A Novel Particle Swarm Optimizer and Its Application to the Yield Curve Estimation Problem

JIAQI ZHANG1 AND BO SHI2
1School of Business, Xi’an University of Finance and Economics, Xi’an, Shaanxi 710064, China
2Department of Mathematical Economics, Northwest University, Xi’ an, Shaanxi 710069, China
Corresponding author: Bo Shi (shibo.nwu@sina.com)

ABSTRACT Particle swarm optimization (PSO) has been considered as one of the main swarm intelligence algorithms for solving single-objective optimization problem. How to update the velocities and positions of a swarm of particles is key to its optimization performance. In this paper, we propose to use the moving average of the local best positions visited so far as a key information to update the velocities and positions. Further, a central learning strategy is proposed in which the center of the local best positions is computed and used to update the global best position. Combining these two strategies with the updating formulas which are inspired by the free electron model in metal conductors placed in an external electric field, we name the proposed PSO algorithm as PSO with moving average and central learning strategies (dubbed as MAPSO). We test the performance of MAPSO on the CEC 2017 test problems with 10 and 30 dimensions. Experimental results show that MAPSO significantly outperforms some well-known PSOs in general. We then apply MAPSO on a very challenging real-world problem, i.e. the yield curve estimation problem, in macroeconomics. Our experimental study on the yield curve estimation problem with Shanghai interbank offered rates shows that MAPSO can effectively solve the problem and achieve the state-of-the-art performance.

INDEX TERMS Particle swarm optimization, yield curve estimation, swarm intelligence.

I. INTRODUCTION

Swarm intelligence is one of the main streams of evolutionary algorithm (EA), along with genetic algorithm (GA), differential evolution (DE), evolutionary strategies (ES), evolutionary programming (EP) and many others. As opposed to other EAs which are established based on the survival of the fittest principle, swarm intelligence algorithms mimic cooperative communication and information exchange among individuals within a swarm of particles (e.g. a flock of fish or a swarm of ants) [1]. It is one of the main branches of population-based stochastic optimization paradigms for solving complex optimization problem.

As a representative of swarm intelligence algorithms, particle swarm optimization (PSO) was proposed by Kennedy and Eberhart in 1995 [1]. Since then it has been widely applied to optimize complex optimization problems and train neural networks. PSO is inspired by the hunting behavior of a flock of birds, where each bird is considered as a particle. In PSO, each particle has its velocity and position. PSO mimics the bird’s group search pattern through cooperative interaction among particles, and evolves the particles to search for the optimal solution of an optimization problem. Different from other EAs, such as GA, DE, ES and EP, etc., which employ crossover and/or mutation operators, PSO determines the search direction of each particle through position sharing among individuals within a swarm of particles (e.g. a flock of fish or a swarm of ants) [1]. It is one of the main branches of population-based stochastic optimization paradigms for solving complex optimization problem.

The associate editor coordinating the review of this manuscript and approving it for publication was Yu-Da Lin.®
neighboring topology for information sharing within the swarm. PSO suffers from fast convergence, lack of exploration capability, decline of solutions variety, and easily trapping in local optimum and other disadvantages. A great number of PSO variants have been developed to address these issues. The development of PSO can be roughly categorized into the following research avenues.\(^1\)

One of the popular research avenues in developing efficient PSO is the introduction of the so-called ‘inertia weight’, which was firstly proposed by Shi and Eberhart. The idea is to equip each particle with an inertia weight on its current velocity. In [6], Shi and Eberhart proposed to decrease linearly the inertia weight along with the evolution. This idea has been shown empirically to be helpful for accelerating the convergence speed of PSO. Yang et al. [7] proposed to change the inertia weight non-linearly according to the positions of the particles, which can be seen as an adaptive strategy. Bansal et al. [8] compared various inertia weight adaption strategies and presented some guidelines. Different from the original PSO in which each particle equips with one inertia weight, Taherkhani et al. [9] proposed to use different weights on different dimensions. Zhang and Li proposed an adaptive inertia weight PSO in [10], in which the speed change and momentum of each dimension of the particle are set adaptively. An adaptive inertia weight technique has been developed in [11] based on the concept of success rate usually adopted in DE [12]. Recently, researchers have developed methods to adapt the weights based on sigmoid function [13] or exponential function [14].

Secondly, the updating method for the velocities and positions of the particles has been studied extensively. In the original PSO, the updating of the velocities involves three parts, including the momentum, the cognitive and the social part. The cognitive part reflects the learning from a particle’s own search experience, while the social part represents its collaboration with other particles. In the original PSO, the cognitive part is associated with the best position a particle has found until current iteration (called ‘local best’ or \(lbest\)), while the social part involves the best position found by all the particles (called ‘global best’ or \(gbest\)).

In the PSOs proposed in [15] and [16], the particle position is determined by sampling from a Gaussian distribution based on the positions of \(lbest\) and \(gbest\). Some researchers proposed to use Levy distribution which has empirically shown better algorithmic performance than using Gaussian [17], [18]. Liang et al. [19] proposed a learning-based PSO, in which the particles to be learned are selected by setting the learning rate. This learning strategy can increase the diversity of the particles and avoid the premature convergence. Sun et al. [20] proposed to update the velocity by taking the social part as the addition of a random motion term and a drift motion term. Their development is inspired by the movement model of electrons in metal conductors.

Cheng et al. [21] proposed a PSO based on human’s social behavior. Tanweer et al. [22] proposed a strategy to improve the particle’s velocity, in which the best particle and the other particles at the current iteration adopt different update rules. The strategy aims to speed up the exploration ability of the best particle and improve the convergence speed of other particles. Karim et al. [23] proposed to incorporate an optimal guide creation module to guide the particles escaping from local or non-optimal solution regions.

An improved PSO inspired by social learning was developed in [24] where the three best particles in the current iteration are updated using a differential mutation strategy. Li et al. [25] proposed to mix some mutation strategies to update the velocity based on the idea of dividing the total population into an elitist population and a general population. Sedighizadeh et al. proposed to add two new terms in the original PSO updating formula [26]. Xu et al. [27] proposed to use reinforcement learning to enhance the usefulness of the communication topology in PSO.

Thirdly, the multi-population strategy has been studied for improving PSO’s exploration capability. In [25], the authors proposed a multi-population method with mixed mutation strategies, in which the whole population is divided into some subpopulations and each subpopulation evolves cooperatively. Peng and Chen [28] proposed a cooperative PSO with multiple strategies, in which each particle represents a fuzzy regulation and each particle in each subpopulation evolves separately to avoid trapping in local optimums.

Tanweer et al. [29] proposed a PSO with dynamic guidance and self-adjustment, in which the particles are divided into three categories according to the Euclidean distances from the particles to the global optimum. In [30], Ye et al. proposed a new multi-swarm PSO algorithm based on a dynamic learning strategy, in which the particles in each subpopulation are divided into common particles and corresponding particles. At each iteration, the common particles mainly pay attention to exploration, while the corresponding particles intend to perform exploitation. Chen et al. [31] proposed to dynamically group particles during the search procedure.

The fourth avenue is to combine PSO with evolutionary operators, such as mutation and crossover, adopted in other EAs, expecting to make use of their advantages. In [32], Angeline proposed to adopt the tournament selection method used in GA to improve the performance of PSO. Lvbjerg et al. proposed a hybrid PSO algorithm which combines the idea of particle swarm with the concept of GA [33].

To improve the diversity of particle swarms in PSO, crossover operators have been used to generate individuals with better quality. In [34], Meng et al. proposed a cross-search particle swarm optimization (CSPSO) algorithm. Unlike PSO and its variants, the particles in CSPSO are sequentially updated in each generation through the original PSO update strategy and the crossover operation. In [35], Ratnaweera et al. proposed a novel parameter automation strategy and two extended methods to update the control parameters in PSO. A time-varying acceleration coefficient

---

\(^1\)Since the study on PSO is very popular, this paper does not intend to review all the literatures.
is then proposed for adapting the inertia weights, which is then combined with the mutation operator in GA.

In PSO, the particles cooperate and share information with each other through a specific structure, known as neighborhood topology. The neighborhood topology structure could greatly affect the performance of PSO since it determines how the particles communicate and how the useful information spreads within the swarm. In a word, the neighborhood topology could influence the particles’ behavior greatly. The original PSO adopts a star topology where each particle considers all the other particles as neighbors. The ring topology, on the other hand, takes its each particle’s two immediate neighbors to form a circle [36]. The Von Neumann topology can be considered as a rectangle matrix which determines which particles are connected [37]. In dynamic topology, the neighborhood is reassigned after a certain number of iterations [38].

Kennedy and Mendes [36] tested the “pyramid” triangle topology based on 3D wireframe, which is a star topology with one central node and the community structure of Von Neumann. Bonyadi and Michalewicz [3] established a time adaptive topology which makes the variant of PSO locate many feasible regions in the early stage and lock the most promising regions in the later stage of the optimization process. Zeng et al. [39] proposed a dynamic neighborhood scheduling strategy to adaptively determine the neighborhood topology of particles, which is realized by monitoring the aggregation factor of particles. In [40], Zhang et al. proposed a hybrid topology, which combines the closed-loop topology with the star topology or quadrilateral topology.

Besides the aforementioned research studies, there are many other research avenues, including but not limited to, hybridizing with other evolutionary algorithms (e.g. with DE [41], whale optimization [42], quantum [43], and others), swarm initialization [44], population size controlling [45], etc. Further, PSO has been also widely studied and applied to a great number of other benchmarking problems such as constrained optimization problems [46], large-scale problems [47], and combinatorial optimization problems [48]; and real-world optimization problems such as the position of ocean-moored buoys [49], the process synthesis and design problem [50], the scale-free network problem [51], the image segmentation problem [52], and many others. Interested reader please refer to [2] for a comprehensive review.

In this paper, we propose a novel PSO focusing on the development of new updating method. The main contributions of the paper can be summarized as follows:

- A new updating strategy is proposed, in which the moving average of the local best positions with correction is rigorously analyzed and applied to update the cognitive part of the velocity.
- A new central learning strategy is proposed to update the global best position;
- The proposed PSO is compared with a number of PSOs on the commonly-used CEC 2017 test suite.

Experimental results show that the proposed algorithm significantly outperforms the compared algorithms in general.

- The proposed algorithm is applied to solve a very challenging real-world non-convex optimization problem, namely the yield curve estimation problem. Experimental results show that the proposed algorithm significantly outperforms a variety of algorithms including some classical optimization methods and EAs.

The rest of the paper is organized as follows. Section II presents the proposed strategies. Section III presents the yield curve structure estimation problem. Experimental results on the CEC 2017 test suite are presented in Section IV, while the application to the yield curve estimation problem is presented in Section V. Section VI concludes the paper.

II. PARTICLE SWARM OPTIMIZER BASED ON MOVING AVERAGE AND CENTRAL LEARNING STRATEGY

A. BASIC PSO

In this paper, we consider box-constrained global optimization problem:

$$\min_{x \in \Omega} f(x)$$

where $x = (x_1, \ldots, x_d)^T$ and $\Omega$ is the decision space. Here we consider box constrains, i.e. $\Omega = \prod_{i=1}^d [l_i, u_i]$ where $l_i$ and $u_i$ represents the lower and upper bound of each decision variable $x_i$, respectively.

PSO maintains a population of particles and searches the decision space by updating the positions and velocities of the particles according to some information stochastically. In PSO, at the $t$-th iteration, each particle is composed of a position vector $x_i^t = (x_{i1}^t, \ldots, x_{id}^t)^T$ (which can be considered as a solution to the optimization problem) and a velocity vector $v_i^t = (v_{i1}^t, \ldots, v_{id}^t)^T$. In the original PSO, let $y_i^t$ be the best position found by the $i$-th particle (i.e. $lbest_i$), and $g_i^t$ be the best position (i.e. $gbest$) found by all the particles, until the $t$-th iteration. The update equations for the position and velocity are as follows:

$$v_{i1}^{t+1} = v_{i1}^t + c_1 r_1 \cdot [y_{i1}^t - x_{i1}^t] + c_2 r_2 \cdot [g_{i1}^t - x_{i1}^t]$$

$$x_{i1}^{t+1} = x_{i1}^t + v_{i1}^{t+1}$$

where $c_1$ and $c_2$ are the cognitive and social acceleration coefficients, $r_1$ and $r_2$ are two random numbers sampled within $[0, 1]$ from the uniform distribution $U(0, 1)$.

The first term in the right hand side of Eq. (1) is the momentum part, while the second and third part is called the cognitive and social part, respectively. The cognitive part reflects the influence of the search experience of an individual particle, while the influence of the rest particles comes from the social part. For a minimization problem, the update of $y_i^t$ is as follows:

$$y_{i1}^{t+1} = \begin{cases} y_{i1}^t & \text{if } f(x_{i1}^{t+1}) \geq f(y_{i1}^t) \\ x_{i1}^{t+1} & \text{otherwise} \end{cases}$$

It should be pointed out that PSO differs from DE mainly on the following aspects. First, in DE, each individual is not
considered to be a particle. There is no velocity associated with each individual in DE. Second, the updating of each individual, usually called the mutation operator, in DE is quite different from PSO. An example DE mutation operator, named rand/1, is as follows:

\[ x_{i}^{t+1} = x_{i}^{t} + F \cdot (x_{i}^{t} - x_{i}^{t}) \]

where \( F \) is a parameter called scaling factor, \( r_{1}, r_{2} \) and \( r_{3} \) are random integers sampled from \([1, N]\). Third, there is no crossover operator in PSO.

**B. THE MOVING AVERAGE OF INDIVIDUAL PARTICLE’S BEST POSITIONS**

In classical PSO, the velocity update formula only uses the particle’s local best position and the global best position at the current iteration. In other words, each particle does not consider the positions it has visited before, which means that the particles in the present PSOs have no memory. However, we argue that such a update pattern has its limitation.

Consider a situation when a particle reaches a position near the global optimum at the \( t \)-th iteration, but falls into a local optimum with a smaller objective function value at the \((t+1)\)-th iteration. That is, \( f(x_{i}^{t}) \geq f(x_{i}^{t+1}) \) but \( x_{i}^{t} \) is closer to the global optimum.

Since \( x_{i}^{t+1} \) is better than \( x_{i}^{t} \), \( x_{i}^{t} \) will be forgotten in the following iterations in the classical PSO, which means that such useful information is lost. As a result, the particle may converge to the local optimum. This situation often occurs in the search process of PSO. Moreover, since the local best position of a particle after an iteration will be updated by a better position, the original optimal position will be lost. However, the lost position may lead to a more promising area. If such a situation happens, the particle swarm will converge prematurely. As a result, the PSO’s global exploration ability will be greatly affected.

In consideration of this shortcoming, we propose to maintain all the local best positions that each particle has visited during the search process. In the sequel, we use \( \mathcal{P}_{1:t} \) to denote all the local best positions until the \( t \)-th iteration. Assume that the population size is \( N \), \( \mathcal{P}_{1:t} \) can then be presented as follows:

\[ \mathcal{P}_{1:t} = \begin{bmatrix} y_{1}^{1} & y_{1}^{2} & \ldots & y_{1}^{t} \\ y_{2}^{1} & y_{2}^{2} & \ldots & y_{2}^{t} \\ \vdots & \vdots & \ddots & \vdots \\ y_{N}^{1} & y_{N}^{2} & \ldots & y_{N}^{t} \end{bmatrix} \]  

(4)

where each row represents all the local best positions that a particle has visited until the \( t \)-th iteration. The global best position of all the local best positions, i.e. \( \mathbf{g}^{t} \), can be derived from \( \mathcal{P}_{1:t} \) as

\[ \mathbf{g}^{t} = \arg \min_{1 \leq i \leq N} f(\mathcal{P}_{1:t}[i]), \]

where \( \mathcal{P}_{1:t}[i] \) represents the \( i \)-th column of \( \mathcal{P}_{1:t} \).

To better make use of the historical information of the particles, we propose the following formula for each particle. That is, for each particle \( i \), let

\[ \tilde{y}_{i}^{t} = \gamma \cdot y_{i}^{t-1} + (1 - \gamma) y_{i}^{t}, \quad 0 \leq \gamma \leq 1 \]  

(5)

It is seen from the equation that \( \tilde{y}_{i}^{t} \) is the weighted average over time on \( y_{i}^{t} \) and \( \gamma \in [0, 1] \) is the weight decay rate.

Let’s only consider one particle’s historical information to better understand the effectiveness of the above equation. Expanding Eq. (5), we have

\[ \tilde{y}_{i}^{t} = \gamma \cdot y_{i}^{t-1} + (1 - \gamma) y_{i}^{t} = \gamma \cdot (\gamma \cdot y_{i}^{t-2} + (1 - \gamma) y_{i}^{t-1}) + (1 - \gamma) y_{i}^{t-1} = \cdots = (1 - \gamma) y_{i}^{1} + \cdots + (1 - \gamma) y_{i}^{t-1} \]  

(6)

From the above equation, it can be found that \( \tilde{y}_{i}^{t} \) is the weighted average of all the last local positions until \( t \). The closer to the current iteration \( t \), the larger the weight value for the corresponding local best position.

Since the weights are exponentially decreasing along with the increasing of \( t \) and it is unwise to keep all the visited local best positions, we therefore propose to truncate the memory in case \( \gamma^{t} < 1/e \) where \( e \) is the mathematical constant. Any local best positions beyond the iteration bigger than \(-1/\log \gamma\) will be ignored. Since \( 1/e \) is the limitation of

\[ f(\gamma) = \gamma^{rac{1}{1-\gamma}}, \]  

(7)

we can determine the number of iteration for truncation depending on the choice of \( \gamma \).

Based on the above analysis, we can regard \( \tilde{y}_{i}^{t} \) as its weighted average value within \( 1/(1 - \gamma) \) iterations. Setting a large \( \gamma \) can weaken the influence of the current local best position on the entire particle swarm as much as possible. For example, let \( \gamma = 0.9 \), \( \tilde{y}_{i}^{t} \) can be viewed as the weighted average value among the local best positions in ten consecutive iterations, i.e. \( y_{i}^{1}, y_{i}^{1-1}, \ldots, y_{i}^{1-9} \). If \( \gamma \) is set smaller, e.g. \( \gamma = 0.5 \), \( \tilde{y}_{i}^{t} \) can be considered as the weighted average value of \( y_{i}^{1} \) and \( y_{i}^{1-1} \). That is, it is only influenced by the local best positions of two previous immediate iterations. It is seen that setting a small \( \gamma \) cannot compensate for the loss of valuable information. Therefore, it is suggested to set \( \gamma \) close to 1.

It is worth noting that if we set the initial value of \( y_{i}^{0} \) to be 0, then \( y_{i}^{1} = (1 - \gamma) y_{i}^{0} \). Since \( \gamma \) is nearly to 1, \( \tilde{y}_{i}^{1} \) is not an accurate estimation for \( y_{i}^{1} \). This is to say that the local best positions of the particle are completely discarded. At the beginning of the PSO iteration, it will cause the distortion of the estimated local best position. To address this problem, we propose to correct it to maintain the accuracy of the estimation by dividing \( \tilde{y}_{i}^{t} \) with \((1 - \gamma^{t})\). That is, we set

\[ \tilde{y}_{i}^{t} = \frac{\tilde{y}_{i}^{t}}{1 - \gamma^{t}}. \]  

(8)
To better explain the bias correction, we consider the relationship between $\tilde{y}_i^t$ and $y_i^t$. Note Eq. (6) can be simplified as

$$\tilde{y}_i^t = (1 - \gamma) \sum_{j=1}^{t} \gamma^{t-j} y_{i}^j$$ (9)

If we take $\tilde{y}_i^t$ as a random variable, its expectation can be obtained as follows:

$$\mathbb{E}[\tilde{y}_i^t] = (1 - \gamma') \sum_{j=1}^{t} \gamma^{t-j} \mathbb{E}[y_{i}^j] = (1 - \gamma') \mathbb{E}[y_{i}^t]$$ (10)

This equation indicates that $\tilde{y}_i^t$ is not an unbiased estimation of $y_i^t$. However, the approximation becomes unbiased if $t$ increases. On the other hand, the corrected history information $\tilde{y}_i^t$ can ensure that the estimated local best position used in the current iteration is an unbiased estimate to the local best position used in Eq. (8).

Through using the corrected local best information, the referenced positions of particles found at each iteration are recorded and new positions can be generated by using the weighted average method. The method avoids particle updating by relying too much on the local best position of the current step, increases the chance that the particles jump out of the local optimum, and reduces the chance of premature convergence to a certain extent.

C. THE CENTRAL LEARNING STRATEGY

The motivation of the proposed central learning strategy comes from the observation of the swarm’s search behavior.

To demonstrate the trajectory change of the local best position $y_i^t$, we show the optimization process of the classical PSO on the two-dimensional Rastrigrin function in Fig. 1. From Fig. 1(a) to (d), the local best position of each particle is shown at the 0-th, 10th, 40th, and 80th iteration, respectively. In the figure, the black circle represents the local best particle, the blue diamond stands for the weighted center of these particles, and the red asterisk represents the global best position. We can find that the distribution of the local best position changes constantly as the search goes. However, the center of the local best (i.e. the blue diamond) is not far from the global optimum (the red asterisk), which indicates that the center of the local best positions reflects some sort of global information.

Further, we show an example search process when optimizing the 30-dimensional Rastrigrin function in Fig. 2, in which the distances between the global best position and the centre of the local best positions to the global optimum solution are plotted against the iteration.

![Figure 1](image1.png)

**FIGURE 1.** The evolution of the local best position during the PSO search process.

**FIGURE 2.** The distance between the center of the global best and the local best to the global optimum. In the figure, the red line is the mean of the local best along the evolution, while the dotted line is that of the global best.

From the figure, we can see that during the search process, the center of the local best positions is closer to the global optimum than that of the global best position, but they gradually converge to the same position as the search iterates. Therefore, it may be useful to take the center of the local best positions as an essential information for the velocity updating.

Based on the above observation, we propose the following central updating strategy. That is, a particle is set to explicitly visit the center of the local best position at each iteration. Particularly, at the $t$-th iteration, we set the $N$-th particle as the central particle:

$$x_N^{t+1} = \frac{1}{N} \sum_{i=1}^{N} \tilde{y}_i^{t+1}$$ (11)

Different from other particles which are updated according to the proposed strategy, the central particle will be updated independently. It will be used only to update the global best position, but not to be used for the updating of other particles’ velocities.

D. PARTICLE SWARM OPTIMIZER BASED ON THE MOVING AVERAGE AND CENTRAL LEARNING STRATEGY

1) THE FREE ELECTRON MODEL

Besides aforementioned strategies, our algorithm is built upon RDPSO [20] which is inspired by the free electron model for conductors that are placed on external electric...
field [53]. According to this model, the movement of an electron is the addition of the thermal and the drift motions. The thermal or random motion always exists, while the drift motion is achieved by moving towards the opposite direction of the external electric field. The electron’s movement will lead it to the position of the minimum potential energy in the conductor.

Since there may exist many local minimum potential energies, the drift motion itself may drive the electron to local minimum, which is analogous to the situation that an optimizer converges to a local minimum. On the other hand, the random motion can help the electron escape a local minimum potential energy. Therefore, the movement of the electron can be considered to be a process of minimizing its potential energy. Particularly, the thermal motion guides the global search of the particle, while the drift motion realizes the local search.

2) THE PROPOSED ALGORITHM

The convergence analysis [20] proved that the $i$-th particle converges to its local attractor $p_i^t = (p_{i1}^t, p_{i2}^t, \cdots, p_{id}^t)$ which can be written as follows:

$$p_i^t = \phi_i^t \odot y_i^t + (1 - \phi_i^t) \odot g^t$$

where $\phi_i^t = [\phi_{i1}^t, \cdots, \phi_{id}^t]^T$ is a random number vector and each element is uniformly sampled in [0, 1], and $\odot$ means element-wise multiplication.

Moving towards the local attractor $p_i^t$ reflects the local searchability of PSO. It is seen that only the information at the current iteration is taken into consideration in Eq. (12). From the discussion in Section II-B, we know that using previous information, i.e. the moving average of the local best positions of each particle, can help the particles escape from local optimum. Therefore, we propose to update the local attractor for the first $N - 1$ particles as follows: for $i = 1$ to $N - 1$, define

$$p_i^t = \phi_i^t \odot y_i^t + (1 - \phi_i^t) \odot g^t$$

where $\tilde{y}_i^t$ is defined as in Eq. (5) and Eq. (8).

The drift motion of a particle can be considered as the movement towards the local attractor, which is analogous to the combination of the cognitive and social recognition parts. In the proposed algorithm, to analogy the electron’s movement, we use the random motion to replace the momentum part in the classical PSO. By combing all the motion terms together, including the random term and the drift term, the velocity of a particle $i$ at the $j$-th dimension in the proposed algorithm can be written as follows:

$$v_i^t = v_{rij}^t + v_{dj}^t, \quad 1 \leq i \leq N - 1, \quad 1 \leq j \leq d$$

where $v_{rij}^t$ and $v_{dj}^t$ represent the random and drift term, respectively. In the sequel, we use

$$v_i^t = VR_i^t + VD_i^t$$

(15)

to represent the vector form of the movement of particle $i$ where $VR_i^t = [v_{r1i}, \cdots, v_{rdi}]^T$ and $VD_i^t = [v_{d1i}, \cdots, v_{didi}]^T$.

Further, we assume that $v_{rij}^t$ follows the Maxwell velocity distribution law [54], which means $v_{rij}^t$ follows a zero-mean normal distribution as follows:

$$f_{v_{rij}^t}(v) = \frac{1}{\sqrt{2\pi}\sigma_{ij}^t} e^{-\frac{1}{2}\frac{v_i^2}{\sigma_{ij}^t}}$$

(16)

where $\sigma_{ij}^t$ is the standard deviation of the distribution. $v_{rij}^t$ can be sampled as follows:

$$v_{rij}^t = \sigma_{ij}^t \cdot \psi_{ij}^t$$

(17)

where $\psi_{ij}^t$ is a random number sampled from the standard Gaussian distribution. To make $\sigma_{ij}^t$ adapt to the optimization process of the particle swarm, we propose to adjust it by using previously collected information. Let $C_j^t = (C_{1j}^t, C_{2j}^t, \cdots, C_{dj}^t)$ where each component is defined as:

$$C_{ij}^t = \frac{1}{N - 1} \sum_{i=1}^{N-1} \tilde{y}_{ij}^t, \quad 1 \leq j \leq d.$$ 

(18)

The standard deviation $\sigma_{ij}^t$ can then be updated as follows:

$$\sigma_{ij}^{t+1} = \alpha \left| C_{ij}^t - x_{ij}^t \right|$$

(19)

where $\alpha$ is the thermal coefficient. Then, $v_{rij}^{t+1}$ is updated as:

$$v_{rij}^{t+1} = \alpha \left| C_{ij}^t - x_{ij}^t \right| \cdot \psi_{ij}^{t+1}$$

(20)

It is expected that the drift velocity component $v_{dij}^t$ realizes the local search of particles. As mentioned above, this can be achieved by moving towards the local attractors. To realize this, we express $v_{dij}^t$ as follows:

$$v_{dij}^t = \beta(p_{ij}^t - x_{ij}^t)$$

where $\beta$ is the drift coefficient.

In summary, the updated formula for each particle $i$, $1 \leq i \leq N - 1$ at the $j$-th dimension is written as follows:

$$\begin{align*}
  v_{ij}^{t+1} &= \alpha \left| C_{ij}^t - x_{ij}^t \right| \psi_{ij}^t + \beta(p_{ij}^t - x_{ij}^t); \\
  x_{ij}^{t+1} &= x_{ij}^t + v_{ij}^{t+1},
\end{align*}$$

(21)

while the $N$-th particle is updated as shown in Eq. (11).

By combining the aforementioned updating strategies, the proposed algorithm, named PSO with moving average and central learning strategies (MAPSO), can be summarized in Alg. 1. In the algorithm, first a swarm of particles are randomly initialized as $X^0$ (line 1). We then set the local best $y_i^0, 1 \leq i \leq N - 1$ to be the initialized particle (line 2) and compute the central position (line 4). The algorithm iterates from line 6 to 19 until the termination criterion has been met. At each iteration, the particles are updated from line 7 to 11. With the updated particles, the local best positions and the central position are computed respectively in line 12 and 13. The new particle swarm is then formed in line 15. The new corrected local best position is updated in
In the current year, deposit-taking financial institutions may participate in the bank lending market. Within the issuance and filing quota, companies, fund products, and members of the national interbank market, their highest degree of interbank marketization. Commercial banks, and they also are the source of financing with the primary source of liabilities for commercial banks, and they also are the source of financing with the highest degree of interbank marketization.

Customer deposits, interbank liabilities, and borrowings from the central bank. The primary source of bank liabilities is customer deposits, while interbank liabilities include the certificates of deposit and financial debts, both of which are important supplementary sources of liabilities for commercial banks, and they also are the source of financing with the highest degree of interbank marketization.

The three main components of bank liabilities include customer deposits, interbank liabilities, and borrowings from the central bank. The primary source of bank liabilities is customer deposits, while interbank liabilities include the certificates of deposit and financial debts, both of which are important supplementary sources of liabilities for commercial banks, and they also are the source of financing with the highest degree of interbank marketization.

As a certification for time deposit issued by depository financial institutions in the national interbank market, their investment and transaction entities are fund management companies, fund products, and members of the national interbank lending market. Within the issuance and filing quota in the current year, deposit-taking financial institutions may determine the terms and amounts of every interbank certificate, but the total amount issued cannot be less than a certain amount, e.g., RMB 50 million in China. In most cases, the time limit of the certificate is within a year.

Interbank certificates of deposit can be circulated in the interbank market before expiration, and the market quotation is determined by a group of banks. In China, eighteen commercial banks with higher credit ratings form a Shibor (Shanghai Interbank Offered Rate) quotation bank group and they determine the official quotations of different varieties of Shibor. Shibor has a total of eight varieties, including overnight, one week, two weeks, one month, three months, six months, nine months, and one year. Taking the official Shibor offer on May 14, 2021, as an example, its quotation form is shown in Fig. 3:

**Algorithm 1 PSO With Moving Average and Central Learning Strategies**

**Require:** The maximum number of iterations \((\text{MaxIter})\) and the parameters \(\alpha\) and \(\beta\);

**Ensure:** The found best solution \(x^*\);

1. Initialize a swarm of \(N - 1\) particles \(X^0 = [x_1^0, \ldots, x_{N-1}^0]\) randomly;
2. Set \(y_i^0 = x_i^0, 1 \leq i \leq N - 1\);
3. Compute \(p_i^t\) and \(C_j^t\) by Eq. (12) and (18), respectively, for \(1 \leq i \leq N - 1\);
4. Compute the \(N\)-th particle \(x_N^0\) by Eq. (11);
5. Set \(t = 0\);
6. while \(t < \text{MaxIter}\) do
   7. for \(i = 1 \rightarrow N - 1\) do
      8. Compute \(v_i^t\) according to Eq. (15);
      9. \(x_i^{t+1} = x_i^t + v_i^t\);
     10. Evaluate \(f(x_i^{t+1})\);
   11. end for
   12. Update \(y_i^{t+1}\) by Eq. (3);
   13. Compute \(X_{N}^{t+1}\) according to Eq. (11);
   14. Evaluate \(f(x_i^{t+1})\);
   15. Form a new swarm \(X^{t+1} = [x_1^{t+1}, \ldots, x_N^{t+1}]\);
   16. Compute \(y_i^t, 1 \leq i \leq N - 1\) by Eq. (5) and Eq. (8);
   17. Compute \(p_i^t\) and \(C_j^t\) by Eq. (12) and (18), respectively, for \(1 \leq i \leq N - 1\);
   18. \(t = t + 1\);
   19. end while
20. Return \(x^* = \arg \max f(X')\).

line 16 while the local attractor and central position is updated in line 17. After termination, the best solution found so far is returned (line 20).

**III. THE YIELD CURVE ESTIMATION PROBLEM**

In this paper, we intend to apply the proposed algorithm to a real-world problem, namely the yield curve estimation problem. In this section, we briefly introduce the problem’s background and its optimization modeling.

**A. BACKGROUND**

The three main components of bank liabilities include customer deposits, interbank liabilities, and borrowings from the central bank. The primary source of bank liabilities is customer deposits, while interbank liabilities include the certificates of deposit and financial debts, both of which are important supplementary sources of liabilities for commercial banks, and they also are the source of financing with the highest degree of interbank marketization.

As a certification for time deposit issued by depository financial institutions in the national interbank market, their investment and transaction entities are fund management companies, fund products, and members of the national interbank lending market. Within the issuance and filing quota in the current year, deposit-taking financial institutions may determine the terms and amounts of every interbank certificate, but the total amount issued cannot be less than a certain amount, e.g., RMB 50 million in China. In most cases, the time limit of the certificate is within a year.

Interbank certificates of deposit can be circulated in the interbank market before expiration, and the market quotation is determined by a group of banks. In China, eighteen commercial banks with higher credit ratings form a Shibor (Shanghai Interbank Offered Rate) quotation bank group and they determine the official quotations of different varieties of Shibor. Shibor has a total of eight varieties, including overnight, one week, two weeks, one month, three months, six months, nine months, and one year. Taking the official Shibor offer on May 14, 2021, as an example, its quotation form is shown in Fig. 3:

**FIGURE 3. The interest rates of the shibor varieties.**

**B. THE OPTIMIZATION MODEL FOR THE YIELD CURVE ESTIMATION PROBLEM**

Macroeconomists and financial market practitioners are very much interested in modeling the term structure of government bonds and interbank offered rates. It is critical for bond and derivatives pricing, risk management, and revealing market expectations, which are critical for monetary policy decision-making.

A yield curve is a graphical depiction of the term structure of interest rates which is a function of time. It is a remarkably meaningful financial tool used in investment and policy decisions. Yield curve estimation is basically a nonlinear parameter optimization problem. Market expectations and conditions are reflected in the term structure of interest rates. Therefore, its estimate should be as accurate as possible to ensure that the information it describes reflects the true market conditions and prospects.

Various efficient structural models have been proposed to estimate the term-rate structure, such as the Nelson-Siegel (NS) model [55], the SV model [56] and so on. Among them, the NS model is one of the most popular term structure models in academia, practitioners, and central banks. The solution accuracy of these nonlinear models directly affects the level of returns in practical applications.
In this paper, we adopt the NS model to transform the yield curve estimation problem into an optimization problem.

1) THE NELSON-SIEGEL MODEL
The NS model was firstly proposed by Nelson and Siegel in 1987 [55]. It involves four parameters with economic meaning, and can describe the changes and states of the interest rate curve profoundly. In the following, we briefly introduce the NS model. First of all, the forward instantaneous interest rate function can be derived as follows:

\[ r(t) = \beta_0 + \beta_1 e^{-\frac{t}{\tau_1}} + \beta_2 \frac{t}{\tau_1} e^{-\frac{t}{\tau_1}} \]  

(22)

where \( \theta = (\beta_0, \beta_1, \beta_2, \tau_1) \) is the model parameter. Since the spot rate and the forward rate have the following relationship:

\[ R(t) = \frac{1}{t} \int_0^t r(x)dx \]  

(23)

and \( r(0) = R(0), r(\infty) = R(\infty) \). The spot rate can then be calculated as:

\[ R(t) = \beta_0 + \beta_1 \left[ 1 - e^{-\frac{t}{\tau_1}} \right] + \beta_2 \left[ 1 - e^{-\frac{t}{\tau_1}} - e^{-\frac{t}{\tau_1}} \right] \]  

(24)

and

\[ \lim_{t \to 0} R(t) = \beta_0 + \beta_1, \lim_{t \to \infty} R(t) = \beta_0 \]  

(25)

In the estimated parameters, the slope factor of the parameter \( \beta_0 \) is 1. It does not decay over time, i.e., the effect of \( \beta_0 \) on all term rates is the same and its movement will cause the yield curve to move up and down horizontally. The slope factor of the parameter \( \beta_1 \) is \( \frac{1 - e^{-\frac{t}{\tau_1}}}{\tau_1} \) and \( \tau_1 \) is varied from 1 to 10. Thus, we have

\[ \lim_{t \to 0} \frac{1 - e^{-\frac{t}{\tau_1}}}{\tau_1} = 1, \lim_{t \to \infty} \frac{1 - e^{-\frac{t}{\tau_1}}}{\tau_1} = 0 \]  

(26)

This shows that the initial value of the slope factor of \( \beta_1 \) is 1 and it decays to 0 at a rather fast rate along the increase of time, which indicates that \( \beta_1 \) has a greater impact on short-term interest rate. It represents the slope factor and affects the slope of the interest rate curve. On the other hand, the slope factor of the parameter \( \beta_2 \) is \( \frac{1 - e^{-\frac{t}{\tau_1}} - e^{-\frac{t}{\tau_1}}}{\tau_1} \), which is a function that increases with time and then gradually decays to 0. This parameter has a weak influence on the short and long ends of the interest rate curve structure and has a greater impact on the mid-end interest rate. This character makes \( \beta_2 \) be very suitable for representing the slope of the interest rate curve. \( \tau_1 \) is the decay degree of the slope factors of \( \beta_1 \) and \( \beta_2 \). The smaller \( \tau_1 \), the faster the decay rate of the slope factor and vice versa.

2) OPTIMIZATION MODELING
By optimizing the yield to maturity (YTM) in the past period, the four parameters of the NS model are updated so that the yield curve is expected to be close to the actual term interest rate structure of the secondary market. According to the updated NS model of the short-term arbitrage, the specific pricing process can be divided into the following three steps:

- Applying the NS model to fit the interest rate term structure and obtain the initial values of the parameters of the NS model according to the quotation reference of Shibor’s official secondary market circulation;
- Using the transaction data of interbank deposit receipts in the secondary market on a certain day to build the objective function and optimize the parameters of the NS model;
- The required YTM can be updated using an NS model with the estimated parameters to perform short-term arbitrage.

The parameters of the NS model are obtained by fitting it to the official quotation of the interbank offered rate, while the quotation of the actual transaction certificate of deposit is not used. In other words, we assume that there is a certain difference between the official quotation and the actual transaction quotation. Our goal is to correct the four parameters of the NS model through the transaction data of the actual certificate of deposit. The objective function can then be written as follows:

\[ \sum_{i=1}^{n} \left( \frac{(pr_i^s - pr_i)}{n} \right)^2 \times \omega_i^2, i = 1, 2, \ldots, n \]  

(27)

where \( \omega_i \) is the weight of a single interbank deposit receipt traded in the total market volume, \( pr_i^s \) is the actual price and \( pr_i \) is the theoretical price, its discount function is calculated as follows,

\[ pr_i = \frac{100}{1 + \frac{R(t) \times 360}{36000}} \]  

(28)

Substituting the above equations, we obtain the final optimization problem, which is shown as follows:

\[ \min_{\theta \in \mathbb{R}^d} \ g(\theta) = \sum_{i=1}^{n} \left( \frac{\omega_i^2}{n} \times \left( pr_i^s - \frac{36000}{R(t) + 360} \right) \right) \]  

(29)

IV. EXPERIMENTAL RESULTS
In this section, we first compare the proposed algorithm with some well-known PSOs. Then the effectiveness of the proposed strategies is studied.

In the experiments, the single-objective functions in the 2017 IEEE Congress on Evolutionary Computing (CEC2017) are used as benchmarks. The function set contains a total of 30 test problems with different characteristics. In our experiment, F2 is excluded since the function is not computationally stable.

A. COMPARISON WITH WELL-KNOW PSOS
The proposed algorithm, named PSO with moving-average and central learning strategies (MAPSO), is compared with several known PSO algorithms. The compared algorithms include
TABLE 1. The parameter settings for the compared algorithms.

| Algorithms | The parameters |
|------------|----------------|
| FIPS       | $\chi = 0.729, c = 2$ |
| DNLSPO     | $w = 0.9 \sim 0.4, c_1 = c_2 = 1.49, m = 3, g = 5$ |
| RDPSO      | $\alpha = 0.7, \beta = 1.5$ |
| SRPSO      | $w = 0.9 \sim 0.4, c_1 = c_2 = 1.49, \lambda = 0.5$ |
| MAPSO      | $c_1 = c_2 = 2, \alpha = 0.7, \beta = 1.5, m = 5$ |

- FIPS [36] which uses all the information in a particle’s neighborhood to update itself based on different topologies;
- DNLSPO [38] which proposes a dynamic neighborhood learning strategy to improve the swarm’s diversity;
- SRPSO [22] which uses different strategies to update the best particle and the rest particles’ velocities;
- RDPSO [20] which is inspired by electron’s motions in conductor.
- EPSO [57] which is a recently proposed PSO with six strategies to address the shortcomings of the classical PSO.

The parameters of these compared algorithms are listed in Table 1, which are the same as those set in the original references. To compare these algorithms fairly, the same experimental parameters are chosen. Specifically, the population size $N = 40$, MaxFES $= 10,000 \times D$, where $D$ is the dimension of the test problem, the search interval is $[-100, 100]$. The termination is that the number of fitness evaluations is greater than MaxFES. The difference between the optimal value found by the algorithm and the actual optimal value at the end of the iteration is called the error value. The error is considered to be 0 when the error value is less than $10^{-8}$.

All the compared algorithms are run 50 times independently on each function and the particles’ velocities are randomly initialized for each run. The means and standard deviation values of the error values are computed and shown in the following tables, where the smallest mean values are typeset in bold. The Wilcoxon’s rank-sum test is used to reflect the statistically-significant differences between the compared algorithms with MAPSO at a significance level of 5%. In this paper, we use symbols ‘+’, ‘−’ and ‘≈’ to represent that MAPSO performs worse than, better than, and similar to the compared algorithms as suggested by the rank sum hypothesis test.

Tables 2 and 3 show the experimental results of the compared algorithms on the CEC2017 test problems with 10D and 30D, respectively. From these tables, we can find that for almost all the tested functions, the mean values of MAPSO are smaller than that of the compared algorithms, which
TABLE 3. Experimental results obtained by MAPSO and the compared algorithms on the 29 test problems with 30 dimensions, where the mean values and the standard deviations are shown. Symbols ‘$+$’, ‘$-$’ and ‘$\approx$’ indicate that MAPSO performs worse than, better than and similar to the compared algorithms according to the the Wilcoxon’s rank sum hypothesis test at a significance level of 5%.

| ins. | MAPSO | RDPSO | FIPS | DNLPSO | SRPSO | EPSO |
|------|-------|-------|------|--------|-------|------|
| F1   | 1.66$e+02$ | 9.03$e+01$ | 7.32$e+01$ | 2.57$e+01$ | 0.00$e+00$ | 3.04$e+01$ | 1.19$e+01$ | 2.38$e+01$ | 6.27$e+01$ | 3.12$e+00$ | 1.13$e+00$ | 7.11$e+01$ | 1.48$e+02$ | 1.38$e+02$ | 3.59$e+02$ | 1.69$e+02$ | 2.62$e+02$ | 4.19$e+02$ | 4.05$e+02$ | 4.82$e+02$ | 3.89$e+02$ | 6.89$e+02$ | 5.24$e+02$ | 5.15$e+02$ | 5.97$e+02$ | 5.04$e+00$ |
|      | 9.53$e+02$ | 2.12$e+00$ | 2.16$e+01$ | 1.26$e+01$ | 6.73$e+01$ | 7.16$e+01$ | 1.16$e+01$ | 2.27$e+01$ | 4.76$e+01$ | 2.30$e+00$ | 2.05$e+00$ | 7.25$e+01$ | 2.12$e+01$ | 2.36$e+01$ | 2.65$e+01$ | 2.75$e+01$ | 4.52$e+01$ | 5.31$e+01$ | 4.16$e+01$ | 9.19$e+01$ | 5.55$e+01$ | 4.79$e+01$ | 2.98$e+01$ | 5.94$e+01$ | 1.54$e+01$ | 2.05$e+01$ | 3.00$e+01$ | 5.10$e+01$ | 3.26$e+01$ |
|      | 2.08$e+02$ | 1.26$e+00$ | 2.72$e+01$ | 1.51$e+01$ | 5.38$e+01$ | 1.98$e+01$ | 1.49$e+01$ | 4.44$e+01$ | 4.87$e+01$ | 2.01$e+01$ | 2.59$e+01$ | 4.63$e+01$ | 2.32$e+01$ | 2.05$e+01$ | 3.56$e+01$ | 1.13$e+01$ | 1.83$e+01$ | 3.00$e+01$ | 4.21$e+01$ | 1.39$e+01$ | 4.24$e+01$ | 2.49$e+01$ | 2.90$e+01$ | 1.79$e+01$ | 1.32$e+00$ | 4.34$e+01$ |
|      | 2.37$e+02$ | 3.35$e+04$ | 3.09$e+00$ | 1.15$e+01$ | 3.09$e+00$ | 3.09$e+00$ | 1.15$e+01$ | 3.77$e+00$ | 4.41$e+00$ | 2.00$e+00$ | 2.59$e+00$ | 4.24$e+00$ | 2.49$e+00$ | 2.90$e+00$ | 1.79$e+00$ | 1.32$e+00$ | 1.83$e+00$ | 3.00$e+00$ | 4.21$e+00$ | 1.39$e+00$ | 4.24$e+00$ | 2.49$e+00$ | 2.90$e+00$ | 1.79$e+00$ | 1.32$e+00$ | 4.34$e+00$ |
|      | 5.85$e+02$ | 3.02$e+02$ | 1.37$e+02$ | 1.05$e+02$ | 8.88$e+01$ | 1.18$e+02$ | 5.41$e+01$ | 3.90$e+00$ | 3.47$e+00$ | 9.14$e+00$ | 4.56$e+00$ | 9.14$e+00$ | 4.56$e+00$ | 9.14$e+00$ | 4.56$e+00$ | 9.14$e+00$ | 4.56$e+00$ | 9.14$e+00$ | 4.56$e+00$ | 9.14$e+00$ | 4.56$e+00$ | 9.14$e+00$ | 4.56$e+00$ | 9.14$e+00$ | 4.56$e+00$ | 9.14$e+00$ |
|      | 3.23$e+02$ | 7.17$e+00$ | 7.17$e+00$ | 7.17$e+00$ | 7.17$e+00$ | 7.17$e+00$ | 7.17$e+00$ | 7.17$e+00$ | 7.17$e+00$ | 7.17$e+00$ | 7.17$e+00$ | 7.17$e+00$ | 7.17$e+00$ | 7.17$e+00$ | 7.17$e+00$ | 7.17$e+00$ | 7.17$e+00$ | 7.17$e+00$ | 7.17$e+00$ | 7.17$e+00$ | 7.17$e+00$ | 7.17$e+00$ | 7.17$e+00$ | 7.17$e+00$ | 7.17$e+00$ | 7.17$e+00$ |

$+/-/\approx$  
0/5/24 5/7/22 6/5/18 4/7/18 8/3/18

indicates that MAPSO achieves a better solution than the compared algorithms in general.

In Table 2, for the 10D problems, it is seen that MAPSO clearly outperforms the compared algorithms on F1. It achieves a 100% success rate on F3, F6 and F9. Further, it can obtain better performance than the compared algorithm on solving simple multimodal functions. In addition, it outperforms the compared algorithms on F12-F19, F22, F24-F26 and F28 which are difficult hybrid and composite functions. The hypothesis test suggests that MAPSO achieves significant improvement over the compared algorithms on at least half of the test functions.

For the 30D problems, MAPSO performs better on F1, F5, F6, F8 and F10, while DNLPSO performs the best on F7, FIPS on F9. But in general, MAPSO reserves an outstanding performance on simple unimodal functions with 10 dimensions. For the hybrid and composite functions (F11-F30), MAPSO performs the best on F13, F15, F17, F19, F20, F22, F23, F26, F28 and F30. In comparison, SRPSO performs the best on F16, F21, F24 and F29. The hypothesis test shows that MAPSO achieves better performance than the compared algorithms in general.

In comparison with EPSO, we see from Table 2 and 3 that EPSO can perform well on the unimodal and multimodal functions (F1-F10), but for the hybrid and composite functions, its performance is clearly worse than MAPSO except few functions (F17, F18 and F21 in 10D, and F24 in 30D). Note that these functions are usually considered to be more challenging than F1-F10. This clearly indicates that MAPSO improves the exploration capability of PSO but not decreasing its exploitation capability.

In comparison with RDPSO, we see that MAPSO outperforms RDPSO significantly in general. The indicates that the proposed strategies can indeed be beneficial to the performance of MAPSO since MAPSO is built upon RDPSO. Since it is claimed in [20] that the exploration ability of RDPSO is

VOLUME 10, 2022

TABLE 4. The APS values obtained by MAPSO and the compared algorithms on the test problems with 10 and 30 dimensions.

|         | MAPSO | FIPS | RDPSO | DNLPSO | SRPSO | EPSO |
|---------|-------|------|-------|--------|-------|------|
| 10D     | 0.38  | 2.90 | 3.72  | 2.59   | 1.86  | 1.38 |
| 30D     | 1.73  | 4.45 | 3.83  | 3.10   | 2.03  | 2.51 |
limited, the better performance of MAPSO implies that the proposed strategies are able to improve its exploration ability which justifies that the proposed strategies are effective. We will carry out more detailed studies in the next subsection to justify the effectiveness of the proposed strategies.

To better demonstrate the performances of the compared algorithms, we apply the average performance score (APS) [58] to give an overall performance on the test function set. If there are totally \( n \) compared algorithms and \( M \) functions in the set, and algorithm \( A_i (i \in \{1, \cdots, n\}) \) outperforms algorithm \( A_j (j \in \{1, \cdots, n\} \setminus \{i\}) \) on the \( m \)-th function \( f_m (m \in \{1, \cdots, M\}) \) with a statistical significance, then set \( \delta_{ij} = 0 \). Otherwise, \( \delta_{ij} = 1 \). When comparing to all other algorithms, the performance score for \( A_i \) on the test function \( f_m \) equals the sum of \( \delta_{ij} \). That is,

\[
P_m(A_i) = \sum_{j \in \{1, \cdots, n\} \setminus \{i\}} \delta_{ij}, \tag{30}
\]

The APS value of \( A_i \) on the entire function set is \( APS(A_i) = \frac{1}{M} \sum_{m=1}^{M} P_m(A_i) \). It is seen that a smaller value of APS for \( A_i \) means that the algorithm \( A_i \) has a better overall performance.

The overall ranking on the test suite in terms of APS is represented in Table 4. It is seen from the table that MAPSO obtains the highest APS values both on 10D and 30D test functions. Thus we may conclude that MAPSO performs the best among the compared algorithms.

To make the comparison more clearer, we show the convergence properties of the compared algorithms in Fig. 4 and Fig. 5, respectively, on the CEC 2017 test functions with 10 and 30 dimensions. From the figure, it is seen that MAPSO displays an obvious advantage over other algorithms in both early and final iterations in general. As shown in Fig. 4 for F5, F8, F10, F16 and F30, MAPSO can gradually escape from a local optimum once trapped. This confirms its exploration capability. In Fig. 5, such an exploration capability can be easily seen on functions F5, F8, F10, and F21.
B. COMPONENT ANALYSIS

In this section, we investigate the effectiveness of the proposed strategies in terms of exploration and exploitation. To carry out the study, we compare the following algorithms. The first algorithm is the proposed algorithm without the central learning strategy, which is named as MAPSO/wcCL. The second algorithm is the proposed algorithm without the moving average strategy, which is named as MAPSO/wcMA. These algorithms are compared with MAPSO on the CEC 2017 test suite with 10 dimensions.

Tables 5 summarizes the obtained results. In the table, the means and standard deviations of the obtained results averaged over 50 independent runs are listed. We also include the rank sum hypothesis test results obtained by comparing with MAPSO and RDPSO. The reason to compare with RDPSO is that MAPSO is built upon RDPSO. Comparing with RDPSO enables us to see which strategy is more effective for the improvement of RDPSO. To save space, we omit the obtained results by MAPSO and RDPSO in the table since these results are available in Table 2.

The same as previous experimental settings, symbols ‘+’, ‘−’ and ‘≈’ indicate that MAPSO performs worse than, better than and similar to the compared algorithms according to the Wilcoxon’s rank sum hypothesis test at a significance level of 5%. Furthermore, we use symbols ‘†’, ‘‡’ and ‘≈’ to indicate the performance comparison with RDPSO in terms of the rank sum hypothesis test.

From Table 5, it is seen that according to the rank sum hypothesis test, without the central learning strategy,
TABLE 5. Experimental results obtained by MAPSO/wcCL and MAPSO/wcMA on the 29 test problems with 10 dimensions, where the mean values and the standard deviations are shown. Symbols ‘+’, ‘−’ and ‘≈’ (resp. †, ‡ and ≈) indicate that MAPSO (resp. RDPSO) performs worse than, better than and similar to the compared algorithms according to the Wilcoxon’s rank sum hypothesis test at a significance level of 5%.

| ins. | MAPSO/wcMA          | MAPSO/wcCL          | MAPSO/wcMA          | MAPSO/wcCL          |
|------|---------------------|---------------------|---------------------|---------------------|
| F1   | 2.56e + 02          | 1.93e + 02          | 3.57e + 01          | 3.27e + 01          |
| F3   | 1.00e − 6            | 1.83e − 6           | 4.05e + 02          | 5.97e + 02          |
| F4   | 3.31e + 00           | 3.54e + 00          | 3.14e + 02          | 2.63e + 02          |
| F5   | 6.28e + 00           | 7.91e + 00          | 1.63e + 01          | 1.34e + 00          |
| F6   | 3.59e − 3            | 1.35e − 04          | 3.54e + 02          | 1.62e + 02          |
| F7   | 9.11e + 01           | 8.32e + 01          | 6.96e + 01          | 5.01e + 02          |
| F8   | 7.06e + 00           | 5.66e + 00          | 6.93e + 02          | 3.14e + 02          |
| F9   | 4.03e − 02           | 1.72e − 07          | 1.59e + 02          | 1.23e + 02          |
| F10  | 3.84e + 02           | 4.63e + 02          | 4.53e + 02          | 4.00e + 02          |
| F11  | 1.65e + 01           | 1.12e + 01          | 2.90e + 01          | 2.50e + 01          |
| F12  | 2.59e + 04           | 2.31e + 04          | 4.03e + 02          | 3.54e + 02          |
| F13  | 3.97e + 02           | 4.73e + 02          | 3.00e + 02          | 2.95e + 02          |
| F14  | 5.01e + 01           | 4.03e + 01          | 2.93e + 02          | 2.60e + 02          |
| F15  | 3.82e + 02           | 3.08e + 02          | 6.01e + 04          | 5.94e + 04          |
| F16  | 1.59e + 01           | 1.57e + 01          | 23/2/4              | 25/2/2              |

+/-\%/— 3/5/21 5/9/15 1/3/1/4

FIGURE 6. The corrected yield curve structure obtained by using the obtained parameters to correct the NS model.

MAPSO/wcCL achieves only three significantly better results than MAPSO, but 23 significantly better results than RDPSO. On the other hand, without the moving average strategy, MAPSO/wcMA achieves five significantly better results than MAPSO, but 25 better results than RDPSO. This clearly indicates that each proposed strategy can improve the performance of RDPSO. We notice that the disadvantage of RDPSO lies on its exploration capability since the random term itself may be not effectively enough to help the search escape from local minimum. In consideration of the better performance of MAPSO/wcCL and MAPSO/wcMA, we may conclude that the proposed strategies can indeed improve the exploration capability.

The results in Table 5 also indicate that each strategy alone cannot beat the usage of the combined strategies. Particularly, it is seen that for the unimodal and multimodal functions (F1-F10), MAPSO/wcMA and MAPSO/wcCL perform similarly. However, for the hybrid functions (F11-F20), MAPSO/wcMA achieves one more better than MAPSO/wcMA in comparison with MAPSO. For the composition functions (F21–F30), however, MAPSO/wcCL achieves one more significantly better, and five similar results than MAPSO/wcMA. Since in MAPSO/wcCL, only the moving average strategy is used, we may conclude that this strategy is more beneficial to improve the performance of PSO than the central learning strategy.

V. APPLICATION ON THE YIELD CURVE ESTIMATION PROBLEM

In this section, we apply MAPSO on the yield curve estimation problem taking Shanghai interbank offered rate (Shibor) as an example, and compare it with a variety of algorithms, including GA [59], DE [60], SRPSO [22] and some classical optimization algorithms. The classical algorithms include the sequential least squares programming (SL-SQP) [61], L-BFGS-B [62] and Nelder-Mead [63].

Table 6 shows the obtained results including the means and standard deviations obtained on the problem with 50 different weight settings. The compared algorithms were allowed to run for 10 seconds. The Wilcoxon rank sum test was carried out at a significance level of 5%.

The hypothesis test suggests that MAPSO perform significantly better than all the compared algorithms. It is seen from the table that MAPSO obtains not only smaller mean values, but rather small variances. This implies that MAPSO is very stable in solving the yield curve estimation problem. It performs even better than the classical algorithms, such as SL-SQP, BFGS, and L-BFGS-B. Notice that these classical optimization algorithms are often considered to be more effective than population-based algorithms.
TABLE 6. Experimental results obtained by different algorithms for the interbank arbitrage problems.

| Algorithms  | mean     | std      |
|------------|----------|----------|
| MAPSO      | 6.43e-06 | 2.04e-05 |
| GA         | 6.88e-05 | 1.48e-05 |
| SRPSO      | 1.75e-05 | 5.21e-05 |
| SL-SQP     | 3.80e-01 | 1.68e-01 |
| L-BFGS-B   | 1.48e-01 | 2.09e-01 |
| BFGS       | 2.68e-01 | 2.20e-01 |
| Nelder-Mead| 5.43e-05 | 3.00e-04 |

Using the obtained optimal parameters by MAPSO to correct the NS model, we can obtain the new yield curve structure as shown in Fig. 6. In the figure, the red dots represent the official Shibor price, while the blue dots are the actual transaction data in the previous day. The green line represents the interest rate time structure corrected by using the actual transaction data. From the figure, we see that the overnight, one week and fortnight Shibor prices (red dots) usually are lower than the actual transaction interest rates (blue dots). However, the term structure of interest rates after our correction fit the actual transaction better for the secondary market. The corrected term structure provides very useful value reference for recent short-term arbitrage operation, which is exactly the motivation for us to find the global optimum of Problem 29.

VI. CONCLUSION

In this paper, we proposed a novel PSO algorithm, named MAPSO, based on the moving average strategy and central learning strategy, inspired by the movement of electron in conductor in an external electric field. Firstly, the moving average of the local best positions was proposed to take effect on the random movement of particles aiming to improve its exploration capability, while the centre position of all the local best positions was used to update the global best position. In the experiments, MAPSO was compared with several known PSOs on the widely-used CEC 2017 test suite with 10 and 30 dimensions. The experimental results revealed that MAPSO performs significantly better than the compared algorithms in general. Finally, MAPSO was applied to a challenging problem in macro-economy, the yield curve estimation problem. The comparison with a variety of algorithms showed that MAPSO achieved the state-of-the-art performance.

REFERENCES

[1] J. Kennedy, “Particle swarm optimization,” in Proc. IEEE Int. Conf. Neural Netw., Perth, WA, Australia, vol. 4, no. 8, Dec. 2011, pp. 1942–1948.
[2] T. Shami, A. El-Saleh, M. Alsawati, Q. Al-Tashi, M. Summailah, and S. Mirjalili, “Particle swarm optimization: A comprehensive survey,” IEEE Access, vol. 10, pp. 10031–10061, 2022.
[3] M. R. Bonyadi and Z. Michalewicz, “Particle swarm optimization for single objective continuous space problems: A review,” Evol. Comput., vol. 25, no. 1, pp. 1–54, Mar. 2017.
[4] A. R. Jordehi, “Parameter selection in particle swarm optimization: A survey,” J. Exp. Theor. Artif. Intell., vol. 25, nos. 527–542, pp. 591–600, 2013.
[5] Y. Shi and R. Eberhart, “A modified particle swarm optimizer,” in Proc. IEEE Int. Conf. Evol. Comput. IEEE World Congr. Comput. Intell., May 1998, pp. 69–73.
[6] Y. Shi and R. C. Eberhart, “Empirical study of particle swarm optimization,” in Proc. Congr. Evol. Comput., vol. 3, Washington, DC, USA, Jul. 1999, pp. 1945–1950.
[7] X. Yang, J. Yuan, J. Yuan, and H. Mao, “A modified particle swarm optimizer with dynamic adaptation,” Appl. Math. Comput., vol. 189, no. 2, pp. 1205–1213, 2007.
[8] J. Bansal, P. Singh, and M. Saraswat, “Itera weight strategies in particle swarm optimization,” in Proc. 3rd World Congr. Nature Biologically Inspired Comput., 2011, pp. 640–647.
[9] M. Taherkhani and R. Safabakhsh, “A novel stability-based adaptive inertia weight for particle swarm optimization,” Appl. Soft. Comput., vol. 38, pp. 281–295, Jan. 2016.
[10] J. Li, J. Zhang, C. Jiang, and M. Zhou, “Composite particle swarm optimizer with historical memory for function optimization,” IEEE Trans. Cybern., vol. 45, no. 10, pp. 2350–2363, Oct. 2015.
[11] A. Nickabadi, M. M. Ebadzadeh, and R. Safabakhsh, “A novel particle swarm optimization algorithm with adaptive inertia weight,” Appl. Soft Comput., vol. 11, no. 4, pp. 3658–3670, Jun. 2011.
[12] A. K. Qin, V. L. Huang, and P. N. Suganthan, “Differential evolution algorithm with strategy adaptation for global numerical optimization,” IEEE Trans. Evol. Comput., vol. 13, no. 2, pp. 398–417, Apr. 2009.
[13] W. Liu, Z. Wang, Y. Yuan, N. Zeng, K. Hone, and X. Liu, “A novel sigmoid-function-based adaptive weighted particle swarm optimizer,” IEEE Trans. Cybern., vol. 51, no. 2, pp. 1085–1093, Feb. 2021.
[14] K. Kassoul, N. Zufferey, N. Cheikhrouhou, and S. Brahimi Belhaouari, “Exponential particle swarm optimization for global optimization,” IEEE Access, vol. 10, pp. 78320–78344, 2022.
[15] J. Kennedy, “Bare bones particle swarms,” in Proc. IEEE Swarm Intell. Symp. (SIS), Apr. 2003, pp. 80–87.
[16] J. Guo, B. Shi, K. Yan, J. Tang, H. Xiao, and Y. Sato, “A twining bare bones particle swarm optimization algorithm,” PLoS ONE, vol. 17, no. 5, 2022, Art. no. e0267197.
[17] T. J. Richer and T. M. Blackwell, “The Lévy particle swarm,” in Proc. IEEE Int. Conf. Evol. Comput., Jul. 2006, pp. 808–815.
[18] H. Hakli and H. Uğuz, “A novel particle swarm optimization algorithm with Levy flight,” Appl. Soft Comput., vol. 23, pp. 333–345, Oct. 2014.
[19] J. J. Liang, A. K. Qin, P. N. Suganthan, and S. Baskar, “Comprehensive learning particle swarm optimizer for global optimization of multimodal functions,” IEEE Trans. Evol. Comput., vol. 10, no. 3, pp. 281–295, Jun. 2006.
[20] J. Sun, X. Wu, W. Fang, Y. Shi, and V. Palade, “Random drift particle swarm optimization algorithm: Convergence analysis and parameter selection,” Mach. Learn., vol. 101, nos. 1–3, pp. 345–376, 2015.
[21] R. Cheng and Y. Jin, “A social learning particle swarm optimization algorithm for scalable optimization,” Inf. Sci., vol. 291, pp. 43–60, Jan. 2015.
[22] M. R. Tanweer, S. Suresh, and N. Sundararajan, “Self regulating particle swarm optimization algorithm,” Inf. Sci., vol. 294, pp. 182–202, Feb. 2015.
[23] A. A. Karim, N. A. Mat Isa, and W. H. Lim, “Modified particle swarm optimization algorithm,” Inf. Sci., vol. 189, no. 2, pp. 281–295, 2015.
[24] X. Zhang, X. Wang, Q. Kang, and J. Cheng, “Differential mutation and novel social learning particle swarm optimization algorithm,” Inf. Sci., vol. 480, pp. 109–129, Apr. 2019.
[25] W. Li, X. Meng, Y. Huang, and Z.-H. Fu, “Multipopulation cooperative particle swarm optimization with a mixed mutation strategy,” Inf. Sci., vol. 529, pp. 179–196, Aug. 2020.
[26] D. Sedighizadeh, E. Masehian, M. Sedighizadeh, and H. Akbaripour, “GEPSO: A new generalized particle swarm optimization algorithm,” Math. Comput. Simul., vol. 179, pp. 194–212, Jan. 2021.
[27] Y. Xu and D. Pi, “A reinforcement learning-based communication topology in particle swarm optimization,” Neural Comput. App., vol. 32, pp. 1007–10032, Jul. 2020.
[28] C.-C. Peng and C.-H. Chen, “Compensatory neural fuzzy network with symbiotic particle swarm optimization for temperature control,” Appl. Math. Model., vol. 39, no. 1, pp. 383–395, Jan. 2015.
[29] M. R. Tanweer, S. Suresh, and N. Sundararajan, “Dynamic mentoring and self-regulation based particle swarm optimization algorithm for solving complex real-world optimization problems,” Inf. Sci., vol. 326, pp. 1–24, Jun. 2016.
JIAQI ZHANG received the B.Sc. degree in human resource management from Shaanxi Normal University, Xi’an, China, in 2015, and the M.Sc. degree in management from the University of Bristol, U.K., in 2016. She is currently working with the Xi’an University of Finance and Economics, Xi’an. Her research interests include evolutionary algorithms and its application in finance and economics.

BO SHI received the Ph.D. degree in economics from Nanjing University, Nanjing, China. He is a Full Professor of economics with the Department of Mathematical Economics and Statistics, Northwest University, Xi’an, China. He has authored over 100 publications on mathematical economics and statistics. His research interests include macro-economics, energy economics, and developing political economics.