The improved mayfly optimization algorithm with Chebyshev map

Juan ZHAO¹,a, Zheng-Ming GAO²,b
¹School of electronics and information engineering, Jingchu University of technology, Jingmen, China
²School of computer engineering, Jingchu University of technology, Jingmen, China
¹juan323@jcut.edu.cn, b gaozming@jcut.edu.cn

Abstract—The mayfly optimization (MO) algorithm was just proposed recently, simulation experiments proved that it was capable to optimize both the benchmark functions and the real problems we met. In this paper, the MO algorithm would be improved with Chebyshev map, simulation experiments were carried out and results showed that the improved algorithm would indeed increase the capability.

1. INTRODUCTION
The chaotic maps were usually introduced to improve the nature-inspired algorithms, especially the swarm-based algorithms[1]. Almost all of the swarm-based algorithms have been improved by chaos. For the grey wolf optimization[2, 3], the chaotic version would greatly increase the steadiness and convergence rate[4], similar performance were verified to the whale optimization algorithm[5], firefly optimization algorithm[6], the bat algorithm[7], the equilibrium optimization algorithm[8]. Unlike the pseudo binary numbers which were involved in algorithm as randomness, the chaotic numbers would be more non-relevant to each other and consequently, the randomness caused by chaotic mapping would be even better for choice of individuals, or selection. Therefore, almost all of the swarm-based optimization algorithms have their improved version of chaos.

The mayfly optimization (MO) algorithm was just proposed and published in this year[9]. Simulation experiments on various benchmark functions and the real engineering problems all proved that it would be capable in optimization. Considering the popular improvements with chaos, we here proposed the improved MO algorithm with Chebyshev map, which would be a good choice to replace the random numbers in uniform distribution involved in the original MO algorithm.

The rest of this paper would be arranged as follows: In section 2, we would describe the MO algorithm and proposed the improved version in Section 3, simulation experiments would be carried on in Section 4. Discussions and conclusions would be drawn in Section 5.

2. THE MO ALGORITHM
In this section, we would have a brief introduction on the MO algorithm.

The mayflies were living in swarms in water for almost several years, while when they evolve to be insects with wings, most of them would survive in one day to seven days. And consequently, the mayflies with wings would be busy all in lives to find their husbands or wives to mate and reproduce themselves. The MO algorithm was inspired by their searching for mates.
2.1. Movements of male mayflies
The male mayflies were all strong enough and they would update the positions with their own speed:

\[ x_i(t + 1) = x_i(t) + v_i(t + 1) \]  

(1)

Where, \( x_i(t) \) and \( x_i(t + 1) \) represent the position of \( i \)-th individual in swarms at the current and next iterations. \( v_i(t + 1) \) represents its speed.

The male mayflies would fly towards to the female mayflies according to their own experience. Therefore, they would search the female mayflies based on the situation that the result or fitness value \( f(x_i) \) for \( i \)-th individual is whether better than its historical best position \( x_i(t) \) in \( t \) iteration or not. If \( f(x_i) > f(x_{hi}) \), then they would update their velocity in the following way:

\[ v_i(t + 1) = g \cdot v_i(t) + a_1 e^{-\beta r^2} [x_{hi} - x_i(t)] + a_2 e^{-\beta r^2} [x_g - x_i(t)] \]  

(2)

Where, \( g \) represents the weights of the current velocity. \( a_1, a_2, \) and \( \beta \) are constants. \( x_{hi} \) and \( x_g \) represent the historical best position for the \( i \)-th individual and the global best position among the swarms. \( r_p \) and \( r_g \) represent the Cartesian distances between the current position and the historical best position, the global best position respectively. Cartesian distance would be the second norm for an array:

\[ \left| x_i - x_j \right| = \sqrt{\sum_{k=1}^{n} (x_{ik} - x_{jk})^2} \]  

(3)

On the contrary, if \( f(x_i) < f(x_{hi}) \), the velocity would be a weighted one with randomness:

\[ v_i(t + 1) = g \cdot v_i(t) + d \cdot r \]  

(4)

Where, \( d \) represents the nuptial dance coefficient. And \( r \) is the random number in uniform distribution and selected from the domain \([-1, 1]\).

2.2. Movements of female mayflies
In order to reproduce themselves of swarms, the female mayflies would be fat and store more energy to give birth. Similar operation as equation (1) would be follow for female individuals to update their positions. However, the female mayflies are in duty to reproduce themselves and consequently, their velocity would be updated based on whether they are better than the male mayflies or not: if \( f(y_i) > f(x_i) \), then:

\[ v_i(t + 1) = g \cdot v_i(t) + a_3 e^{-\beta r^2} [x_{hi} - y_i(t)] \]  

(5)

Where, \( a_3 \) is also a constant, and \( r_{mf} \) represents the Cartesian distance between the male and female mayflies.

Similar to the male mayflies, if \( f(y_i) < f(x_i) \), the female mayflies also changed their updating equation:

\[ v_i(t) = g \cdot v_i(t) + f l \cdot r \]  

(6)

Where, \( fl \) is a random walk coefficient. And \( r \) is another random number in uniform distribution in domain \([-1, 1]\).

2.3. Mating of mayflies
During iterations, the male and female mayflies would update their positions and then, they would sort by their fitness values, mating would be selected as a rule that the best female would be the mate for the best male, the second best become the second pair, and so on. Half of them would be chosen to reproduce themselves and two offspring would be generated with the following ways:

\[ \text{offspring1} = L \times \text{male} + (1 - L) \times \text{female} \]  

(7)

\[ \text{offspring2} = L \times \text{female} + (1 - L) \times \text{male} \]  

(8)

Furthermore, the two offspring would also be mutated with a proportion. And then the top numbers of male individuals would be chosen and the rest of them would be female mayflies in the next iteration.
3. THE CHAOTIC MO ALGORITHM WITH CHEBYSHEV MAP

Chebyshev map would fluctuate between -1 and 1, as shown in Figure 1. Chebyshev chaos is similar to the random numbers in uniform distribution. Therefore, Chebyshev chaos could be used to replace the random numbers \( r_1 \) and \( r_2 \) in uniform distributions. And then, equations (4) and (6) would be changed as follows:

\[
v_i(t + 1) = g \cdot v_i(t) + d \cdot c_{r1}
\]

(9)

\[
v_i(t) = g \cdot v_i(t) + f_l \cdot c_{r2}
\]

(10)

Where, \( c_{r1} \) and \( c_{r2} \) are two chaotic numbers randomly selected from the chaos.

Obviously, there would be very similar for the improved MO and the original MO algorithm. We would introduce simulation experiments to verify the capability of the improved algorithm in the next section.

4. SIMULATION EXPERIMENTS

In this section, we would carry on several kinds of experiments to verify the capabilities of algorithms in optimizing the benchmark functions. The Monte Carlo method would also be introduced to reduce the influence of randomness involved in the algorithms.

4.1. Simulation experiments on unimodal benchmark functions

The unimodal benchmark functions only have one local optima, so they are easy to optimize for almost every algorithm. In this simulation experiment, we would introduce Step 2 function:

\[
f(x) = \sum_{i=1}^{d} (\lfloor x_i + 0.5 \rfloor)^2
\]

(11)

Step 2 function is unimodal and the best fitness value is zero at the Origin. The profile for Step 2 function is smooth and round, as shown in Figure 2. The best point would be easy to approach, just as the quick convergent curve shown in Figure 3.

Apparently, both the improved and original MO algorithms would find the best solutions while the improved one would fetch the best solution faster and steadier.

4.2. Simulation experiments on multimodal benchmark functions

Traditionally, the multimodal benchmark functions would have many local optima, and consequently, they are difficult to optimize.
In this experiment, we would use Pathological function:

\[
f(x) = \sum_{i=1}^{d-1} \left( 0.5 + \frac{\sin^2 \sqrt{100x_i^2 + x_{i+1}^2} - 0.5}{1 + 0.001(x_i^2 - 2x_ix_{i+1} + x_{i+1}^2)} \right)
\]  

(12)

Pathological function has many local optima, as shown in its three-dimensional profiles in Figure 4. Results were shown in Figure 5. Final results in this time do not verify the better performance for the improved MO algorithm. However, the residual errors seem to be more decreased in steadier state than that of the original algorithm.

4.3. Simulation experiments on non-symmetric benchmark functions

Regardless of the normal characteristics of benchmark functions such as the dimensionality, modality, or separability[10], the symmetry would also reduce the performance greatly.
Literal researches have proved that if the location of the global optima are not at the Origin, the optimization algorithms would perform bad\textsuperscript{[11]}, even sometimes fail to do so. If the global optima is not at the Origin, then the profile would not be axial symmetric or mirror symmetric in the whole domain. We call such characteristic symmetry.

\[
\begin{align*}
  f(x) &= 2.345811576101292 \\
  &\quad + \left(1 - 8x_1 + 7x_1^2 - \frac{7}{3}x_1^3 + \frac{1}{4}x_1^4\right)x_2^2 e^{-x_2}
\end{align*}
\]  \hspace{1cm} (13)

Hosaki’s function has its global position at point (4, 2), as shown in Figure 6. Results of this experiments were shown in Figure 7. Apparently, both of the original and the improved MO algorithm could perform well in optimizing the non-symmetric benchmark functions.
5. DISCUSSIONS AND CONCLUSIONS

In this paper, we introduce Chebyshev maps to replace the random numbers involved in the MO algorithm, which was just proposed recently.

Simulation experiments showed that the improved algorithm would perform a little better than the original one. The improved MO algorithm with Chebyshev maps would result in a better performance in either the steadiness, or accuracy.

However, simulation experiments did not confirm the better performance for all of the benchmark functions involved in this paper. While we found that the MO algorithm, either improved or not, could perform well in optimizing non-symmetric benchmark functions.

Therefore, the improved MO algorithm with Chebyshev map might not result in better performance as expected. Other methods or chaotic mapping or replacements of other variables might be considered in the future.
ACKNOWLEDGMENTS
The authors would like to thank the supports of the following projects: The second batch of scientific research team of Jingchu University of technology with grant number TD202001. The general Excellent Students Work Funding Project of Hubei Provincial Colleges with grant number 2019XGJPB3013. The key research and development project of Jingmen with grant number 2019YFZD009 and 2020YFYB033. Hubei Provincial Natural Science Foundation with grant number 2019CFF661. The research project of Hubei Provincial Department of Education with grant number B2019213. The cultivatable science foundations of Jingchu University of technology with grant number PY201903 and PY202003.

REFERENCES
[1] Seyedali Mirjalili, Andrew Lewis. The Whale Optimization Algorithm[J]. Advances in Engineering Software, 2016, 95: 51-67. https://doi.org/10.1016/j.advengsoft.2016.01.008
[2] Zheng-Ming Gao, Juan Zhao. An Improved Grey Wolf Optimization Algorithm with Variable Weights[J]. Computational Intelligence and Neuroscience, 2019, 2019: 2981282. 10.1155/2019/2981282
[3] Seyedali Mirjalili, Seyed Mohammad Mirjalili, Andrew Lewis. Grey Wolf Optimizer[J]. Advances in Engineering Software, 2014, 69 (0): 46-61.
http://dx.doi.org/10.1016/j.advengsoft.2013.12.007
[4] Alireza Farshin, Saeed Sharifian. A chaotic grey wolf controller allocator for Software Defined Mobile Network (SDMN) for 5th generation of cloud-based cellular systems (5G)[J]. Computer Communications, 2017, 108: 94-109.
https://doi.org/10.1016/j.comcom.2017.05.003
[5] Gaganpreet Kaur, Sankalap Arora. Chaotic whale optimization algorithm[J]. Journal of Computational Design and Engineering, 2018, 5 (3): 275-284.
https://doi.org/10.1016/j.jcde.2017.12.006
[6] A. H. Gandomi, X. S. Yang, S. Talatahari, et al. Firefly algorithm with chaos[J]. Communications in Nonlinear Science and Numerical Simulation, 2013, 18 (1): 89-98.
http://dx.doi.org/10.1016/j.cnsns.2012.06.009
[7] Jiann-Horng Lin, Chao-Wei Chou, Chorng-Horng Yang, et al. A Chaotic Levy Flight Bat Algorithm for Parameter Estimation in Nonlinear Dynamic Biological Systems[J]. Journal of Computer and Information Technology, Summer 2012, 2 (2): 56-73.
[8] Zheng-ming GAO, Su-Ruo LI, Juan ZHAO, et al.The improved Equilibrium Optimization Algorithm with Tent Map: 2020 IEEE 5th International Conference on Computer and Communication Systems (ICCCS), Shanghai, China, 2020[C]. IEEE, 343-347.
[9] Konstantinos Zervoudakis, Stelios Tsafarakis. A mayfly optimization algorithm[J]. Computers & Industrial Engineering, 2020, 145: 106559. https://doi.org/10.1016/j.cie.2020.106559
[10] Momin Jamil, Xin-She Yang. A literature survey of benchmark functions for global optimization problems[J]. Int. Journal of Mathematical Modelling and Numerical Optimisation, 2013, 4 (2): 150-194.
[11] Peifeng Niu, Songpeng Niu, Nan Liu, et al. The defect of the Grey Wolf optimization algorithm and its verification method[J]. Knowledge-Based Systems, 2019, 171: 37-43.
HTTPS://DOI.ORG/10.1016/J.KNOSYS.2019.01.018