chameleon swarm algorithm: A bio-inspired optimizer for solving engineering design problems. *Expert Systems with Applications, 174*. https://doi.org/10.1016/j.eswa.2021.114685

Hatamlou, A. (2013). Black hole: A new heuristic optimization approach for data clustering. *Information Sciences, 222*. https://doi.org/10.1016/j.ins.2012.08.023

Mirjalili, S., Mirjalili, S. M., & Hatamlou, A. (2016). Multi-Verse optimizer: A nature-inspired algorithm for global optimization. *Neural Computing and Applications, 27*(2). https://doi.org/10.1007/s00521-015-1870-7

Mirjalili, S. (2016). SCA: A sine cosine algorithm for solving optimization problems. *Knowledge-Based System, 96*. https://doi.org/10.1016/j.knosys.2015.12.022

Hashim, F. A., Houssein, E. H., Mabrouk, M. S., Al-Atabany, W., & Mirjalili, S. (Dec. 2019). Henry gas solubility optimization: A novel physics-based algorithm. *Future Generation Computer Systems, 101*, 646–667. https://doi.org/10.1016/j.future.2019.07.015

Azizi, M., Talatahari, S., & Giaralis, A. (2021). Optimization of engineering design problems using atomic orbital search algorithm. *IEEE access : practical innovations, open solutions, 9*, 102497–102519. https://doi.org/10.1109/ACCESS.2021.3096726

Rao, R. V., Savsani, V. J., & Vakharia, D. P. (2012). Teaching-Learning-Based optimization: An optimization method for continuous non-linear large scale problems. *Inf. Sci. (Ny)., 183*(1). https://doi.org/10.1016/j.ins.2011.08.006

Sadollah, A., Bahreininejad, A., Eskandar, H., & Hamdi, M. (2013). Mine blast algorithm: A new population based algorithm for solving constrained engineering optimization problems. *Applied Soft Computing Journal, 13*(5). https://doi.org/10.1016/j.asoc.2012.11.026

Kaveh, A., & Mahdavi, V. R. (2014). Colliding bodies optimization: A novel meta-heuristic method. *Computers and Structures, 139*. https://doi.org/10.1016/j.compstruc.2014.04.005

Gandomi, A. H. (2014). Interior search algorithm (ISA): A novel approach for global optimization. *ISA Transactions, 53*(4). https://doi.org/10.1016/j.isatra.2014.03.018

Mohamed, A. W., Hadi, A. A., & Mohamed, A. K. (2020). Gaining-sharing knowledge based algorithm for solving optimization problems: A novel nature-inspired algorithm. *International Journal of Machine Learning and Cybernetics, 11*(7). https://doi.org/10.1007/s13042-019-01053-x

Yan, Y. (Jun. 2020). Gingivitis detection by fractional Fourier entropy with optimization of hidden neurons. *International Journal of Cognitive Computing in Engineering, 1* (June), 36–44. https://doi.org/10.1016/j.ijcce.2020.09.003

Koksal, E., Hegde, A. R., Pandiarajan, H. P., & Veeravalli, B. (Jun. 2021). Performance characterization of reinforcement learning-enabled evolutionary algorithms for integrated school bus routing and scheduling problem. *International Journal of Cognitive Computing in Engineering, 2*(February), 47–56. https://doi.org/10.1016/j.ijcce.2021.02.001

Mustaffa, Z., & Sulaiman, M. H. (Jun. 2023). Stock price predictive analysis: An application of hybrid barnacles mating optimizer with artificial neural network. *International Journal of Cognitive Computing in Engineering, 4*(December 2022), 109–117. https://doi.org/10.1016/j.ijcce.2023.03.003

Kavita, P., Alli, D. R., & Rao, A. B. (Jun. 2022). Study of image fusion optimization techniques for medical applications. *International Journal of Cognitive Computing in Engineering, 3*(November 2021), 136–143. https://doi.org/10.1016/j.ijcce.2022.05.002

Toscano, R., & Lyonnet, P. (2009). Heuristic Kalman algorithm for solving optimization problems. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 39* (5). https://doi.org/10.1109/TSMCB.2009.2014777

Ibrahim, Z., Aziz, N. H. A., Nor, N. A., Razali, S., & Mohamad, M. S. (2016). Simulated Kalman Filter: A novel estimation-based metaheuristic optimization algorithm. *Advanced Science Letters, 22*(10), 2941–2946. https://doi.org/10.1166/asl.2016.7083

Abdul Aziz, N. H., Ibrahim, Z., Ab Aziz, N. A., Mohamad, M. S., & Watada, J. (2018). Single-solution simulated Kalman filter algorithm for global optimisation problems. *Sadhana - Academy Proceedings in Engineering Sciences, 43*(7), 1–15. https://doi.org/10.1007/s12046-018-0888-9

Nor, N. A., Ibrahim, Z., Aziz, N. H. A., & Rahman, T. A. (2018). Asynchronous simulated Kalman filter optimization algorithm. *International Journal of Engineering & Technology, 7*(4), 44–49. https://doi.org/10.14419/ijet.v7i4.27.22478

Ab Rahman, T., Ibrahim, Z., Ab Aziz, N. A., Zhao, S., & Abdul Aziz, N. H. (2018). Single-Agent finite impulse response optimizer for numerical optimization problems. *IEEE Access : Practical Innovations, Open Solutions, 6*, 9358–9374. https://doi.org/10.1109/ACCESS.2017.2777894

Toscano, R., & Lyonnet, P. (2012). A Kalman optimization approach for solving some industrial electronics problems. *IEEE Transactions on Industrial Electronics, 59*(11). https://doi.org/10.1109/TIE.2011.2169637

Robert, O., Hankun, Z., Shifeng, L., & Borut, B. (2018). Improved Heuristic Kalman algorithm for solving multi-objective flexible job shop scheduling problem. In *Procedia Manufacturing, 17*. https://doi.org/10.1016/j.promfg.2018.10.142

Z.M. Yusof, I. Ibrahim, S.N. Satiman, Z. Ibrahim, N.H.A. Aziz, and N.A.A. Aziz, "BSKF: Simulated Kalman filter," 2016, 10.1109/AIMS.2015.23.

Yusof, Z. M., et al. (2016b). Distance evaluated simulated Kalman filter for combinatorial optimization problems. *ARPN: Journal of Engineering and Applied Sciences, 11*(7), 4911–4916.

Duong, P. L. T., & Raghavan, N. (2018). Heuristic Kalman optimized particle filter for remaining useful life prediction of lithium-ion battery. *Microelectronics and Reliability, 81*. https://doi.org/10.1016/j.microrel.2017.12.028

B. Muhammad, Z. Ibrahim, K. Zakwan, and M. Azmi, "Four Different Methods to Hybrid Simulated Kalman Filter (SKF) with Particle Swarm Optimization (PSO)," 2016.

Muhammad, B., Ibrahim, Z., Jusof, M. F. M., Aziz, N. A. A., Aziz, N. H. A., & Mokhtar, N. (2017). A hybrid Simulated Kalman Filter - Gravitational Search Algorithm (SKF-GSA. In *, 22. Proc. Int. Conf. Artif. Life Robot.* https://doi.org/10.5954/icarob.2017.gs11-5

Zhang, H., Buchmeister, B., Li, X., & Ojstersek, R. (2021). Advanced metaheuristic method for decision-making in a dynamic job shop scheduling environment. *Mathematics, 9*(8). https://doi.org/10.3390/math9080909

Rahman, T. A., Ibrahim, Z., Aziz, N. A. A., Zhao, S., & Aziz, N. H. A. (2018). Single-Agent finite impulse response optimizer for numerical optimization problems. *IEEE access : practical innovations, open solutions, 6*. https://doi.org/10.1109/ACCESS.2017.2777894

Aziz, N. H. A., Aziz, N. A. A., Ibrahim, Z., Razali, S., Abas, K. H., & Mohamad, M. S. (2017). A Kalman Filter approach to PCB drill path optimization problem. In *Proc. - 2016 IEEE Conf. Syst. Process Control. ICSPC 2016* (pp. 33–36). https://doi.org/10.1109/SPC.2016.7920699

J.J. Liang, B.Y. Qu, and P.N. Suganthan, Problem definitions and evaluation criteria for the CEC 2014 special session and competition on single objective real-parameter numerical optimization, no. December 2013. 2014.

Alcalá-Fdez, J., et al. (2009). KEEL: A software tool to assess evolutionary algorithms for data mining problems. *Soft Computing, 13*(3). https://doi.org/10.1007/s00500-008-0323-y

Eberhart, R. C., & Shi, Y. (2000). Comparing inertia weights and constriction factors in particle swarm optimization. In , *1. Proceedings of the 2000 Congress on Evolutionary Computation, CEC 2000*. https://doi.org/10.1109/CEC.2000.870279

Ahn, C. W. (2006). Practical genetic algorithms. *Studies in Computational Intelligence, 18*, 7–22. https://doi.org/10.1007/11543138_2

Mirjalili, S., Mirjalili, S. M., & Lewis, A. (Mar. 2014). Grey wolf optimizer. *Advances in Engineering Software, 69*, 46–61. https://doi.org/10.1016/j.advengsoft.2013.12.007

Derrac, J., García, S., Molina, D., & Herrera, F. (2011). 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*(1). https://doi.org/10.1016/j.swevo.2011.02.002

Zulkifli Musa received the Diploma and B.Eng. degree (Hons) in electrical engineering from Universiti Teknikal Malaysia Melaka, Malaysia, in 2005 and 2008, respectively, and the M. Eng. degree in image processing from Universiti Kebangsaan Malaysia, Bangi, Malaysia, in 2014. He is currently pursuing the Ph.D. degree with the Malaysia-Japanese International Institute, Universiti Teknologi Malaysia, Kuala Lumpur, Malaysia. Her current research interests include computational intelligence and its application in engineering.

Zuwairie Ibrahim received the B.Eng. degree in electrical engineering and the M.Eng. degree in image processing from Universiti Teknologi Malaysia, Johor, Malaysia, in 2000 and 2003, respectively, and the Ph.D. degree in DNA computing from Meiji University, Tokyo, Japan, in 2006. From 2002 to 2008, he was a Lecturer with Universiti Teknologi Malaysia, Johor, Malaysia. He was then promoted to Senior Lecturer in 2008 and was with Universiti Teknologi Malaysia until 2012. In 2012, he moved to Universiti Malaysia Pahang, Pahang, Malaysia, to serve as an Associate Professor. He is one of the inventors of the simulated Kalman filter and Finite Impulse Response Optimizer which is an optimization algorithm based on the estimation framework. He has co-authored the book Bioevaluation of World Transport Networks (World Scientific, 2012). He has been appointed to the Editorial of International Journal of Simulation: Systems, Science, and Technology by the U.K. Simulation Society and Journal Elektrika by the Faculty of Electrical Engineering, Universiti Teknologi Malaysia. He has also been a Visiting Researcher in universities in Japan and Malaysia. He has authored and co-authored more than 50 publications in international journals and more than 100 publications in conferences. His-current research interests include computational intelligence, image processing, and unconventional computation such as molecular or DNA computing.

Mohd Ibrahim Shapiai (Member, IEEE) received the M.Eng. degree from the University of York, U.K., in 2007, and the Ph. D. degree in the area of machine learning from the Universiti Teknologi Malaysia, in 2013.

From March 2010 to April 2010, he was a Visiting Researcher with the Graduate School of Information, Production and Systems, Waseda University, Japan, under the supervision of Dr. Junzo Watada, and the Faculty of Engineering, Leeds University, U.K., from June 2012 to July 2012, under the supervision of Dr. Vassili Toropov. He is currently a Senior Lecturer with the Universiti Teknologi Malaysia and a Researcher with the Center of Artificial Intelligence and Robotics (CAIRO). He has also been appointed as a Certified NVIDIA Deep Learning Instructor. His-research interests include artificial intelligence, machine learning, brain–computer interface, and swarm intelligence.