An investigation of the effects of chaotic maps on the performance of metaheuristics

Iannick Gagnon | Alain April | Alain Abran

Software and Information Technology Engineering Department, École De Technologie Supérieure, Quebec, Canada

Correspondence
Iannick Gagnon, Software and Information Technology Engineering Department, École de Technologie Supérieure, 1100 Notre-Dame W., Montreal, Quebec H3C 1K3, Canada. Email: iannick.gagnon.1@ens.etsmtl.ca

Abstract
This article presents an empirical investigation of the effects of chaotic maps on the performance of metaheuristics. Particle Swarm Optimization and Simulated Annealing are modified to use chaotic maps instead of the traditional pseudo-random number generators and then compared on five common benchmark functions using nonparametric null hypothesis statistical testing. Contrary to what has often been assumed, results show that chaotic maps do not generally appear to increase the performance of swarm metaheuristics in a statistically significant way, except possibly for noisy functions. No performance differences were observed with the single-state Simulated Annealing algorithm. Finally, it is shown that sequence effects may be responsible for the observed performance increase. These findings reveal new research directions in using chaotic maps for metaheuristics research. The MATLAB code used in this article is available in a GitHub repository for suggestions and/or corrections.

KEYWORDS
chaotic maps, metaheuristics, particle swarm optimization, simulated annealing

1 | INTRODUCTION

Metaheuristics use randomness to diversify the search process and escape local minima. A recent trend (Figure 1) in metaheuristics research is to replace traditional pseudo-random number generators (PRNGs) by chaotic maps. Figure 1 shows an increasing trend in yearly publications on metaheuristics with chaotic maps as reported in Web of Science using the query “chaos AND optimization” between 2010 and 2018. It is a re-emerging phenomenon dating back to the mid-1990s (e.g., Reference 2).

This movement leaves important questions unanswered. For example, there does not appear to be any fundamental reason chaotic maps should increase the performance of metaheuristics. Articles on the subject generally conclude that chaotic maps improve the performance of metaheuristics, but do not investigate further. This article gathers evidence for performance differences between metaheuristics with and without chaotic maps. More data is required partly because null hypothesis statistical testing (NHST) is rarely used in the literature. When statistical significance measures are present, as in Reference 3 for example, effect sizes are almost never discussed and there are often easily identifiable methodological flaws that prevent firm conclusions. Such flaws include problematic parameterization and arbitrary measures of performance, among others. Unfortunately, since source code is rarely made available, further investigation
is often impossible. This research distinguishes itself by using NHST with large samples, using experimental design for parameterization, reporting effect sizes, introducing the notion of sequence effects (Section 5) and making the data and the computer code available to the readers.

The rest of this article is organized as follows. Section 2 contains a brief literature review of recent applications of chaotic maps with metaheuristic algorithms. Section 3 explains the methodology used in this research. Section 5 and Section 6 present the results for PSO and simulated annealing (SA), respectively. Section 7 discusses the limitations of this research, and Section 8 offers a conclusion and proposes further research directions.

### 1.1 Chaotic maps

Chaotic maps differ from PRNGs because they are deterministic. In theory, given an infinitely precise initial condition \( x_0 \) and an iterative function of the form \( x_{t+1} = f(x_t) \), the state \( x \) of a system can be calculated at any future time \( t \). Chaotic maps are random for two reasons: (1) the fact that infinite precision is impossible and (2) the high sensitivity to initial conditions. It is therefore impossible to make good long-term predictions because the initial conditions cannot be known (i.e., stored and/or read) exactly. The error, however small, will eventually cause divergence between the ideal system and the real one. For example, Figure 2 shows the iterative function \( x_{t+1} = x_t(1 - x_t) \) for \( t \in [0, 40] \) with initial conditions \( x_0 = 0.4 \) (black) and \( x_0 = 0.4 + 10^{-8} \) (red).

Figure 3 shows that up to \( t = 15 \), the absolute difference between the two curves is nearly zero before shooting up rapidly and oscillating.
Chaotic maps can therefore be used as PRNGs that are “random enough” for metaheuristics research. Figure 4 shows the six maps plotted on the interval $T = t \in [0, 40]$.

The Chebyshev map is defined by the following iterative function:

$$x_{t+1} = \cos[t\cos(x_t)].$$  \hspace{1cm} (1)

The Chebyshev map EPDF is given in Figure 5.
The Circle map is defined by the following iterative function with $\Omega = 0.2$ and $K = 0.5$:

$$x_{t+1} = \left(x_t + \Omega - \frac{K}{2\pi} \sin(2\pi x_t)\right) \mod 1.$$  

(2)

The Circle map EPDF is given in Figure 6.

The Gauss map is defined by the following iterative function:

$$x_{t+1} = \left(\frac{1}{x_t}\right) \mod 1.$$  

(3)

The Gauss map EPDF is given in Figure 7.

The Logistic map is defined by the following iterative function with $r = 4$:

$$x_{t+1} = rx_t (1 - x_t)$$  

(4)

The Logistic map EPDF is given in Figure 8.

A variant of the Sine map is defined by the iterative formula:

$$x_{t+1} = \sin(\pi x_t).$$  

(5)

**FIGURE 6** Circle map EPDF

**FIGURE 7** Gauss map EPDF
The Sine map EPDF is given in Figure 9.

The Tent map is defined by the following recurrence formula with $\mu = 10/3$:

$$x_{t+1} = \begin{cases} 
\mu x_t & \text{for } x_t < \frac{1}{2} \\
\mu (1 - x_t) & \text{for } \frac{1}{2} \leq x_t
\end{cases} \tag{6}$$

The Tent map EPDF is given in Figure 10.

Once Equations (1)–(6) are seeded with initial values ($x_0$), they are used to produce pseudorandom sequences that replace PRNGs.

### 1.2 Particle swarm optimization and simulated annealing

The particle swarm optimization (PSO) algorithm (Algorithm 1) is used as a baseline for swarm algorithms in this paper. While it is true that the performance and behavior of PSO highly depend on parameters, many of the more recent nature-inspired metaheuristics, such as the above Bat Algorithm, are considered by some to be marginally different variants of PSO (see Reference 4 for example).
Algorithm 1. Pseudocode for PSO

1. **initialize** positions, velocities, global and personal best positions
2. **while** $t < t_{\text{max}}$ **do**
3.   **for** $i = 1$: number_of_particles
4.     $\vec{v}_{i}^{t+1} \leftarrow \omega \vec{v}_{i}^{t} + c_{1} r_{1} \left( \vec{x}^{*} - \vec{x}_{i}^{t} \right) + c_{2} r_{2} \left( \vec{p}_{i} - \vec{x}_{i}^{t} \right)$
5.     $\vec{x}_{i}^{t+1} \leftarrow \vec{x}_{i}^{t} + \vec{v}_{i}^{t+1}$
6.     **if** $f(\vec{x}_{i}^{t+1}) < f(\vec{x}^{*})$ **then**
7.         $\vec{x}^{*} \leftarrow \vec{x}_{i}^{t+1}$
8.     **end if**
9.     **if** $f(\vec{x}_{i}^{t+1}) < f(\vec{p}_{i})$ **then**
10.    $\vec{p}_{i} \leftarrow \vec{x}_{i}^{t+1}$
11. **end if**
12. **end for**
13. **end while**
14. **return** $\vec{x}^{*}$

PSO updates the particles’ positions in $(\vec{x}_{i})$ $d$-dimensional space using what is called a velocity term $(\vec{v}_{i})$ which is a weighted sum of the previous velocities using an inertia term $(\omega)$, the difference vector between the swarm’s best-known position $(\vec{x}^{*})$ with current positions multiplied by a constant $(c_{1})$ and the random number $(r_{1})$ and finally the difference vector between each particles’ own historical best position $(\vec{p}_{i})$ with current positions also multiplied by a constant $(c_{2})$ and a random number $(r_{2})$. For a more in-depth explanation, see Reference 5. The random numbers $r_{1}$ and $r_{2}$ are usually picked from a random uniform distribution with support $[0, 1]$ and are the ones replaced by chaotic maps. The version of PSO used in this research limits each particle’s velocity to 15% of the length of the search space (i.e., the length of the squares described in Section 5). When a particle’s velocity exceeds this value, it is truncated to the maximum allowed velocity. This is known in the PSO literature as velocity clamping.

SA (Algorithm 2) is one of the simplest metaheuristics to implement, but it has proved to be very capable. It uses a probabilistic acceptance function that allows moves in directions that decrease fitness. This trade-off allows it to escape local minima provided they are not too deep (i.e., $\Delta Q$ is small, see Algorithm 2). SA uses a decreasing temperature schedule of the form $T = f(t)$ and a move operator implemented as a function called GenerateNewSolution. The new position is accepted according a probabilistic function based on fitness difference ($\Delta Q$) and temperature ($T$). This article uses a linearly decreasing function from $T_{\text{max}} = 1000$ to 0 in $t_{\text{max}} = 10^{4}$ time steps. The random uniform distribution of line 5 is replaced by chaotic maps in Chaotic Simulated Annealing (CSA).
Algorithm 2. Pseudocode for SA

1. **initialize** cooling schedule, initial position, best fitness ($Q_{\text{best}}$)
2. **while** $t < t_{\text{max}}$ **do**
   3. $\vec{x}_{\text{candidate}} \leftarrow \text{GenerateNewSolution} \left( \vec{x}' \right)$
   4. $Q_{\text{candidate}} \leftarrow f \left( \vec{x}' + 1 \right)$
   5. **if** $U(0,1) < e^{-\frac{\Delta}{kT(t)}}$ **or** $Q_{\text{candidate}} > Q_{\text{best}}$ **then**
      6. $\vec{x} \leftarrow \vec{x}_{\text{candidate}}$
   **end if**
8. **end while**
9. **return** $\vec{x}$

2 | LITERATURE REVIEW

Several metaheuristics have been reworked with chaotic maps including the Firefly Algorithm, the Bat Algorithm, Cuckoo Search, Krill Herd, Fruit Fly Optimization Algorithm, Particle Swarm Optimization (PSO), Ant Colony Optimization, and more. These articles unanimously conclude that chaotic maps improve performance and explain this phenomenon mostly by an increased capacity to avoid/escape local minima.

Articles use descriptive statistics with $n = 100$ observations without NHST, but the article still concludes that some maps are better than others. The same approach is found in Reference 8 with $n = 1000$ observations. The chaotic Krill Herd algorithm in Reference 9 is declared superior to its original version without NHST and a sample size of only $n = 10$ observations. The Chebyshev map is declared superior “in terms of reliability of global optimality and algorithm success rate” to other chaotic maps in Reference 3 based on comparisons made between descriptive statistics with $n = 50$ observations. A similar approach is used in References 10, 11 to conclude that chaotic maps improve the search performance of metaheuristics. The most apparent shortcoming observed is that the surveyed literature foregoes NHST and instead uses descriptive statistics alone for performance comparisons. There is also no discussion about the practical significance of the magnitudes of the alleged differences in performance. Some articles such as Reference 12 use statistical tests to conclude that chaotic maps improve performance, but no hypotheses are given, or analysis done to explain why. The original algorithm’s random variable is sampled from a Lévy distribution that, once its EPDF is examined, shows that over 99% of its area is comprised between 0 and 0.1. This means that the original algorithm is likely too greedy for multimodal functions. The article concludes that the Gauss map (Figure 4) outperforms others. Interestingly, the Gauss map EPDF is a right-skewed, which tends to make the algorithm greedy also. In numbers, the Gauss map will return values between 0 and 0.1. It is believed that this result is inconclusive because the effect sizes are not reported and because the Lévy distribution results in poor baseline performance. Chaotic maps evidently generate much interest in the research community, but their effectiveness remains an open question.

3 | METHODOLOGY

This section describes the methodological approach used to investigate the research question. Section 4.1 describes the performance metric used to compare the algorithms. Section 4.1 covers NHST and effect size considerations. Section 4.2 describes how PSO and chaotic particle swarm optimization (CPSO) parameters were selected using experimental design. Finally, Section 4.3 describes the benchmark functions used for the experiments. The data and code that support the findings of this study are openly available at https://github.com/iangagn/ENG-2019-12-0887.

4 | DEFINITION OF PERFORMANCE

Performance is typically defined as either the best or mean value found by an optimizer over several runs. This research deviates from this standard by using a measure called the first hitting time model (FHT). It measures the number of
objective function evaluations necessary to reach the global optimum within a Euclidian distance $\epsilon$ set to $10^{-2}$. The FHT measure was selected because it combines solution quality with computational effort and therefore represents a more balanced and practical gauge of performance.

### 4.1 NHST and effect size

Preliminary experiments were run to decide between parametric and nonparametric statistical tests. It was found that median FHT distributions follow exponential-like distributions. The nonparametric two-sided Wilcoxon Rank-Sum test was selected to test the null hypothesis that two FHT distributions (i.e., two algorithms) have equal medians against the alternative hypothesis that they do not. The homogeneity of variance assumption was validated using the modified Levene’s test presented in Reference 13. All tests were performed at the 95% confidence level ($\alpha = 0.05$). When statistical significance was observed, 95% confidence intervals were calculated for median FHT differences ($\bar{Y}_{CPSO} - \bar{Y}_{PSO}$) using 10,000 bootstrap samples.

### 4.2 Parameter selection for PSO

To make fair comparisons, PSO and CPSO parameters were defined to be the ones that give the best rank sum of median FHT values in a full factorial experiment using all benchmark functions. The considered values for $\omega$, $c_1$ and $c_2$ based on typical values found in the literature are \{0.5, 0.7, 0.9, 1.5\} and the numbers of particles ($n$) considered are \{20, 30, 40, 50\} for a total of $4^4 = 256$ configurations. Each configuration is run $10^3$ times until convergence to within $\epsilon = 10^{-2}$ of the global minimum. The resulting best parameters are given in Table 1.

### 4.3 Test bench

The test bench comprises the five De Jong test functions first described in Reference 14. These functions were selected because of their distinct characteristics: $f_1$ (Figure 11) is convex, $f_2$ (Figure 12) is multimodal with a low-gradient valley, $f_3$ (Figure 13) has multiple discontinuous flat regions, $f_4$ (Figures 14 and 15) is a convex quartic function with noise and

| Parameter | Value |
|-----------|-------|
| $n$       | 40    |
| $\omega$  | 1.5   |
| $c_1$     | 0.7   |
| $c_2$     | 0.9   |

**TABLE 1** Parameters for particle swarm optimization (PSO) and CPSO

![FIGURE 11 Sphere function ($f_1$)](image)
**FIGURE 12** Inverted Rosenbrock function ($f_2$)

**FIGURE 13** Step function ($f_3$)

**FIGURE 14** Level curves for Step

$f_5$ (Figure 16) is a combination of multiple steep basins with local minima and a large plateau. The small size of the test bench makes the results more amenable to analysis.

\[
f_1(\vec{x}) = \sum_{i=1}^{n} x_i^2. \tag{7}
\]

The domain for $f_1$ is usually $-5.12 \leq x_i \leq 5.12$ and its minimum is $\min(f_1) = f_1(0, \ldots, 0) = 0$.

\[
f_2(\vec{x}) = \sum_{i=1}^{n-1} \left[ 100 \cdot (x_{i+1} - x_i^2)^2 + (x_i - 1)^2 \right]. \tag{8}
\]
The domain for $f_2$ is usually $-5.12 \leq x_i \leq 5.12$ and its minimum is $\text{min}(f_2) = f_2(1, \ldots, 1) = 0$. The plateaus can be seen more easily on Figure 14 that shows level curves for some fixed $x_i$.

$$f_3(\vec{x}) = 6n + \sum_{i=1}^{n} |x_i|.$$  \hfill (9)

The domain for $f_3$ is usually $-5.12 \leq x_i \leq 5.12$ and its minimum is $\text{min}(f_3) = f_3([-5.12, 6-5], \ldots, [-5.12, -5]) = 0$.

$$f_4(\vec{x}) = ix^4 + N(0, 1).$$  \hfill (10)

The domain for $f_4$ is usually $-1.28 \leq x_i \leq 1.28$ and its minimum is $\text{min}(f_4) = f_4(0, \ldots, 0) = 0$. It can be seen from the level curves that $f_4$ is highly nonconvex.

$$f_5(\vec{x}) = \left( \frac{1}{K} + \sum_{j=1}^{25} \frac{1}{c_j + \sum_{i=1}^{2} (x_i - a_{ij})^6} \right)^{-1}.$$  \hfill (11)

$$a_{ij} = \begin{bmatrix} -32 & -16 & 0 & 16 & 32 & \ldots & 0 & 16 & 32 \\ -32 & -32 & -32 & -32 & -32 & \ldots & 32 & 32 & 32 \end{bmatrix}.$$  

$$K = 500; c_j = j.$$

The domain for $f_5$ is $-65.536 \leq x_i \leq 65.536$ and its minimum is $\text{min}(f_5) = f_5(-32, \ldots, -32) \approx 0.9980$. 
5 | RESULTS AND ANALYSIS FOR PSO

Table 2 contains the median FHT values for PSO and CPSO. The p-values for the Wilcoxon Rank-Sum test (Table 3) are calculated using the normal approximation (see Reference 15, p. 613 for details). Statistically significant p-values (p < 0.05) are boldfaced.

Table 3 reports 95% confidence intervals for the effect sizes of statistically significant median differences (Table 2) with respect to original PSO. Statistically significant results were observed for f₁ and f₃ only. For f₁, it is observed that PSO’s performance degrades with the introduction of chaotic maps. It is hypothesized that may be caused by the order in which the random values are presented. This is called a sequence effect. Since original PSO uses a uniform distribution whose PDF closely approximates the Tent map’s Empirical Probability Density Function (EPDF) (Figure 10), one should not expect a statistically significant impact on performance unless sequence effects were present. The fact that the Logistic and Sine maps degrade performance to similar extents also points in the direction of sequence effects since their EPDFs are similar. It is also hypothesized that the Chebyshev maps’ distinctively poor performance is because its support is [−1, 1]. For a convex function like f₁, this results in fitness-decreasing moves away from the global optimum half of the time. The only case where a statistically significant positive effect was observed is with f₄ and the Chebyshev map. The effect size confidence interval was narrowed down from the value given in Table 4 to (−80, −440) with 5000 runs (p = 0.01), which represents a 2%–11% improvement over original PSO. Further experimentation showed that when the Chebyshev map is randomly shuffled, the median FHT difference was not statistically significant on f₄ (p = 0.08) for n = 10³ samples. This supports the hypothesis that sequence effects are responsible for the observed statistically significant performance increase.

**TABLE 2** Median first hitting time model for particle swarm optimization (PSO) and CPSO

| Map    | f₁    | f₂    | f₃    | f₄    | f₅    |
|--------|-------|-------|-------|-------|-------|
| Uniform| 480   | 18,160| 59,200| 4120  | 55,580|
| Chebyshev| 1200  | 15,440| 58,100| 3680  | 53,400|
| Circle | 380   | 17,640| 62,660| 4080  | 56,660|
| Gauss  | 440   | 16,680| 55,000| 4220  | 57,880|
| Logistic| 840   | 16,720| 58,700| 3840  | 53,360|
| Sine   | 840   | 17,760| 60,800| 3800  | 56,840|
| Tent   | 880   | 18,020| 64,440| 3880  | 62,280|

**TABLE 3** Wilcoxon Rank-Sum test p-values for CPSO

| Map    | f₁    | f₂    | f₃    | f₄    | f₅    |
|--------|-------|-------|-------|-------|-------|
| Chebyshev| 0.00  | 0.40  | 0.61  | 0.01  | 0.27  |
| Circle | 0.25  | 0.25  | 0.61  | 0.48  | 0.95  |
| Gauss  | 0.34  | 0.13  | 0.96  | 0.81  | 0.75  |
| Logistic| 0.00  | 0.36  | 0.80  | 0.15  | 0.30  |
| Sine   | 0.00  | 0.35  | 0.56  | 0.08  | 0.95  |
| Tent   | 0.00  | 0.94  | 0.32  | 0.36  | 0.08  |

**TABLE 4** Particle swarm optimization (PSO) and CPSO median first hitting time model differences 95% CI

| Map    | f₁    | f₂    | f₃    | f₄    | f₅    |
|--------|-------|-------|-------|-------|-------|
| Chebyshev| (680,800) | —     | —     | (−80,−980) | —     |
| Circle | —     | —     | —     | —     | —     |
| Gauss  | —     | —     | —     | —     | —     |
| Logistic| (320,460) | —     | —     | —     | —     |
| Sine   | (280,440) | —     | —     | —     | —     |
| Tent   | (360,520) | —     | —     | —     | —     |
6 | RESULTS FOR SIMULATED ANNEALING

Table 5 contains the median FHT values for SA and CSA. The p-values for the Wilcoxon Rank-Sum test (Table 6) are calculated using the normal approximation. Statistically significant p-values (p < 0.05) are boldfaced. The column for \( f_5 \) is empty because SA and CSA failed to converge within 10,000 objective function evaluations. SA and CSA are simply not well suited for highly multimodal functions because they are single state methods. The initial temperature is set to \( T_0 = 1000 \).

Since there are no statistically significant performance differences, the effect size table (e.g., Table 4) is omitted. This suggests that the sequence effects identified in Section 5 may only apply to swarm metaheuristics like PSO. Possibly this is because it has a specific influence on swarm diversity and therefore the exploration and exploitation capabilities of the algorithms.

7 | LIMITATIONS OF THIS STUDY

As stated in Section 1, this study collects evidence for possible differences in performance between metaheuristics that use chaotic maps and those that use traditional PRNGs. One limitation of this study is that no a priori power analysis was performed. The reason for this is that since no distributional assumptions are made, there are no computationally affordable methods to estimate statistical power for a specified effect size. What is more, the effect size is difficult to fix in advance since the median FHT is not known a priori. This could be partly circumvented in further studies by using probabilistic distribution fitting to FHT distributions obtained for small sample sizes and Monte Carlo simulation. It should be noted that despite this, the sample sizes of \( n = 1000 \) used in this research are considerably larger than what was generally observed in the literature (see Section 2 for details). This results in greater statistical power. Another limitation of this study is that problem dimensionality is fixed (i.e., 2). As such, no conclusions are made about the scalability of the effects of chaotic maps on performance with respect to problem size. The reason for this is that since it was shown in Section 2 that research on chaotic maps and metaheuristics is not yet mature, the purpose is to gather evidence that may or may not warrant further research into subtopics like scalability. Finally, it should be noted that chaotic maps can be applied to other aspects of metaheuristics such as position initialization, but this research focuses on the most used method which consists of replacing the random number generators “inside” the algorithm.
8 | CONCLUSION

The results presented in Section 6 and Section 7 show that chaotic maps do not improve performance in a statistically significant and general way for PSO and SA. For PSO, performance degraded between 58% and 444% on the convex function $f_1$ when chaotic maps were used. There were four cases where performance degraded against one where performance improved between 2% and 24%. In the latter case, it was found that when the chaotic map values are uniformly shuffled, no statistically significant performance increase was observed. These tests were performed on the only noisy function of the test bench. Further investigation could focus on the link between chaotic maps and performance on noisy test functions, which is an active area of research. It is hypothesized the observed performance increase is due to sequence effects, because the map that improved performance (i.e., Chebyshev) has an EPDF similar to the Logistic and Sine maps whose impacts were not statistically significant. Interestingly, no statistically significant performance differences were observed between SA and CSA, which suggests that chaotic maps have a different impact on swarm metaheuristics than on single state ones. This may be due to the effects on swarm diversity, which is known to influence the exploration and exploitation properties of swarm metaheuristics. This could be easily verified empirically in further experiments. Finally, sequence effects could be examined further by using time series analysis to expose the properties of chaotic maps under a different framework.

PEER REVIEW INFORMATION

Engineering Reports thanks the anonymous reviewers for their contribution to the peer review of this work.

DATA AVAILABILITY STATEMENT

The data and code that support the findings of this study are openly available at https://github.com/iangagn/ENG-2019-12-0887.

CONFLICT OF INTEREST

The authors declare no potential conflict of interest.

AUTHOR CONTRIBUTIONS

Iannick Gagnon: Conceptualization; data curation; formal analysis; investigation; methodology; resources; software; validation; writing-original draft; writing-review and editing. Alain April: Investigation; methodology; project administration; supervision; validation; writing-original draft; writing-review and editing. Alain Abran: Methodology; supervision; writing-original draft; writing-review and editing.

ORCID

Iannick Gagnon https://orcid.org/0000-0002-6446-5187

REFERENCES

1. Clerc M. Guided Randomness in Optimization. (Vol. 1). John Wiley & Sons; 2015
2. Chen L, Aihara K. Chaotic simulated annealing by a neural network model with transient chaos. Neural Netw. 1995;8(6):915-930.
3. Mitic M, Vukovic N, Petrovic M, Miljkovic Z. Chaotic fruit fly optimization algorithm. Knowl Based Syst. 2015;89:446-458.
4. Gagnon I, April A, Abran A. A critical analysis of the bat algorithm. Eng Rep. 2020;2(8):e12212.
5. Eberhart R, Kennedy J. Particle swarm optimization. Paper presented at: Proceedings of the IEEE International Conference on Neural Networks; Vol. 4, November 1995:1942-1948.
6. Gandomi AH, Yang XS, Talatahari S, Alavi AH. Firefly algorithm with Chaos. Commun Nonlinear Sci Numer Simulat. 2013;18(1):89-98.
7. Gandomi AH, Yang XS. Chaotic bat algorithm. J Comput Sci. 2014;5(2):224-232.
8. Wang GG, Deb S, Gandomi AH, Zhang Z, Alavi AH. Chaotic cuckoo search. Soft Comput. 2016;20(9):3349-3362.
9. Saremi S, Mirjalili SM, Mirjalili S. Chaotic krill herd optimization algorithm. Proc Technol. 2014;12:180-185.
10. Alatas B, Akin E, Ozer AB. Chaos embedded particle swarm optimization algorithms. Chaos Soliton Fract. 2009;40(4):1715-1734.
11. Cai J, Ma X, Li L, Yang Y, Peng H, Wang X. Chaotic ant swarm optimization to economic dispatch. Electr Pow Syst Res. 2007;77(10):1373-1380.
12. Sayed GI, Tharwat A, Hassanien AE. Chaotic dragonfly algorithm: an improved metaheuristic algorithm for feature selection. Appl Intell. 2019;49(1):188-205.
13. Brown MB, Forsythe AE. Robust tests for the equality of variances. J Am Stat Assoc. 1974;69(346):364-367.
14. De Jong KA. *Analysis of the Behavior of a Class of Genetic Adaptive Systems* (Doctoral dissertation); 1975.
15. Walpole RE, Myers RH. *Probability and Statistics for Engineers and Scientists*. New York: Macmillan, Pearson Education Limited; 2002.

How to cite this article: Gagnon I, April A, Abran A. An investigation of the effects of chaotic maps on the performance of metaheuristics. *Engineering Reports*. 2021;3:e12369. [https://doi.org/10.1002/eng2.12369](https://doi.org/10.1002/eng2.12369)