mathematics

MDPI

Article

Optimal Randomness in Swarm-Based Search

Jiamin Wei [1,2], YangQuan Chen [2,*], Yongguang Yu [1] and Yuquan Chen [2,3]

[1] Department of Mathematics, Beijing Jiaotong University, Beijing 100044, China
[2] School of Engineering, University of California, Merced, 5200 Lake Road, Merced, CA 95343, USA
[3] Department of Automation, University of Science and Technology of China, Hefei 230027, China
* Correspondence: ychen53@ucmerced.edu

Received: 23 July 2019; Accepted: 3 September 2019; Published: 6 September 2019

Abstract: Lévy flights is a random walk where the step-lengths have a probability distribution that is heavy-tailed. It has been shown that Lévy flights can maximize the efficiency of resource searching in uncertain environments and also the movements of many foragers and wandering animals have been shown to follow a Lévy distribution. The reason mainly comes from the fact that the Lévy distribution has an infinite second moment and hence is more likely to generate an offspring that is farther away from its parent. However, the investigation into the efficiency of other different heavy-tailed probability distributions in swarm-based searches is still insufficient up to now. For swarm-based search algorithms, randomness plays a significant role in both exploration and exploitation or diversification and intensification. Therefore, it is necessary to discuss the optimal randomness in swarm-based search algorithms. In this study, cuckoo search (CS) is taken as a representative method of swarm-based optimization algorithms, and the results can be generalized to other swarm-based search algorithms. In this paper, four different types of commonly used heavy-tailed distributions, including Mittag-Leffler distribution, Pareto distribution, Cauchy distribution, and Weibull distribution, are considered to enhance the searching ability of CS. Then four novel CS algorithms are proposed and experiments are carried out on 20 benchmark functions to compare their searching performances. Finally, the proposed methods are used to system identification to demonstrate the effectiveness.

Keywords: optimal randomness; swarm-based search; cuckoo search; heavy-tailed distribution; global optimization

1. Introduction

Swarm-based search algorithms have attracted great interest of researchers in fields of computational intelligence, artificial intelligence, optimization, data mining, and machine learning during the last two decades [1]. Moreover, the swarm intelligence algorithms and artificial intelligence have been successfully used in complex real-life applications, such as wind farm decision system, social aware cognitive radio handovers, feature selection, truck scheduling and so on [2–5]. Up to now, a lot of swarm-based search algorithms have been presented, including artificial bee colony (ABC) [6], cuckoo search (CS) [7], firefly algorithm (FA) [8], particle swarm optimization (PSO) [9] and so on.

Among the existing swarm-based search algorithms, CS is presented in terms of the obligate brood parasitic behavior of some cuckoo species and the Lévy flight behavior of some birds and fruit flies. CS searches for new solutions by performing a global explorative random walk together with a local exploitative random walk. One advantage of CS is that its global search utilizes Lévy flights or process, instead of standard random walks. Lévy flights play a critical role in enhancing randomness, as Lévy flights is a random walk where the step-lengths have a probability distribution that is heavy-tailed. At each iteration process, CS firstly searches for new solutions in Lévy flights

random walk. Secondly, CS proceeds to obtain new solutions in local exploitative random walk. After each random walk, a greedy strategy is used to select a better solution from the current and newly generated solutions according to their fitness values. Due to the salient features such as simple concept, limited parameters, and implementation simplicity, CS has aroused extensive attention and has been accepted as a simple but efficient optimization technique for solving optimization problems. Accordingly, many new CS variants have been continuously presented recently [10–14]. However, there is still a lot of space in designing newly improved or enhanced techniques to help to increase the accuracy and convergence speed and enhance the searching stability for the original CS algorithm.

In nature, the movements of many foragers and wandering animals have been shown to follow a Lévy distribution [15] rather than a Gaussian distribution. It is found that foragers frequently take a large step to enhance their searching efficiency since it is the product of natural evolution over millions of years. Inspired by the mentioned natural phenomena, CS is proposed in combination with Lévy, where the step-length is drawn from a heavy-tailed probability distribution and large steps frequently take place flights. In fact, before CS, the idea of Lévy flights was applied in Reference [16] to solve a problem of non-convex stochastic optimization, due to big jumps of the Lévy flights process. In this way, it can enhance the searching ability compared with the Gaussian distribution where large steps seldom happen. More exactly, we have to say the foragers should move following a heavy-tailed distribution since the Lévy distribution is a simple heavy-tailed distribution which is easy to analyze. There are many other heavy-tailed distributions such as the Mittag-Leffler distribution, Pareto distribution, Cauchy distribution and the Weibull distribution, and large steps still frequently happen when using them to generate the steps. For swarm-based optimization algorithms, randomness plays a significant role in both exploration and exploitation or diversification and intensification [17]. Therefore, it is necessary to discuss the optimal randomness in swarm-based search algorithms.

In this paper, we mainly focus on the discussion on the impact of different heavy-tailed distributions on the performance of swarm-based search algorithms. In the study, CS is taken as a representative method of swarm-based optimization algorithms, and the results can be generalized to other swarm-based search algorithms. At first, some basic definitions of the heavy-tailed distributions and how to generate the random numbers according to the given distribution are provided. Then by replacing the Lévy flight with steps generated from other heavy-tailed distributions, four different randomness-enhanced CS algorithms (namely CSML, CSP, CSC, and CSW) are presented by applying Mittag-Leffler distribution, Pareto distribution, Cauchy distribution and Weibull distribution. Finally, dedicated experimental studies are carried out on a test suite of 20 benchmark problems with unimodal, multimodal, rotated and shifted properties to compare the performance of different variant algorithms. The experimental results demonstrate that the four proposed randomness-enhanced CS algorithms show a significant improvement over the original CS algorithm. This suggests that the performance of CS can be improved by means of integrating different heavy-tailed probability distributions rather than Lévy flights into it. At last, an application problem of parameter identification of unknown fractional-order chaotic systems is further considered. Based on the observations and results analysis, the randomness-enhanced CS algorithms are able to exactly identify the unknown specific parameters of the fractional-order system with better effectiveness and robustness. The randomness-enhanced CS algorithms can be regarded as an efficient and promising tool for solving the real-world complex optimization problems besides the benchmark problems.

The remainder of this paper is organized as follows. The principle of the original CS algorithm is described in Section 2. Section 3 gives details of four randomness-enhanced CS algorithms after a brief review of several commonly used heavy-tailed distributions. Experimental results and discussions of randomness-enhanced CS algorithms are presented in Section 4. Finally, Section 5 summarizes the conclusions and future work.

2. Cuckoo Search Algorithm

Cuckoo search (CS), developed by Yang and Deb, is considered to be a simple but promising stochastic nature-inspired swarm-based search algorithm [7,18]. CS is inspired by the intriguing brood parasitism behaviors of some species of cuckoos, and is enhanced by Lévy flights instead of simple isotropic random walks. Cuckoos are considered to be fascinating birds not only for their beautiful sounds but also for their aggressive reproduction strategy. Some cuckoo species lay their eggs in host nests, and at the same time, they may remove host birds' eggs in order to increase the hatching probability of their own eggs. For simplicity in describing the standard CS, there are three idealized rules as follows [7]: (1) Only one egg is laid by each cuckoo bird at a time, and dumped in a randomly chosen nest; (2) The next generations of cuckoos search for new solutions using the best nests with high-quality; (3) The number of available host nests is fixed, and the egg laid by a cuckoo is discovered by the host bird with a probability $P_a \in [0, 1]$. In this condition, the host bird can either remove the egg or simply abandon the nest and build a completely new nest.

The purpose of CS is to substitute a not-so-good solution in the nests with the new and potentially better solutions (cuckoos). At each iteration process, CS employs a balanced combination of a local random walk and the global explorative random walk under control of a switching parameter P_a. A greedy strategy is used after each random walk to select better solutions from the current and newly generated solutions based on their fitness values.

2.1. Lévy Flights Random Walk

At generation t, a global explorative random walk carried out by using Lévy flights can be defined as follows:

$$U_i^t = X_i^t + \alpha \otimes \text{Lévy} \otimes (X_i^t - X_{best}), \tag{1}$$

where U_i^t denotes a new solution generated in Lévy flights random walk, and X_{best} is the best solution obtained so far. $\alpha > 0$ is the step size related to the scales of the problem of interest, X_{best} is the best solution obtained so far, the product $\otimes$ represents entrywise multiplications, and Lévy(λ) is defined according to a simple power-law formula as follows:

$$\text{Lévy}(\lambda) \sim t^{-1-\lambda}, \tag{2}$$

where t is a random variable, $0 < \lambda \leq 2$ is a stability index. Moreover, it is worth mentioning that the well-known Gaussian and Cauchy distribution are its special cases when its stability index λ is respectively set to 2 and 1.

In practice, Lévy(λ) can be updated as follows:

$$\text{Lévy}(\lambda) \sim \frac{\mu}{|v|^{1/\lambda}}\phi, \tag{3}$$

where λ is suggested as 1.5 [18], μ and v are random numbers drawn from a normal distribution with mean of 0 and standard deviation of 1, $\blacksquare(\cdot)$ denotes the gamma function, and ϕ is expressed as:

$$\phi = \left[\frac{\blacksquare(\lambda) \sin\left(\frac{\pi\lambda}{2}\right)}{\blacksquare\left(\frac{1+\lambda}{2}\right) 2^{\frac{\lambda-1}{2}}} \right]^{1/\lambda}. \tag{4}$$

2.2. Local Random Walk

The local random walk can be defined as:

$$U_i^t = X_i^t + r \otimes H(P_a - \epsilon) \otimes (X_j^t - X_k^t), \tag{5}$$

where X_j^t and X_k^t are two different selected random solutions, r and ϵ are two independent random numbers with uniform distribution, and $H(u)$ is a Heaviside function. The local random walk utilizes a far field randomization to generate a substantial fraction of new solutions which are sufficiently far from the current best solution. The pseudo-code of the standard CS algorithm is given in Algorithm 1.

Algorithm 1 Pseudo code of the standard CS algorithm

Input: Population size NP, fraction probability P_a, dimensionality D, the maximum number of function evaluations Max_FEs, iteration number $t = 1$, objective function $f(X)$.
Output: The best solution.

```
t = 1;
Generate an initial population of NP host nests X_i^t, (i = 1, 2, ..., NP);
Evaluate the fitness value of each nest X_i^t;
FES = NP;
Determine the best nest with the best fitness value;
while FES<Max_FEs do
    // Lévy flights random walk
    for i = 1, 2, ..., NP do
        Generate a new solution U_i^t randomly using Lévy flights random walk according to Equation (1);
        Greedily select a better solution from U_i^t and X_i^t according to their fitness values;
        FES = FES + 1;
    end for
    // Local random walk, a fraction (P_a) of worse nests are abandoned and new ones are built
    for i = 1, 2, ..., NP do
        Search for a new solution U_i^t using local random walk according to Equation (5);
        Greedily select a better solution from U_i^t and X_i^t according to their fitness values;
        FES = FES + 1;
    end for
    Obtain the best solution so far X_best;
    t = t + 1;
end while
```

3. Randomness-Enhanced CS Algorithms

The standard CS algorithm uses Lévy flights in global random walk to explore the search space. The Lévy step is taken from the Lévy distribution which is a heavy-tailed probability distribution. In this case, a fraction of large steps is generated, which plays an important role in enhancing the search capability of CS. Although many foragers and wandering animals have been shown to follow a Lévy distribution [15], the investigation into the impact of other different heavy-tailed probability distributions on CS is still insufficient up to now. This motivates us to make an attempt to apply the well-known Mittag-Leffler distribution, Pareto distribution, Cauchy distribution and Weibull distribution to the standard CS algorithm, by using which, more efficient searches are supposed to take place in the search space thanks to the long jumps. In this section, a brief review of several commonly used heavy-tailed distributions is given and then the scheme of the randomness-enhanced CS algorithms is introduced.

3.1. Commonly Used Heavy-Tailed Distributions

This subsection provides the definition of heavy-tailed distribution and several examples of commonly used heavy-tailed distributions.

Definition 1 (Heavy-Tailed Distribution). *The distribution of a real-valued random variable X is said to have a heavy right tail if the tail probabilities $P(X > x)$ decay more slowly than those of any exponential distribution, that is, if*

$$\lim_{x\to\infty} \frac{P(X > x)}{e^{-\lambda x}} = \infty \tag{6}$$

for every $\lambda > 0$. Heavy left tails are defined in a similar way [19].

Example 1 (Mittag-Leffler Distribution). *A random variable is said to subject to Mittag-Leffler distribution if its distribution function has the following form*

$$F_\beta(x) = \sum_{k=1}^{\infty} \frac{(-1)^{k-1} x^{k\beta}}{\blacksquare(1 + k\beta)}, \tag{7}$$

where $0 < \beta \leq 1$, $x > 0$, and $F_\beta(x) = 0$ for $x \leq 0$. For $0 < \beta < 1$, the Mittag-Leffler distribution is a heavy-tailed generalization of the exponential, and reduces to the exponential distribution when $\beta = 1$.

A Mittag-Leffler random number can be generated using the most convenient expression proposed by Kozubowski and Rachev [20]:

$$\tau_\beta = -\gamma \ln u \left(\frac{\sin(\beta\pi)}{\tan(\beta\pi v)} - \cos(\beta\pi) \right)^{1/\beta}, \tag{8}$$

where γ is the scale parameter, $u, v \in (0, 1)$ are independent uniform random numbers, and τ_β is a Mittag-Leffler random number.

Example 2 (Pareto Distribution). *A random variable is said to subject to Pareto distribution if its cumulative distribution function has the following expression:*

$$F(x) = \begin{cases} 1 - \left(\frac{b}{x}\right)^a, & x \geq b, \\ 0, & x < b, \end{cases} \tag{9}$$

where $b > 0$ is the scale parameter, $a > 0$ is the shape parameter (Pareto's index of inequality).

Example 3 (Cauchy Distribution). *A random variable is said to subject to Cauchy distribution if its cumulative distribution function has the following expression:*

$$F(x) = \frac{1}{\pi} \arctan\left(\frac{2(x - \mu)}{\sigma} \right) + \frac{1}{2}, \tag{10}$$

where μ is the location parameter, σ is the scale parameter.

Example 4 (Weibull Distribution). *A random variable is said to subject to Weibull distribution if it has a tail function F as follows:*

$$F(x) = e^{-(x/\kappa)^\xi}, \tag{11}$$

where $\kappa > 0$ is the scale parameter, $\xi > 0$ is the shape parameter. If and only if $\xi < 1$, the Weibull distribution is a heavy-tailed distribution.

3.2. Improving CS with Different Heavy-Tailed Probability Distributions

For swarm-based search algorithms, randomness plays a significant role in both exploration and exploitation or diversification and intensification [17]. It is very necessary to discuss the optimal randomness in swarm-based search algorithms. Randomness is normally realized by employing

pseudorandom numbers, based on some common stochastic processes. Generally, randomization is achieved by simple random numbers that are drawn from a uniform distribution or a normal distribution. But in other cases, more sophisticated randomization approaches are considered, for example, random walks and Lévy flights. Here, we have to say the foragers should move following a heavy-tailed distribution since Lévy distribution is a simple heavy-tailed distribution which is easy to analyze. There are many other heavy-tailed distributions such as Mittag-Leffler distribution, Pareto distribution, Cauchy distribution, and Weibull distribution, and large steps still frequently happen when using them to generate the steps. In this paper, we mainly focus on the discussion on the impact of different heavy-tailed distributions on the performance of swarm-based search algorithms. In the study, CS is taken as a representative method of swarm-based optimization algorithms, and the results can be generalized to other swarm-based search algorithms.

In this section, four randomness-enhanced cuckoo search algorithms are proposed in this paper. Specifically, the following modified CS methods are considered: (1) CS with the Mittag-Leffler distribution, denoted as CSML; (2) CS with the Pareto distribution, denoted as CSP; (3) CS with the Cauchy distribution, denoted as CSC; (4) CS with the Weibull distribution, referred to CSW. In the modified CS methods, the aforementioned four different heavy-tailed probability distributions are respectively used to be integrated into CS instead of the original Lévy flights in the global explorative random walk. By using these heavy-tailed probability distributions, the updating Equation (1) can be reformulated as follows

$$U_i^t = X_i^t + \alpha \otimes \text{Mittag} - \text{Leffler}(\beta, \gamma) \otimes (X_i^t - X_{best}), \tag{12}$$

$$U_i^t = X_i^t + \alpha \otimes \text{Pareto}(b, a) \otimes (X_i^t - X_{best}), \tag{13}$$

$$U_i^t = X_i^t + \alpha \otimes \text{Cauchy}(\mu, \sigma) \otimes (X_i^t - X_{best}), \tag{14}$$

$$U_i^t = X_i^t + \alpha \otimes \text{Weibull}(\xi, \kappa) \otimes (X_i^t - X_{best}), \tag{15}$$

where $\text{Mittag} - \text{Leffler}(\beta, \gamma)$ in Equation (12) denotes a random number drawn from Mittag-Leffler distribution; $\text{Pareto}(b, a)$ in Equation (13) represents a random number drawn from Cauchy distribution; $\text{Cauchy}(\mu, \sigma)$ in Equation (14) denotes a random number drawn from Cauchy distribution; $\text{Weibull}(\alpha, \kappa)$ in Equation (15) means a random number drawn from Weibull distribution. Compared with the standard CS algorithm, the differences of randomness-enhanced cuckoo search methods lie in line 9 from Algorithm 1.

Remark 1. *In this paper, our emphasis is to study the effects of different heavy-tailed distributions on the swarm-based search algorithms.*

Remark 2. *Since CS is a popular swarm-based search algorithm, we only use it as an representative. Similar analyses for optimal randomness can be applied to other swarm-based algorithms.*

Remark 3. *The source code of randomness-enhanced cuckoo search algorithms (namely CSML, CSP, CSC, CSW), written in Matlab, is available at https://www.mathworks.com/matlabcentral/fileexchange/71758-optimal-randomness-in-swarm-based-search.*

4. Experimental Results

This study focuses on discussing the effectiveness and efficiency of the proposed randomness-enhanced CS algorithms. To fulfill this purpose, extensive experiments are carried out on a test suite of 20 benchmark functions. The superiority of randomness-enhanced CS algorithms over

the standard CS is tested, then a scalability study is performed. Finally, an application to parameter identification of fractional-order chaotic systems is also investigated.

4.1. Experimental Setup

For parameter settings of CS, CSML, CSP, CSC and CSW, the probability P_a is set to 0.25 [7], the scaling factor α is set to 0.01. The proposed randomness-enhanced CS algorithms introduce new parameters to CS: the scale parameter γ and the Mittag-Leffler index β in CSML; the scale parameter b and the shape parameter a in CSP; the location parameter μ and the scale parameter σ in CSC; the scale parameter $\kappa > 0$ and the shape parameter ξ in CSW. As for these newly introduced parameters, their values are given in Table 1 after analysis in Section 4.2.

Table 1. Parameters for randomness-enhanced cuckoo search (CS) algorithms.

Distribution	Algorithm	Parameters
Mittag-Leffler distribution	CSML	$\gamma = 4.5$, $\beta = 0.8$
Pareto distribution	CSP	$a = 1.5$, $b = 4.5$
Cauchy distribution	CSC	$\sigma = 4.5$, $\mu = 0.8$
Weibull distribution	CSW	$\xi = 0.3$, $\kappa = 4$

All the selected benchmark functions are minimization problems, and a more detailed description of these benchmark functions can be found in References [21,22]. The test suite consists of 20 unconstrained single-objective benchmark functions with different characteristics, including unimodal (F_{sph}, F_{ros}), multimodal (F_{ack}, F_{grw}, F_{ras}, F_{sch}, F_{sal}, F_{wht}, F_{pn1} and F_{pn2}) and rotated and/or shifted functions (F_1-F_{10}) as described in Appendix A. Moreover, the population size satisfies $NP = D$ where D denotes the dimension of the problem unless a change is mentioned.

In the experimental studies, the maximum number of function evaluations (namely Max_FEs) is taken as the termination criterion and set to $10,000 \times D$. The average of the function error value $f(X_{best}) - f(X^*)$ is used to assess the optimization performance, where X_{best} is the best solution found by the algorithms in each run and X^* is the actual global optimal solution of the test function. All the algorithms are evaluated for 50 times and the averaged experimental results are recorded for each benchmark function respectively. Besides, two non-parametric statistical tests for independent samples are taken to detect the differences between the proposed algorithm and the compared algorithms. The tests contain the Wilcoxon signed-rank test at the 5% significance level and the Friedman test. The symbol "‡", "†" and "≈" respectively denote the average performance gained by the chosen approach is weaker than, better than, and similar to the compared algorithm. Meanwhile, the best experimental results for each benchmark problem are marked in boldface, for clarity.

4.2. Parameter Tuning

From Section 3.2, it can be seen that each of the four randomness-enhanced CS algorithms brings two new user-defined parameters, for example, the scale parameter γ and the Mittag-Leffler index β in CSML. To illustrate the impact of these two parameters on the optimization results and to offer reference values to users of our algorithm, parameter analyses are conducted in advance and corresponding experiments are performed on unimodal function F_{sph} and multimodal function F_{ack} with dimension D set to 30. The optimal value of selected benchmark functions is 0. $10,000 \times D$ is the default value for Max_FEs. 15 independent runs are carried out for each parameter setting to reduce statistical sampling effects. The experimental results are plotted in Figure 1. For simplicity of description, only the result of parameter tuning for CSML is shown here, and the same operation is conducted on CSP, CSC, and CSW. In Figure 1a, γ varies within interval $[0.5, 4.5]$ in steps of 0.5, β varies from 0.1 to 0.9 in steps of 0.1, and 'Error' represents the average value of the differences between the benchmark function value and its optimal value over 15 independent runs.

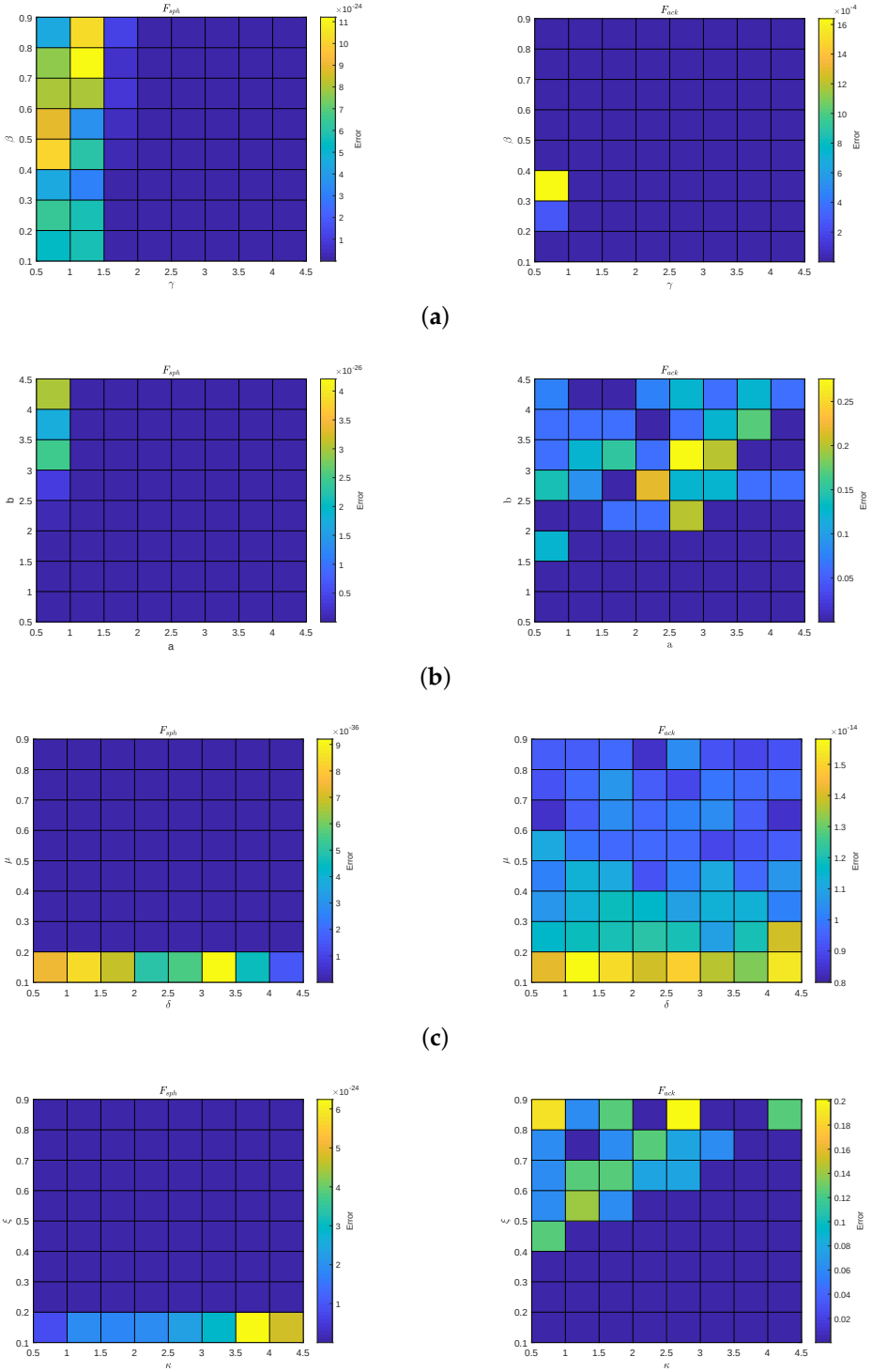

Figure 1. Impact of user-defined parameter values of CSML, CSP, CSC and CSW on the results for selected benchmark functions. (**a**) CS with the Mittag-Leffler distribution, CSML. (**b**) CS with the Pareto distribution, CSP. (**c**) CS with the Cauchy distribution, CSC. (**d**) CS with the Weibull distribution, CSW.

From Figure 1a, we can see that the Mittag-Leffler index β, in general, has a slight effect on the performance of CSML, whereas the value of scale parameter γ shows a more significant impact on the experimental results. According to the right part of each subfigure in Figure 1a, the larger the value of scale parameter γ is, the better the performance of CSML will be. In view of the above

considerations, we set the values of γ and β to 4.5 and 0.8 for all the experiments being conducted in the next subsections. For Pareto distribution, Cauchy distribution and Weibull distribution, the same parameter analysis is performed according to Figure 1b–d. The user-defined parameter values for all the randomness-enhanced CS algorithms are listed in Table 1.

4.3. Performance Evaluation of Randomness-Enhanced Cs Algorithms

In this section, lots of experiments are performed in order to probe into the effectiveness and efficiency of different heavy-tailed distributions on the performance of CS, and meanwhile, to decide the optimal randomness in improving CS. In our experiments, the standard CS and four proposed randomness-enhanced CS algorithms (namely, CSML, CSP, CSC, and CSW) are tested on 20 test functions where D is set to 30. The experimental results are presented in Table 2.

Table 2. Comparisons between CS and four randomness-enhanced CS algorithms at D = 30.

Fun	CS	CSML	CSP	CSC	CSW
F_{sph}	9.58×10^{-31}	4.90×10^{-54}‡	$\mathbf{4.74}\times10^{-59}$‡	1.17×10^{-57}‡	4.40×10^{-51}‡
F_{ros}	1.20×10^{1}	5.22×10^{0}‡	3.10×10^{0}‡	$\mathbf{2.74}\times10^{0}$‡	8.62×10^{0}‡
F_{ack}	7.70×10^{-13}	1.06×10^{-14}‡	1.07×10^{-14}‡	9.56×10^{-15}‡	$\mathbf{8.28}\times10^{-15}$‡
F_{grw}	7.11×10^{-17}	$\mathbf{0.00}\times10^{0}$‡	$\mathbf{0.00}\times10^{0}$‡	$\mathbf{0.00}\times10^{0}$‡	$\mathbf{0.00}\times10^{0}$‡
F_{ras}	2.32×10^{1}	1.38×10^{1}‡	1.88×10^{1}‡	1.49×10^{1}‡	$\mathbf{8.34}\times10^{0}$‡
F_{sch}	1.57×10^{3}	5.37×10^{2}‡	1.32×10^{3}‡	4.80×10^{2}‡	$\mathbf{3.56}\times10^{1}$‡
F_{sal}	3.76×10^{-1}	2.96×10^{-1}‡	3.00×10^{-1}‡	2.84×10^{-1}‡	$\mathbf{2.20}\times10^{-1}$‡
F_{wht}	3.73×10^{2}	2.00×10^{2}‡	2.49×10^{2}‡	2.27×10^{2}‡	$\mathbf{1.93}\times10^{2}$‡
F_{pn1}	2.07×10^{-3}	$\mathbf{1.57}\times10^{-32}$‡	$\mathbf{1.57}\times10^{-32}$‡	2.07×10^{-3}≈	$\mathbf{1.57}\times10^{-32}$‡
F_{pn2}	4.82×10^{-28}	$\mathbf{1.35}\times10^{-32}$‡	$\mathbf{1.35}\times10^{-32}$‡	$\mathbf{1.35}\times10^{-32}$‡	$\mathbf{1.35}\times10^{-32}$‡
F_1	6.48×10^{-30}	$\mathbf{0.00}\times10^{0}$‡	$\mathbf{0.00}\times10^{0}$‡	$\mathbf{0.00}\times10^{0}$‡	$\mathbf{0.00}\times10^{0}$‡
F_2	1.05×10^{-2}	1.10×10^{-3}‡	$\mathbf{2.77}\times10^{-4}$‡	1.40×10^{-3}‡	1.23×10^{-2}†
F_3	$\mathbf{2.17}\times10^{6}$	3.04×10^{6}†	2.99×10^{6}†	3.25×10^{6}†	3.61×10^{6}†
F_4	1.79×10^{3}	4.98×10^{2}‡	$\mathbf{3.58}\times10^{2}$‡	4.02×10^{2}‡	5.51×10^{2}‡
F_5	3.17×10^{3}	2.44×10^{3}‡	1.98×10^{3}‡	2.11×10^{3}‡	$\mathbf{1.94}\times10^{3}$‡
F_6	2.78×10^{1}	1.57×10^{1}‡	$\mathbf{9.91}\times10^{0}$‡	1.23×10^{1}‡	1.59×10^{1}‡
F_7	$\mathbf{1.34}\times10^{-3}$	2.22×10^{-3}†	5.79×10^{-3}†	3.73×10^{-3}†	2.49×10^{-3}†
F_8	$\mathbf{2.09}\times10^{1}$	$\mathbf{2.09}\times10^{1}$≈	$\mathbf{2.09}\times10^{1}$≈	$\mathbf{2.09}\times10^{1}$≈	$\mathbf{2.09}\times10^{1}$≈
F_9	2.84×10^{1}	1.30×10^{1}‡	2.74×10^{1}‡	1.28×10^{1}‡	$\mathbf{6.81}\times10^{0}$‡
F_{10}	1.69×10^{2}	1.21×10^{2}‡	1.31×10^{2}‡	1.18×10^{2}‡	$\mathbf{1.03}\times10^{2}$‡
‡/≈ /†	-	17/1/2	17/1/2	16/2/2	16/1/3
p-value	-	8.97×10^{-3}	1.00×10^{-2}	1.00×10^{-2}	1.87×10^{-2}
Avg. rank	4.35	2.78	2.88	2.58	**2.43**

"‡", "†" and "≈" respectively denote the performance of CS is worse than, better than, and similar to those of the proposed algorithms according to the Wilcoxon's rank test at a 0.05 significance level.

According to Table 2, it can be clearly found that CS with different heavy-tailed probability distributions provides significantly better results when compared with the original CS. Specifically speaking, in terms of the total number of "‡/≈ /†", CS is inferior to CSML, CSP, CSC, and CSW on 17, 17, 16 and 16 test functions, similar to CSML, CSP, CSC and CSW on 1, 1, 2 and 1 test functions, and superior to CSML, CSP, CSC, and CSW on 2, 2, 2 and 3 test functions, respectively. It is worth noting that CSML, CSP, CSC and CSW are capable of achieving the global optimum on test problem F_{grw} and F_1, while CS does not. Moreover, all the p-values are less than 0.05. These results suggest that CSML, CSP, CSC, and CSW are able to significantly improve the performance of CS for the test functions at D = 30. The comprehensive ranking orders are CSW, CSC, CSML, CSP, and CS in a descending manner. This indicates that the integration of different heavy-tailed probability distributions into CS not only retains the merit of CS but also performs even better. Besides, the Weibull

distribution performs the best in enhancing the search ability of CS, that is, CSW is supposed to be the optimal randomness in improving CS among all the comparison methods for solving benchmark problems at $D = 30$.

To further discuss the convergence speed of the four randomness-enhanced CS algorithms, several test problems (namely F_{sph}, F_{grw}, F_1 and F_{10}) at $D = 30$ are selected to plot the convergence curves of the averages of the function error values within Max_FEs over 50 independent runs, which are presented in Figure 2. From Figure 2, it can be observed that CSML, CSP, CSC, and CSW converge outstandingly faster than CS according to the convergence curves. In summary, it can be concluded that the standard CS algorithm can be improved by integrating different heavy-tailed probability distributions rather than Lévy distribution into it.

Besides, to analyze the reasons for different performances among the four proposed randomness-enhanced CS algorithms, the jump lengths of CS, CSML, CSP, CSC, and CSW (namely, $\alpha \otimes$ Lévy(λ), $\alpha \otimes$ MittagLeffler(β, γ), $\alpha \otimes$ Pareto(b, a), $\alpha \otimes$ Cauchy(μ, σ), and $\alpha \otimes$ Weibull(ξ, κ)) are depicted in Figure 3, where the parameters are given in Table 1 and the scaling factor is set to 0.01. From Figure 3, it can be observed that (1) Lévy distribution and Cauchy distribution are one-sided distribution where all the random numbers are positive, and the other three distributions are two-sided; (2) large steps frequently take place for all distributions; (3) since the tail of Weibull distribution is the lightest, the extreme large steps (compared with its mean) are less likely to happen.

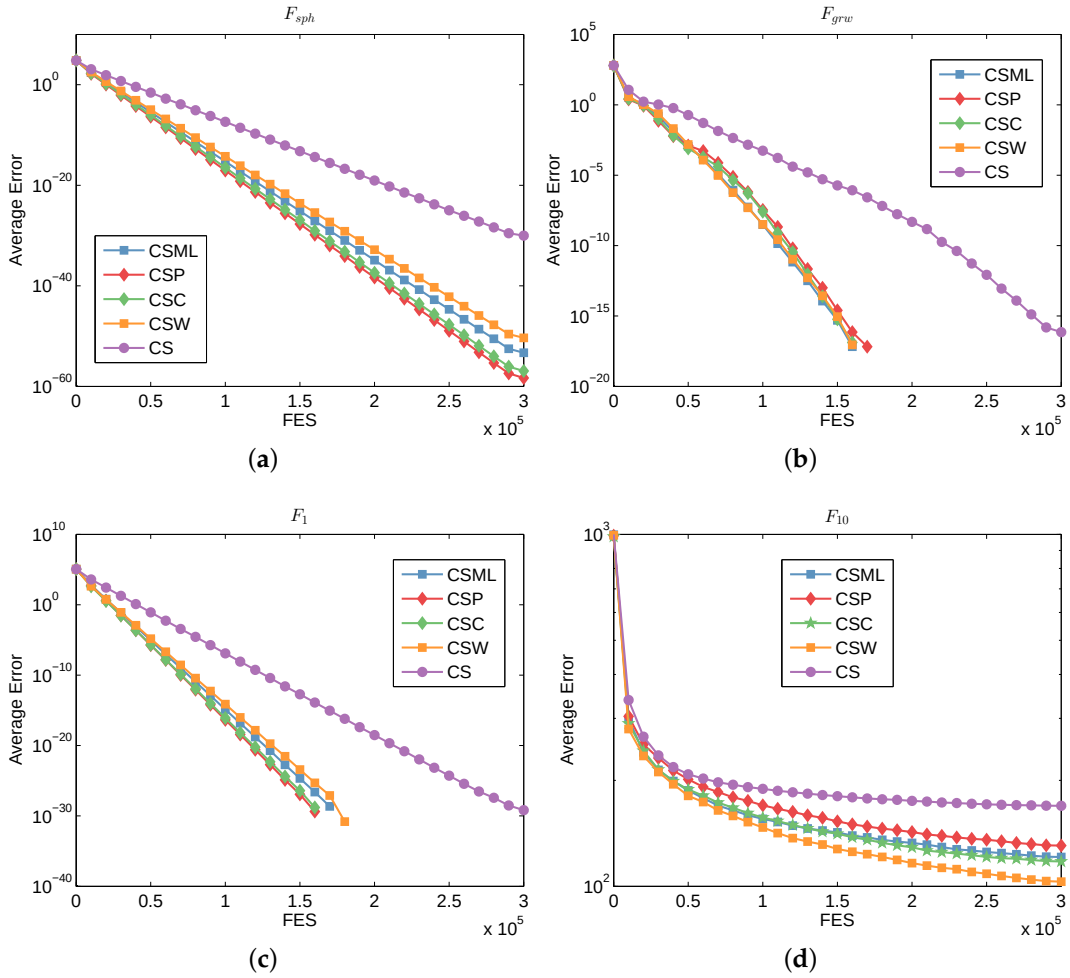

Figure 2. Convergence curves of CS and different improved CS algorithms for selected functions at $D = 30$. (**a**) Sphere's Function F_{sph}. (**b**) Griewank's Function F_{grw}. (**c**) Shifted Sphere Function F_1. (**d**) Shifted Rotated Rastrigin's Function F_{10}.

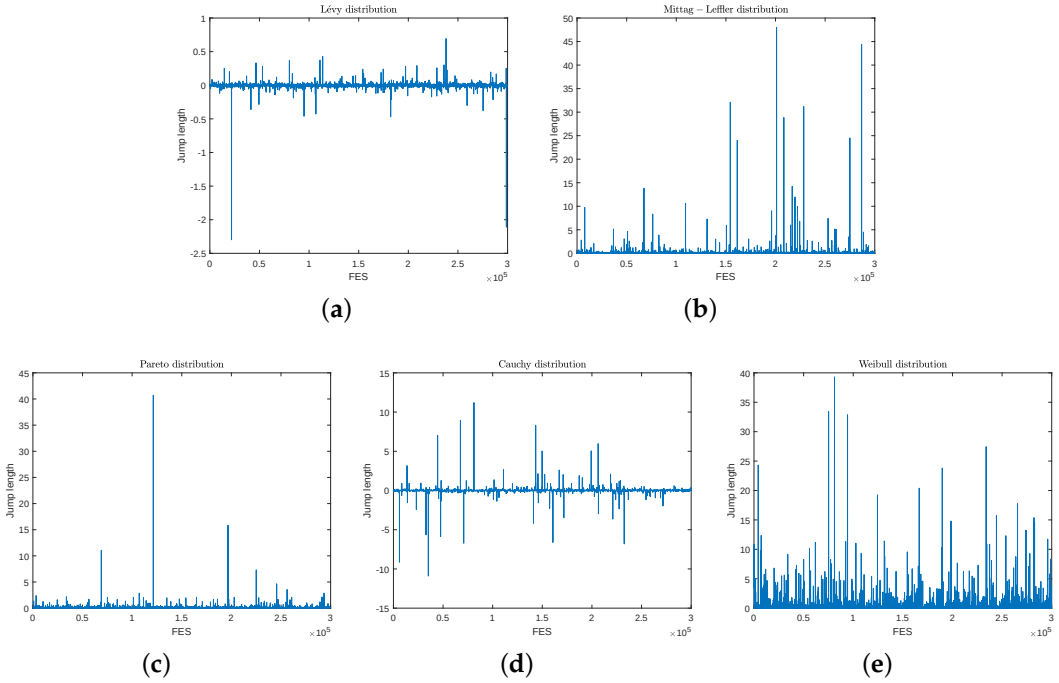

Figure 3. Jump lengths of CS, CSML, CSP, CSC and CSW. (**a**) Lévy distribution of CS. (**b**) Mittag-Leffler distribution of CSML. (**c**) Pareto distribution of CSP. (**d**) Cauchy distribution of CSC. (**e**) Weibull distribution of CSW.

4.4. Scalability Study

In this section, a scalability study comparing with the standard CS algorithm is conducted in order to study the effect of problem size on the performance of the four proposed randomness-enhanced CS algorithms. We carry out experiments on the 20 benchmark functions with dimension D set to 10 and 50. When $D = 10$, the population size is chosen as $NP = 30$; meanwhile, when $D = 50$, the population size is selected as $NP = D$. All the other control parameters are kept unchanged. The experimental results achieved by CS and four proposed randomness-enhanced CS algorithms at $D = 10$ and $D = 50$ are listed in Tables 3 and 4, respectively, and the results of the Wilcoxon signed-rank test are also given in the tables.

According to Table 3, CSML, CSP, CSC, and CSW are significantly better than CS on 7, 17, 18 and 19 test functions, similar to CS on 0, 1, 1 and 1 test functions, and worse than CS on 13, 2, 1 and 0 test functions, respectively. The comprehensive ranking orders in the case of $D = 10$ are CSW, CSC, CSP, CS, and CSML in descending manner. The results show that the performance improvement of using different heavy-tailed probability distributions persists expect CSML when the problem dimension reduces to 10. In the case of $D = 50$, it can be observed from Table 4 that CSML, CSP, CSC and CSW perform better than CS on 16, 14, 16 and 16 test functions, to CS on 1, 1, 1 and 1 test functions, and worse than CS on 3, 5, 3 and 3 test functions, respectively. Meanwhile, the corresponding comprehensive ranking orders when $D = 50$ are CSC, CSML, CSP, CSW and CS. In general, we can draw conclusions that the advantages of four randomness-enhanced CS algorithms over the standard CS are overall stable when the problem dimension increases, except CSML which deteriorates to a certain extent when D set to 10. Furthermore, regarding to the different comprehensive ranking orders obtained at every dimension, it is pointed out that CS with Lévy flights seems not the optimal randomness when compared with those using different heavy-tailed probability distributions in CS.

Table 3. Comparisons between CS and four randomness-enhanced CS algorithms at $D = 10$.

Fun	CS	CSML	CSP	CSC	CSW
F_{sph}	4.87×10^{-26}	3.39×10^{-31}‡	$\mathbf{4.21}\times10^{-48}$‡	2.04×10^{-46}‡	2.48×10^{-46}‡
F_{ros}	9.63×10^{-1}	3.02×10^{1}†	$\mathbf{1.12}\times10^{-1}$‡	1.75×10^{-1}‡	2.97×10^{-1}‡
F_{ack}	4.16×10^{-11}	2.50×10^{-14}‡	$\mathbf{4.37}\times10^{-15}$‡	4.44×10^{-15}‡	4.44×10^{-15}‡
F_{grw}	3.44×10^{-2}	$\mathbf{0.00}\times10^{0}$‡	2.22×10^{-2}‡	2.07×10^{-2}‡	1.44×10^{-2}‡
F_{ras}	3.00×10^{0}	6.93×10^{1}†	2.25×10^{0}‡	2.95×10^{-1}‡	$\mathbf{2.26}\times10^{-9}$‡
F_{sch}	6.72×10^{1}	3.80×10^{3}†	1.38×10^{1}‡	6.91×10^{-3}‡	$\mathbf{1.27}\times10^{-4}$‡
F_{sal}	1.04×10^{-1}	4.78×10^{-1}†	$\mathbf{9.99}\times10^{-2}$‡	$\mathbf{9.99}\times10^{-2}$‡	$\mathbf{9.99}\times10^{-2}$‡
F_{wht}	2.40×10^{1}	9.89×10^{2}†	1.54×10^{1}‡	1.01×10^{1}‡	$\mathbf{5.72}\times10^{0}$‡
F_{pn1}	1.96×10^{-16}	2.01×10^{-28}‡	$\mathbf{4.71}\times10^{-32}$‡	$\mathbf{4.71}\times10^{-32}$‡	$\mathbf{4.71}\times10^{-32}$‡
F_{pn2}	4.86×10^{-23}	9.17×10^{-30}‡	$\mathbf{1.35}\times10^{-32}$‡	$\mathbf{1.35}\times10^{-32}$‡	$\mathbf{1.35}\times10^{-32}$‡
F_1	4.13×10^{-26}	$\mathbf{0.00}\times10^{0}$‡	$\mathbf{0.00}\times10^{0}$‡	$\mathbf{0.00}\times10^{0}$‡	$\mathbf{0.00}\times10^{0}$‡
F_2	8.16×10^{-14}	3.73×10^{2}†	$\mathbf{1.33}\times10^{-21}$‡	4.54×10^{-19}‡	1.51×10^{-16}‡
F_3	$\mathbf{2.08}\times10^{2}$	1.70×10^{7}†	7.20×10^{2}†	6.78×10^{2}†	8.31×10^{2}†
F_4	1.01×10^{-5}	1.96×10^{4}†	$\mathbf{1.46}\times10^{-9}$‡	1.20×10^{-8}‡	4.82×10^{-8}‡
F_5	9.30×10^{-5}	6.82×10^{3}†	$\mathbf{6.13}\times10^{-10}$‡	5.11×10^{-9}‡	9.27×10^{-9}‡
F_6	9.78×10^{-1}	4.11×10^{1}†	6.38×10^{-1}‡	3.48×10^{-1}‡	$\mathbf{2.69}\times10^{-1}$‡
F_7	5.33×10^{-2}	$\mathbf{1.07}\times10^{-3}$‡	5.91×10^{-2}†	4.72×10^{-2}‡	4.39×10^{-2}‡
F_8	2.04×10^{1}	2.11×10^{1}†	2.04×10^{1}≈	2.04×10^{1}≈	$\mathbf{2.03}\times10^{1}$‡
F_9	2.75×10^{0}	7.37×10^{1}†	1.80×10^{0}‡	1.79×10^{-1}‡	$\mathbf{2.35}\times10^{-10}$‡
F_{10}	1.99×10^{1}	2.89×10^{2}†	1.63×10^{1}‡	1.59×10^{1}‡	$\mathbf{1.43}\times10^{1}$‡
‡/≈ /†	-	7/0/13	17/1/2	18/1/1	19/1/0
Avg. rank	4.10	4.28	2.28	2.25	**2.10**

"‡", "†" and "≈" respectively denote the performance of CS is worse than, better than, and similar to those of the proposed algorithms according to the Wilcoxon's rank test at a 0.05 significance level.

Table 4. Comparison results between CS and four randomness-enhanced CS algorithms at $D = 50$.

Fun	CS	CSML	CSP	CSC	CSW
F_{sph}	3.79×10^{-17}	3.47×10^{-31}‡	$\mathbf{7.41}\times10^{-36}$‡	5.75×10^{-32}‡	2.73×10^{-24}‡
F_{ros}	4.22×10^{1}	3.07×10^{1}‡	$\mathbf{2.82}\times10^{1}$‡	2.99×10^{1}‡	3.41×10^{1}‡
F_{ack}	2.85×10^{-2}	2.43×10^{-14}‡	$\mathbf{2.05}\times10^{-14}$‡	$\mathbf{2.05}\times10^{-14}$‡	7.40×10^{-13}‡
F_{grw}	1.93×10^{-10}	$\mathbf{0.00}\times10^{0}$‡	$\mathbf{0.00}\times10^{0}$‡	$\mathbf{0.00}\times10^{0}$‡	$\mathbf{0.00}\times10^{0}$‡
F_{ras}	8.44×10^{1}	$\mathbf{6.80}\times10^{1}$‡	8.69×10^{1}†	7.54×10^{1}‡	7.37×10^{1}‡
F_{sch}	4.87×10^{3}	4.14×10^{3}‡	6.05×10^{3}†	4.38×10^{3}‡	$\mathbf{2.38}\times10^{3}$‡
F_{sal}	6.69×10^{-1}	4.68×10^{-1}‡	4.87×10^{-1}‡	4.22×10^{-1}‡	$\mathbf{3.68}\times10^{-1}$‡
F_{wht}	1.36×10^{3}	$\mathbf{9.58}\times10^{2}$‡	1.21×10^{3}‡	1.09×10^{3}‡	1.13×10^{3}‡
F_{pn1}	8.13×10^{-3}	6.74×10^{-28}‡	1.04×10^{-27}‡	$\mathbf{7.21}\times10^{-30}$‡	1.47×10^{-23}‡
F_{pn2}	3.25×10^{-14}	1.02×10^{-29}‡	$\mathbf{1.57}\times10^{-32}$‡	1.44×10^{-30}‡	2.01×10^{-23}‡
F_1	1.40×10^{-16}	$\mathbf{0.00}\times10^{0}$‡	$\mathbf{0.00}\times10^{0}$‡	$\mathbf{0.00}\times10^{0}$‡	3.57×10^{-24}‡
F_2	2.34×10^{2}	3.57×10^{2}†	$\mathbf{1.86}\times10^{2}$‡	4.49×10^{2}†	8.59×10^{2}†
F_3	$\mathbf{8.53}\times10^{6}$	1.66×10^{7}†	1.47×10^{7}†	1.83×10^{7}†	1.85×10^{7}†
F_4	2.72×10^{4}	1.99×10^{4}‡	$\mathbf{1.72}\times10^{4}$‡	1.91×10^{4}‡	1.88×10^{4}‡
F_5	1.06×10^{4}	6.95×10^{3}‡	6.49×10^{3}‡	6.65×10^{3}‡	$\mathbf{6.30}\times10^{+03}$‡
F_6	6.38×10^{1}	3.90×10^{1}‡	4.15×10^{1}‡	$\mathbf{3.63}\times10^{1}$‡	4.43×10^{1}‡
F_7	$\mathbf{1.30}\times10^{-3}$	1.81×10^{-3}†	3.56×10^{-3}†	2.43×10^{-3}†	3.64×10^{-3}†
F_8	$\mathbf{2.11}\times10^{1}$	$\mathbf{2.11}\times10^{1}$≈	$\mathbf{2.11}\times10^{1}$≈	$\mathbf{2.11}\times10^{1}$≈	$\mathbf{2.11}\times10^{1}$≈
F_9	1.24×10^{2}	7.04×10^{1}‡	1.27×10^{2}†	7.47×10^{1}‡	$\mathbf{6.50}\times10^{1}$‡
F_{10}	3.87×10^{2}	2.87×10^{2}‡	3.13×10^{2}‡	2.85×10^{2}‡	$\mathbf{2.69}\times10^{2}$‡
‡/≈ /†	-	16/1/3	14/1/5	16/1/3	16/1/3
Avg. rank	4.10	2.58	2.70	**2.55**	3.08

"‡", "†" and "≈" respectively denote the performance of CS is worse than, better than, and similar to those of the proposed algorithms according to the Wilcoxon's rank test at a 0.05 significance level.

4.5. Application to Parameter Identification of Fractional-Order Chaotic Systems

In this section, the four proposed randomness-enhanced CS algorithms (namely, CSML, CSP, CSC and CSW) are applied to identify unknown parameters of fractional-order chaotic systems, which is a critical issue in chaos control and synchronization. Our main task of this section is to further demonstrate that improving CS with different heavy-tailed probability distributions can also effectively tackle the real-world complex optimization problems besides the benchmark problems. In fact, by using a non-Lyapunov way according to problem formulation suggested in Reference [23], the nonlinear function optimization can be converted to from parameter identification of uncertain fractional-order chaotic systems.

In the numerical simulation, the fractional-order financial system [24] under the Caputo definition is taken for example, which can be described as

$$\begin{cases} {}_0D_t^{q_1}x(t) = z(t) + x(t)(y(t) - a), \\ {}_0D_t^{q_2}y(t) = 1 - by(t) - x^2(t), \\ {}_0D_t^{q_3}z(t) = -x(t) - cz(t), \end{cases} \tag{16}$$

where q_1, q_2, q_3 and a, b, c are fractional orders and systematic parameters. When $(q_1, q_2, q_3) = (1, 0.95, 0.99)$, $(a, b, c) = (1, 0.1, 1)$ and initial point $(x_0, y_0, z_0) = (2, -1, 1)$, the system above is chaotic.

Suppose the structure of system (16) is known and the systematic parameters a, b, c are unknown, then the objective function is defined as

$$F = \sum_{k=1}^{N} \| X_k - \hat{X}_k \|^2, \tag{17}$$

where X_k and $\hat{X}_k$ denote the state vector of system (16) and its estimated system at time k, respectively. $k = 1, 2, \ldots, M$ is the sampling time point and M denotes the length of data used for parameter estimation. $\|\cdot\|$ denotes Euclid norm. Parameter identification can be achieved by searching suitable (a^*, b^*, c^*) such that the objective function (17) is minimized, that is,

$$(a^*, b^*, c^*) = \arg \min_{(\hat{a}, \hat{b}, \hat{c}) \in \blacksquare} F, \tag{18}$$

where $\blacksquare$ is the searching space admitted for systematic parameters. Figure 4 depicts the distribution figure of system (16) for the objective function values.

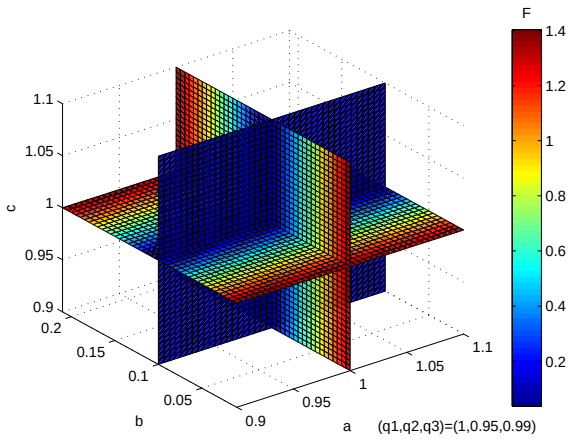

Figure 4. Distribution of the objective function values for system (16).

The validation of the proposed methods in this paper is further proved by comparing CSML, CSP, CSC and CSW with the standard CS algorithm for parameter identification. In the simulations, the maximum iteration number is set to 200 and the population size is set to 40. For the system to be identified, the step size is set to 0.005 and the number of samples set to 200. In addition, it is worth mentioning that the same computation effort is used in implementation for all the compared algorithms to make a fair comparison. Table 5 lists the statistical results of the average identified values, the corresponding relative error values and the objective function values for system (16). From Table 5, it can be clearly observed that all the four proposed randomness-enhanced CS algorithms outperform CS according to the average objective function values and they are able to generate estimated values with much higher accuracy than CS. Besides, it can be seen that CSP surpasses CS, CSML, CSW and CSC in obtaining the best average identified values, the corresponding relative error values and the objective function values.

Table 5. Statistical results of different methods for system (16), in terms of the average estimated values, the relative error values and objective function values.

Method	**CS**	**CSML**	**CSP**	**CSC**	**CSW**
a	0.999999825481796	0.999999979386471	**1.000000001165006**	0.999999930875086	0.999999994619958
$\frac{\lvert a-1.00\rvert}{1.00}$	1.75×10^{-7}	2.28×10^{-8}	$\mathbf{1.17}\times10^{-9}$	6.91×10^{-8}	5.38×10^{-9}
b	0.100000078306700	0.100000006492360	**0.099999999732393**	0.100000038684769	0.100000001325757
$\frac{\lvert b-0.10\rvert}{0.10}$	7.83×10^{-7}	1.12×10^{-7}	$\mathbf{2.68}\times10^{-9}$	3.87×10^{-7}	2.06×10^{-8}
c	1.000000126069434	0.999999979588057	**0.999999995606294**	0.999999876500337	0.999999979353103
$\frac{\lvert c-1.00\rvert}{1.00}$	1.26×10^{-7}	4.61×10^{-8}	$\mathbf{4.39}\times10^{-9}$	1.23×10^{-7}	1.33×10^{-8}
$F_{Avg\pm Std}$	1.07×10^{-5}	4.75×10^{-7}	$\mathbf{7.46}\times10^{-8}$	1.89×10^{-6}	1.03×10^{-7}
F_{Std}	5.46×10^{-6}	2.74×10^{-7}	$\mathbf{3.29}\times10^{-8}$	9.38×10^{-7}	6.12×10^{-8}

a, b and c are parameters to be identified. Their actual values are 1.00, 0.10 and 1.00, respectively.

Moreover, Figure 5 shows the convergence curves of the relative error values of the estimated parameters and objective function values for the corresponding system via CSML, CSP, CSC, CSW and CS. From Figure 5a–c, the relative error values of the estimated values generated by the randomness-enhanced CS algorithms converge to zero more quickly than the original CS. This indicates that CS algorithms with the four different heavy-tailed probability distributions are able to obtain more accurate values of the estimated parameters. In terms of Figure 5d, the objective function values of CSML, CSP, CSC, CSW also decline faster than CS and among which CSP performs the best. It is noteworthy that CSW has a similar convergence curve of objective function values with CSP and can converge to the nearby area of CSP. Therefore, CSW can still be considered as an efficient tool for solving optimization problems.

According to the foregoing discussion, it can be summarized that the randomness-enhanced CS algorithms are able to exactly identify the unknown specific parameters of the fractional-order system (16) with better effectiveness and robustness and CSP together with CSW may be treated as a useful tool for handling the problem of parameter identification.

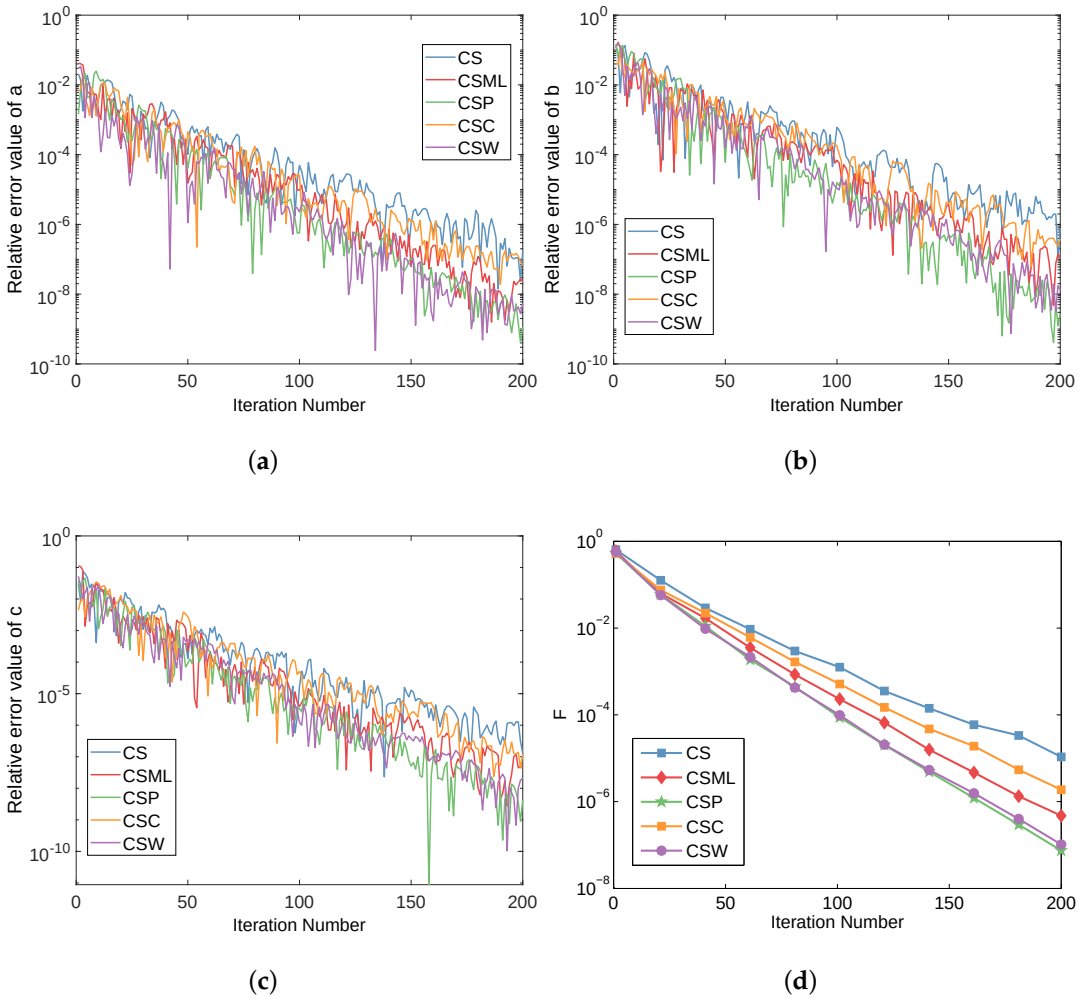

Figure 5. The convergence curves of the relative error values and objective function values for system (16). (**a**) The relative error value of a. (**b**) The relative error value of b. (**c**) The relative error value of c. (**d**) Objective function value of F.

5. Conclusions

The purpose of this paper is to study the optimal randomness in swarm-based search algorithms. In the study, CS is taken as a representative method of swarm-based optimization algorithms and the results can be generalized to other swarm-based search algorithms. The impact of different heavy-tailed distributions on the performance of CS is investigated. By replacing Lévy flights with steps generated from other heavy-tailed distributions in CS, four different randomness-enhanced CS algorithms (namely CSML, CSP, CSC and CSW) are presented by applying Mittag-Leffler distribution, Pareto distribution, Cauchy distribution and Weibull distribution, in order to improve the optimization performance of CS. The improvement in effectiveness and efficiency is validated through dedicated experiments. The experimental results indicate that all four proposed randomness-enhanced CS algorithms show a significant improvement in effectiveness and efficiency over the standard CS algorithm. Furthermore, the randomness-enhanced CS algorithms are successfully applied to system identification. In summary, CS with different heavy-tailed probability distributions can be regarded as an efficient and promising tool for solving the real-world complex optimization problems besides the benchmark problems.

Future promising topics can be directed to (1) theoretically analyze the global convergence of randomness-enhanced CS algorithms; (2) do a similar analyses to other swarm-based search

algorithms for the optimal randomness; (3) since the search range is always finite for swarm-based search algorithms, it is necessary to study the optimal randomness in a finite range.

Author Contributions: Methodology, J.W. and Y.C. (Yuquan Chen); Conceptualization, Y.C. (YangQuan Chen); Implementation of numerical schemes and Writing of the manuscript was completed by J.W.; Review & Editing by Y.C. (YangQuan Chen); Supervision, Y.C. (YangQuan Chen) and Y.Y. All authors read and approved the final manuscript.

Funding: This work is supported by the Fundamental Research Funds for the Central Universities (No. 2017YJS200), China Scholarship Council (No. 201807090092) and the National Nature Science Foundation of China (No. 61772063).

Acknowledgments: The authors are thankful to the anonymous referees for their invaluable suggestions.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

The description of the 20 benchmark functions is given as follows. The formulae of the corresponding benchmark functions are presented in Table A1.

1. F_{sph}: Sphere's Function.
2. F_{ros}: Rosenbrock's Function.
3. F_{ack}: Ackley's Function.
4. F_{grw}: Griewank's Function.
5. F_{ras}: Rastrigin's Function.
6. F_{sch}: Generalized Schwefel's Problem 2.26.
7. F_{sal}: Salomon's Function.
8. F_{wht}: Whitely's Function.
9. F_{pn1}: Generalized Penalized Function 1.
10. F_{pn2}: Generalized Penalized Function 2.
11. F_1: Shifted Sphere Function.
12. F_2: Shifted Schwefel's Problem 1.2.
13. F_3: Shifted Rotated High Conditioned Elliptic Function.
14. F_4: Shifted Schwefel's Problem 1.2 with Noise in Fitness.
15. F_5: Schwefel's Problem 2.6 with global Optimum on Bounds.
16. F_6: Shifted Rosenbrock's Function.
17. F_7: Shifted Rotated Griewank's Function without Bounds.
18. F_8: Shifted Rotated Ackley's Function with Global Optimum on Bounds.
19. F_9: Shifted Rastrigin's Function.
20. F_{10}: Shifted Rotated Rastrigin's Function.

Table A1. Description of the benchmark functions at dimension D used in experiments.

No.	Formula	Range	Optimum
1	$F_{sph}(x) = \sum_{i=1}^{D} x_i^2$	$[-100, 100]^D$	0
2	$F_{ros}(x) = \sum_{i=1}^{D-1} \left(100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2\right)$	$[-100, 100]^D$	0
3	$F_{ack}(x) = 20 + \exp(1) - 20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)\right)$	$[-32, 32]^D$	0
4	$F_{grw}(x) = \sum_{i=1}^{D} \frac{x_i^2}{4000} - \prod_{i=1}^{D}\cos\frac{x_i}{\sqrt{i}} + 1$	$[-600, 600]^D$	0
5	$F_{ras}(x) = 10D + \sum_{i=1}^{D}\left(x_i^2 - 10\cos(2\pi x_i)\right)$	$[-5, 5]^D$	0
6	$F_{sch}(x) = 418.9829D - \sum_{i=1}^{D}\left(x_i \sin\left(\sqrt{\lvert x_i\rvert}\right)\right)$	$[-500, 500]^D$	0
7	$F_{sal}(x) = -\cos\left(2\pi\sqrt{\sum_{i=1}^{D} x_i^2}\right) + 0.1\sqrt{\sum_{i=1}^{D} x_i^2} + 1$	$[-100, 100]^D$	0
8	$F_{wht}(x) = \sum_{j=1}^{D}\sum_{i=1}^{D}\left(\frac{y_{i,j}^2}{4000} - \cos(y_{i,j}) + 1\right)$, where $y_{i,j} = 100(x_j - x_i^2)^2 + (1 - x_i)^2$	$[-100, 100]^D$	0
9	$F_{pn1}(x) = \frac{\pi}{D}\left\{10\sin^2(\pi y_1) + \sum_{i=1}^{D-1}(y_i - 1)^2\left[1 + 10\sin^2(\pi y_{i+1})\right] + (y_D - 1)^2\right\} + \sum_{i=1}^{D} u(x_i, 10, 100, 4)$, where $y_i = 1 + \frac{1}{4}(x_i + 1)$ and $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m, & x_i > a, \\ 0, & -a \le x_i \le a, \\ k(-x_i - a)^m, & x_i < -a. \end{cases}$	$[-50, 50]^D$	0
10	$F_{pn2}(x) = 0.1\left\{\sin^2(3\pi x_1) + \sum_{i=1}^{D-1}(x_i - 1)^2\left[1 + \sin^2(3\pi x_{i+1})\right] + (x_D - 1)^2\left[1 + \sin^2(2\pi x_D)\right]\right\} + \sum_{i=1}^{D} u(x_i, 5, 100, 4)$	$[-50, 50]^D$	0
11	$F_1(x) = \sum_{i=1}^{D} z_i^2 + f_{bias_1}$, where $z = x - o$, o: the shifted global optimum, $f_{bias_1} = -450$	$[-100, 100]^D$	0
12	$F_2(x) = \sum_{i=1}^{D}\left(\sum_{j=1}^{i} z_j\right)^2 + f_{bias_2}$, where $z = x - o$, o: the shifted global optimum, $f_{bias_2} = -450$	$[-100, 100]^D$	0
13	$F_3(x) = \sum_{i=1}^{D}(10^6)^{\frac{i-1}{D-1}} z_i^2 + f_{bias_3}$, where $z = (x - o) * M$, o: the shifted global optimum, M: orthogonal matrix, $f_{bias_3} = -450$	$[-100, 100]^D$	0
14	$F_4(x) = \left(\sum_{i=1}^{D}\left(\sum_{j=1}^{i} z_j\right)^2\right)(1 + 0.4 \mid N(0,1) \mid) + f_{bias_4}$, where $z = x - o$, o: the shifted global optimum, $f_{bias_4} = -450$	$[-100, 100]^D$	0
15	$F_5(x) = \max\{\mid A_i x - B_i \mid\} + f_{bias_5}$, where A is a $D * D$ matrix, $a_{i,j}$ are integer random numbers in the range $[-500, 500]$, $\det(A) \neq 0$, A_i is the ith row of A, $i = 1, \ldots, D$, $B_i = A_i * o$ is a $D * D$ matrix, o_i are random number in the range $[-100, 100]$ and $f_{bias_5} = -310$	$[-100, 100]^D$	0
16	$F_6(x) = \sum_{i=1}^{D-1}\left(100\left(z_i^2 - z_{i+1}\right)^2 + \left(z_i - 1\right)^2\right) + f_{bias_6}$, where $z = x - o + 1$, o: the shifted global optimum, $f_{bias_6} = 390$	$[-100, 100]^D$	0
17	$F_7(x) = \sum_{i=1}^{D}\frac{z_i^2}{4000} - \prod_{i=1}^{D}\cos\left(\frac{z_i}{\sqrt{i}}\right) + 1 + f_{bias_7}$, where $z = (x - o) * M$, o: the shifted global optimum, $M = M'(1 + 0.3 \mid N(0,1) \mid)$, M: linear transformation matrix, condition number=3, $f_{bias_7} = -180$ // initialize population in $[0, 600]^D$ and no bounds for x	-	0
18	$F_8(x) = -20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} z_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos\left(2\pi z_i\right)\right) + 20 + \exp(1) + f_{bias_8}$, where $z = (x - o) * M$, o: the shifted global optimum, M: linear transformation matrix, condition number=100, $f_{bias_8} = -140$	$[-32, 32]^D$	0
19	$F_9(x) = \sum_{i=1}^{D}\left(z_i^2 - 10\cos\left(2\pi z_i\right) + 10\right) + f_{bias_9}$, where $z = x - o$, o: the shifted global optimum, $f_{bias_9} = -330$	$[-5, 5]^D$	0
20	$F_{10}(x) = \sum_{i=1}^{D}\left(z_i^2 - 10\cos\left(2\pi z_i\right) + 10\right) + f_{bias_{10}}$, where $z = (x - o) * M$, o: the shifted global optimum, M: linear transformation matrix, condition number=2, $f_{bias_{10}} = -330$,	$[-5, 5]^D$	0

References

1. Yang, X.S. *Nature-Inspired Metaheuristic Algorithms*; Luniver Press: Beckington, UK, 2010.
2. Anandakumar, H.; Umamaheswari, K. A bio-inspired swarm intelligence technique for social aware cognitive radio handovers. *Comput. Electr. Eng.* **2018**, *71*, 925–937. [CrossRef]
3. Brezočnik, L.; Fister, I.; Podgorelec, V. Swarm intelligence algorithms for feature selection: A review. *Appl. Sci.* **2018**, *8*, 1521. [CrossRef]
4. Zhao, X.; Wang, C.; Su, J.; Wang, J. Research and application based on the swarm intelligence algorithm and artificial intelligence for wind farm decision system. *Renew. Energy* **2019**, *134*, 681–697. [CrossRef]
5. Dulebenets, M.A. A novel memetic algorithm with a deterministic parameter control for efficient berth scheduling at marine container terminals. *Marit. Bus. Rev.* **2017**, *2*, 302–330. [CrossRef]
6. Karaboga, D.; Basturk, B. A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm. *J. Glob. Optim.* **2007**, *39*, 459–471. [CrossRef]
7. Yang, X.S.; Deb, S. Cuckoo search via Lévy flights. In Proceedings of the Nature & Biologically Inspired Computing, Coimbatore, India, 9–11 December 2009; IEEE: Piscataway, NY, USA, 2009; pp. 210–214.
8. Yang, X.S. Firefly algorithms for multimodal optimization. In Proceedings of the International Symposium on Stochastic Algorithms, Sapporo, Japan, 26–28 October 2009; Springer: Berlin/Heidelberg, Germany, 2009; pp. 169–178.
9. Kennedy, J.; Eberhart, R. Particle swarm optimization (PSO). In Proceedings of the IEEE International Conference on Neural Networks, Perth, Australia, 27 November–1 December 1995; pp. 1942–1948.
10. Zheng, H.; Zhou, Y. A novel cuckoo search optimization algorithm based on Gauss distribution. *J. Comput. Inf. Syst.* **2012**, *8*, 4193–4200.
11. Wang, L.; Zhong, Y.; Yin, Y. Nearest neighbour cuckoo search algorithm with probabilistic mutation. *Appl. Soft Comput.* **2016**, *49*, 498–509. [CrossRef]
12. Rakhshani, H.; Rahati, A. Snap-drift cuckoo search: A novel cuckoo search optimization algorithm. *Appl. Soft Comput.* **2017**, *52*, 771–794. [CrossRef]
13. Cui, Z.; Sun, B.; Wang, G.; Xue, Y.; Chen, J. A novel oriented cuckoo search algorithm to improve DV-Hop performance for cyber–physical systems. *J. Parallel Distrib. Comput.* **2017**, *103*, 42–52. [CrossRef]
14. Salgotra, R.; Singh, U.; Saha, S. New cuckoo search algorithms with enhanced exploration and exploitation properties. *Expert Syst. Appl.* **2018**, *95*, 384–420. [CrossRef]
15. Richer, T.J.; Blackwell, T.M. The Lévy particle swarm. In Proceedings of the 2006 IEEE International Conference on Evolutionary Computation, Vancouver, BC, Canada, 16–21 July 2006; IEEE: Piscataway, NY, USA, 2006; pp. 808–815.
16. Pavlyukevich, I. Lévy flights, non-local search and simulated annealing. *J. Comput. Phys.* **2007**, *226*, 1830–1844. [CrossRef]
17. Yang, X.S. *Nature-Inspired Optimization Algorithms*; Elsevier: Amsterdam, The Netherlands, 2014.
18. Yang, X.S.; Deb, S. Engineering optimisation by cuckoo search. *Int. J. Math. Model. Numer. Optim.* **2010**, *1*, 330–343. [CrossRef]
19. Foss, S.; Korshunov, D.; Zachary, S. *An Introduction to Heavy-Tailed and Subexponential Distributions*; Springer: Berlin/Heidelberg, Germany, 2011; Volume 6.
20. Kozubowski, T.J.; Rachev, S.T. Univariate geometric stable laws. *J. Comput. Anal. Appl.* **1999**, *1*, 177–217.
21. Noman, N.; Iba, H. Accelerating differential evolution using an adaptive local search. *IEEE Trans. Evol. Comput.* **2008**, *12*, 107–125. [CrossRef]
22. Suganthan, P.N.; Hansen, N.; Liang, J.J.; Deb, K.; Chen, Y.P.; Auger, A.; Tiwari, S. Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization. *KanGAL Rep.* **2005**, *2005005*, 2005.

23. Gao, F.; Fei, F.X.; Lee, X.J.; Tong, H.Q.; Deng, Y.F.; Zhao, H.L. Inversion mechanism with functional extrema model for identification incommensurate and hyper fractional chaos via differential evolution. *Expert Syst. Appl.* **2014**, *41*, 1915–1927. [CrossRef]
24. Chen, W.C. Nonlinear dynamics and chaos in a fractional-order financial system. *Chaos Solitons Fractals* **2008**, *36*, 1305–1314. [CrossRef]