Extended Initial Study on the Performance of Enhanced PSO Algorithm with Lozi Chaotic Map

Michal Pluhacek, Vera Budikova, Roman Senkerik, Zuzana Oplatkova, and Ivan Zelinka

Tomas Bata University in Zlin, Faculty of Applied Informatics,
Nam T.G. Masaryka 5555, 760 01 Zlin, Czech Republic
{pluhacek,budikova,senkerik,oplatkova,zelinka}@fai.utb.cz

Abstract. In this paper, it is proposed the utilization of discrete Lozi map based chaos random number generator to enhance the performance of PSO algorithm with inertia weight. Performance tests and results are presented. Results are analyzed and compared with another evolutionary algorithm. Tuning experiment was performed.

1 Introduction

PSO algorithm is one of evolutionary algorithms (class of soft computing methods that are inspired by nature). In recent years, various evolutionary algorithms were designed and used with great results in many areas of optimization [1-5]. More recently some studies indicated that using of chaotic number generators may improve the performance of evolutionary optimization algorithms on such tasks as PID controller design [6] or fuzzy modelling of an experimental thermal-vacuum system [7].

This initial study is focused on the investigation on the performance of PSO algorithm with implemented chaotic Lozi map as a random number generator. Firstly, Particle swarm optimization algorithm is explained. The next sections are aimed on the description of used chaotic systems and benchmark test functions. Results and conclusion follow afterwards.

2 Particle Swarm Optimization Algorithm

Particle swarm optimization (PSO) algorithm is based on the natural behaviour of birds and fishes. It was firstly introduced by Eberhart and Kennedy in 1995 [1,2] as an alternative to genetic algorithms [4] and differential evolution [5],

In each generation, a new location of a particle is calculated based on its previous location and velocity (or "velocity vector"). One of PSO algorithm disadvantages is the rapid acceleration of particles which causes them abandoning the

I. Zelinka et al. (Eds.): Nostradamus: Mod. Meth. of Prediction, Modeling, AISC 192, pp. 167–177.
springerlink.com

defined area of interest. For this reason, several modifications of PSO were introduced to handle with this problem. Main principles of PSO algorithm and its modifications are well described in [1-3].

Within this research, chaos driven PSO strategy with inertia weight was used. The selection of inertia weight modification of PSO was based on numerous previous experiments. Default values of all PSO parameters were chosen according to the recommendations given in [2, 3].

Inertia weight is designed to influence the velocity of each particle differently over the time [8]. In the beginning of the optimization process, the influence of inertia weight factor w is minimal. As the optimization continues, the value of w is decreasing, thus the velocity of each particle is decreasing, since w is always the number < 1 and it multiplies previous velocity of particle in the process of new velocity value calculation. Inertia weight modification PSO strategy has two control parameters w_{start} and w_{ends}. New w for each generation is then given by Eq. 1, where i stand for current generation number and n for total number of generations.

$$w = w_{start} - \frac{\left(\left(w_{start} - w_{end}\right) * i\right)}{n} \tag{1}$$

Chaos driven random number generator is used in the main PSO formula (Eq. 2) that determines new "velocity" and thus the position of each particle in the next generation (or migration cycle).

$$v(t+1) = v(t) + c_1 \cdot Rand \cdot (pBest - x(t)) + c_2 \cdot Rand \cdot (gBest - x(t)) \tag{2}$$

Where:
v(t+1) – New velocity of particle.
v(t) – Current velocity of particle.
c1,c2 – Priority factors.
pBest – Best solution found by particle.
gBest – Best solution found in population.
x(t) – Current position of particle.
Rand – Random number, interval <0,1>. Within Chaos PSO algorithm, the basic inbuilt computer (simulation software) random generator is replaced with chaotic generator (in this case, by using of Lozi map).

New position of particle is then given by Eq. 3, where x(t+1) is the new position:

$$x(t+1) = x(t) + v(t+1) \tag{3}$$

3 Chaotic Lozi Map

The Lozi map is a simple discrete two-dimensional chaotic map. The Lozi map is depicted in Fig. 1. The map equations are given in Eq. 4 and 5. The parameters used in this work are: a = 1.7 and b = 0.5 as suggested in [9].

$$X_{n+1} = 1 - a|X_n| + bY_n \tag{4}$$

$$Y_{n+1} = X_n \tag{5}$$

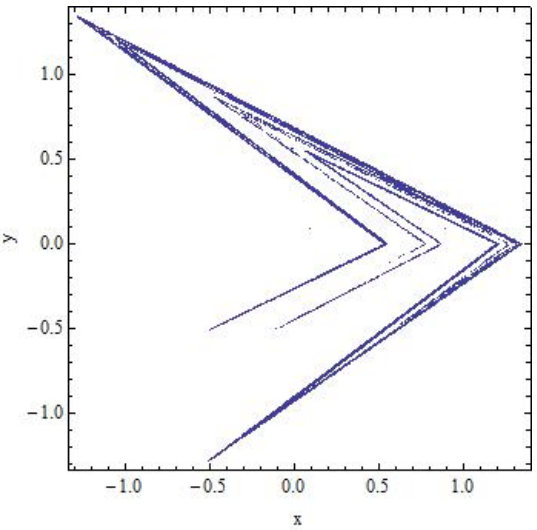

Fig. 1 Lozi map

4 Experiment Setup

In section 2 the PSO with inertia weight was described. To compare the impact of using Lozi map as a chaotic number generator, performance tests were performed for both PSO with chaotic and non-chaotic random number generator. Classic version of PSO with inertia weight modification is labelled PSO Weight. The proposed novel Lozi map enhanced PSO with inertia weight is labelled PSO Lozi. For further comparison with different heuristic, tests were also performed for differential evolution (DE) and its strategy DERand1Bin.

Basic PSO control parameters were set based on previous experiments and literature [1-4,7] as follows:

Population size: 30 (for sections 5.1 – 5.4), 50, 75, 100, 150, 200, 300, 400
Iterations / generations: 10 * dimension
w_{start}: 0.9
w_{end}: 0.4
Dimension: 2, 5, 10, 20, 40

The algorithms were tested on 4 different benchmark functions. From the statistical reasons, optimization for each dimension value was repeated 30 times.

4.1 Test Functions

The First De Jong`s function is given by Eq. 6.

$$f(x) = \sum_{i=1}^{dim} x_i^2 \tag{6}$$

Function minimum:
Position for E_n: $(x_1,x_2\ldots x_n) = E_n$: $(x_1,x_2\ldots x_n) = (0,0,\ldots, 0)$
Value for E_n: $y = 0$

The Second De Jong`s function is given by Eq. 7.

$$f(x) = \sum_{i=1}^{dim-1} 100\,(x_i^2 - x_{i+1})^2 + (1 - x_i)^2 \tag{7}$$

Function minimum:
Position for E_n: $(x_1,x_2\ldots x_n) = (1,1,\ldots,1)$
Value for E_n: $y = 0$

Rastrigin`s function is given by Eq. 8.

$$f(x) = 10dim \sum_{i=1}^{dim} x_i^2 - 10\cos(2\pi x_i) \tag{8}$$

Function minimum:
Position for E_n: $(x_1,x_2\ldots x_n) = (0,0,\ldots,0)$
Value for E_n: $y = 0$
Schwefel`s function is given by Eq. 9.

$$f(x) = \sum_{i=1}^{dim} -x_i \sin\left(\sqrt{|x_i|}\right) \tag{9}$$

Function minimum:
Position for E_n: $(x_1,x_2\ldots x_n) = (420.969, 420.969,\ldots, 420.969)$
Value for E_n: $y = -418.983 * \text{dimension}$

5 Results and Brief Analysis

The results of experiments and brief commentary on these results follow in this section. Following Tables 1 – 8 contain the best, the worst and the median of obtained

final results for all 30 runs of evolutionary algorithm. For the comparison between algorithms, the best individual results are highlighted by bold number in all tables. Results of PSO algorithm are also compared with the performance of DE.

5.1 The First De Jong`s Function

Following tables 1 and 2 contain results for 1st De Jong´s function. Proposed implementation of chaotic Lozi map to PSO algorithm seems to have improved the performance of the algorithm. Values for PSO Lozi (chaos number generator) are better in all cases than values for PSO Weight (classic number generator). Furthermore those results are better or comparable with those of DE. However PSO seems to be less efficient in solving higher dimension with the setting used than DE.

Table 1 Results for the first De Jong`s function for dim = 2, 5 and 10

	dim = 2			dim = 5			dim = 10		
	PSO Weight	PSO LoziDE		PSO Weight	PSO LoziDE		PSO Weight	PSO LoziDE	
The worst result	2.3877E-03	**1.6038E-04**	2.3719E-04	3.9290E-03	**2.5665E-05**	1.0020E-03	3.7508E-02	2.0817E-02	**2.8169E-03**
The best result	1.1267E-05	**2.0428E-07**	2.4734E-07	1.6523E-05	**2.3375E-07**	7.4393E-05	1.4580E-04	**4.2146E-07**	5.2609E-04
Median	2.0761E-04	2.3685E-05	**2.3118E-05**	1.7489E-04	**2.8169E-06**	1.9458E-04	6.2556E-03	**6.3030E-04**	1.1871E-03

Table 2 Results for the first De Jong`s function for dim = 20 and 40

	dim = 20			dim = 40		
	PSO Weight	PSO Lozi	DE	PSO Weight	PSO Lozi	DE
The worst result	1.6283E+00	1.9538E+00	**4.0143E-02**	9.0627E+00	8.6062E+00	**1.6867E+00**
The best result	8.4300E-02	4.6076E-02	**1.0478E-02**	2.4010E-01	**2.4387E-03**	6.5759E-01
Median	4.7874E-01	4.0532E-01	**1.8644E-02**	4.4943E+00	4.2540E+00	**9.9788E-01**

5.2 The Second De Jong`s function

Results in tables 3 and 4 were obtained by optimizing the 2nd De Jong´s function. Almost similar trends as in previous section (1st De Jong´s function) can be seen in those results with the exception being the worse performance of DE in comparison with PSO Lozi for the higher dimensions.

Table 3 Results for the second De Jong`s function for dim = 2, 5 and 10

	dim = 2			dim = 5			dim = 10		
	PSO Weight	PSO Lozi	DE	PSO Weight	PSO Lozi	DE	PSO Weight	PSO Lozi	DE
The worst result	2.7002E-02	**1.5978E-02**	5.5802E-02	1.6401E-01	**1.1535E-02**	3.7692E+00	1.9219E-01	**1.7063E-02**	1.9067E+01
The best result	6.3008E-05	**2.9166E-05**	2.0637E-04	1.2517E-03	**5.1098E-05**	6.2450E-01	3.3509E-03	**8.6427E-05**	7.7341E+00
Median	3.4439E-03	**8.4753E-04**	1.0980E-02	3.3932E-02	**7.9757E-04**	2.1158E+00	3.8709E-02	**1.1631E-03**	1.4072E+01

Table 4 Results for the second De Jong`s function for dim = 20 and 40

	dim = 20			dim = 40		
	PSO Weight	PSO Lozi	DE	PSO Weight	PSO Lozi	DE
The worst result	**2.6044E-01**	3.1728E-01	5.2821E+01	1.0324E+00	**1.3342E+00**	1.3381E+02
The best result	5.2920E-03	**7.8609E-04**	2.5292E+01	7.9342E-02	**2.2454E-02**	5.0176E+01
Median	8.0123E-02	**3.6793E-02**	3.7204E+01	3.7495E-01	**2.1875E-01**	7.7725E+01

5.3 Rastrigin`s Function

In the case of the Rastrigin`s benchmark function there is no significant improvement in PSO performance, however both PSO algorithms were significantly worse for dim = 40 than those of DE (see tables 5 and 6). This finding was one of those that encouraged the extended analysis in section 6.

Table 5 Results for second Rastrigin`s function for dim = 2, 5 and 10

	dim = 2			dim = 5			dim = 10		
	PSO Weight	PSO Lozi	DE	PSO Weight	PSO Lozi	DE	PSO Weight	PSO Lozi	DE
The worst result	2.4048	1.9933	**1.7853**	13.5637	9.9854	**9.6667**	**25.1741**	27.2665	26.7314
The best result	0.0032	**0.0015**	0.0056	1.0756	**0.0022**	0.1615	**0.2599**	3.0098	6.2952
Median	0.5399	**0.3790**	0.5707	6.5017	**3.0761**	4.0811	**9.5158**	10.4711	15.7813

Table 6 Results for Rastrigin`s function for dim = 20 and 40

	dim = 20			dim = 40		
	PSO Weight	PSO Lozi	DE	PSO Weight	PSO Lozi	DE
The worst result	74.9295	**68.5083**	75.9887	186.5780	192.4930	**155.9540**
The best result	27.4552	**17.2733**	18.7197	100.0440	47.8408	**38.8948**
Median	47.5319	**46.6510**	50.0244	152.0650	159.0090	**119.5870**

5.4 Schwefel`s Function

Presented results in Tables 7 and 8 show increasing difference between values of median for PSO Weight and PSO Lozi together with increasing dimension in favour of Lozi map enhanced PSO. As in the previous case, both algorithms were surpassed by DE (most significantly in higher diemensions).

Table 7 Results for Schwefel`s function for dim = 2, 5 and 10

	dim = 2			dim = 5			dim = 10		
	PSO Weight	PSO Lozi	DE	PSO Weight	PSO Lozi	DE	PSO Weight	PSO Lozi	DE
The worst result	-673.39	-697.48	**-702.79**	-1191.59	-1262.01	**-1635.06**	-1915.02	-1980.15	**-3174.74**
The best result	-837.87	**-837.94**	-837.91	-1760.30	-1942.73	**-1964.83**	-2740.49	-3037.15	**-3562.67**
Median	**-820.35**	-819.43	-816.07	-1458.34	-1525.28	**-1842.63**	-2234.00	-2561.77	**-3372.21**

Table 8 Results for Schwefel`s function for dim = 20 and 40

	dim = 20			dim = 40		
	PSO Weight	PSO Lozi	DE	PSO Weight	PSO Lozi	DE
The worst result	-2886.88	-3133.19	**-5830.19**	-4839.42	-6112.29	**-9104.93**
The best result	-5372.33	-5722.76	**-6482.53**	-8400.45	-9935.40	**-11411.20**
Median	-3477.26	-4217.83	**-6358.48**	-6558.88	-7527.70	**-10308.10**

From the results presented in tables 1 - 8 it may be stated, that the majority of results obtained by proposed PSO Lozi algorithm were better than results of classic PSO Weight. The observed median of final cost function values for all 30 runs was better in 17 of 20 experiments - 4 benchmark functions x 5 dimensions values (see Fig. 2).

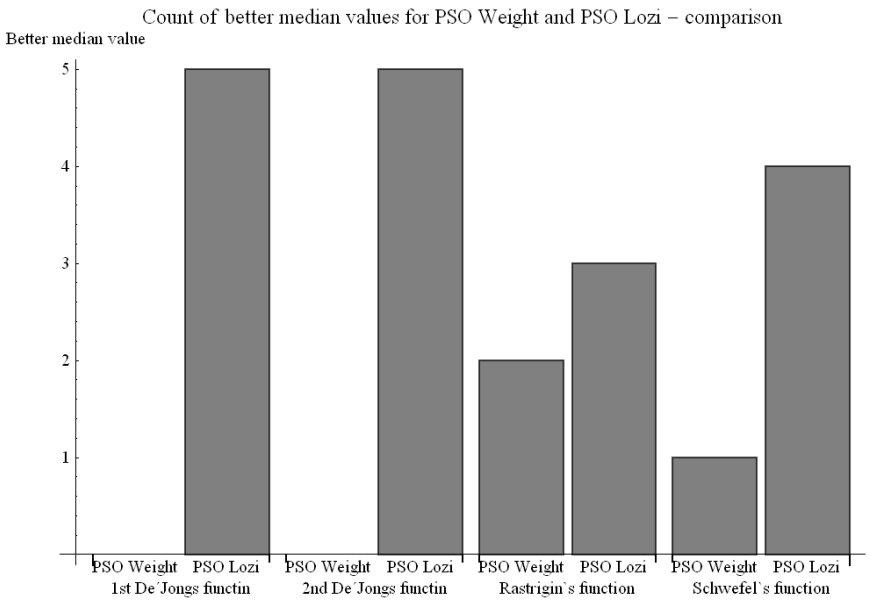

Fig. 2 PSO Weight and PSO Lozi results comparison

6 Additional Tests and Analysis

The analysis of the results presented in previous section shows an interesting phenomenon, that the performance of DE in comparison with both PSO algorithms is sometimes much better (especially in the case of Schwefel´s function). This section presents an investigation on this phenomenon. Based on the previous experiences, the PSO algorithm is able to achieve better results for higher population size (NP). To prove this theory an experiment was set-up as follows:

Population size: 30, 50, 75, 100 (Schwefel´s f. only), 150, 200, 300, 400
Iterations / generations: 200
w_{start}: 0.9
w_{end}: 0.4
Dimension: 20
Benchmark functions: Schwefel´s, 1st De Jong

Results are shown in following tables (9 and 10) and depicted on figures 3 and 4. Analysis follows afterwards.

From Tables 9 and 10, it follows, that increasing the size of population (NP) led to significant improvement in the performance of both PSO algorithms with inertia weight. Nevertheless the performance of DE had opposite trend. These trends

can be seen on figures 3 and 4. Presented results in this section support the claims, that using Lozi map as a chaotic number generator could lead to improvement of the performance of PSO algorithm thus to achieve better or at least similar results when comparing PSO algorithms with different evolutionary algorithm, which was DE.

Table 9 Mean value for 30 runs; Schwefel`s function; dim = 20; generations = 200

NP:	30	50	75	100	150	200	300	400
PSO Weight	-3697.63	-3873.62	-4140.99	-4255.84	-4329.57	-4866.41	-5316.41	-5377.72
PSO Lozi	-4340.06	-4560.42	-5032.82	-5241.78	**-5801.99**	**-5998.05**	**-6174.55**	**-6225.63**
DE	**-6100.9**	**-5737.68**	**-5649.1**	**-5500.01**	-5635.55	-5651.5	-5673.33	-5651.23

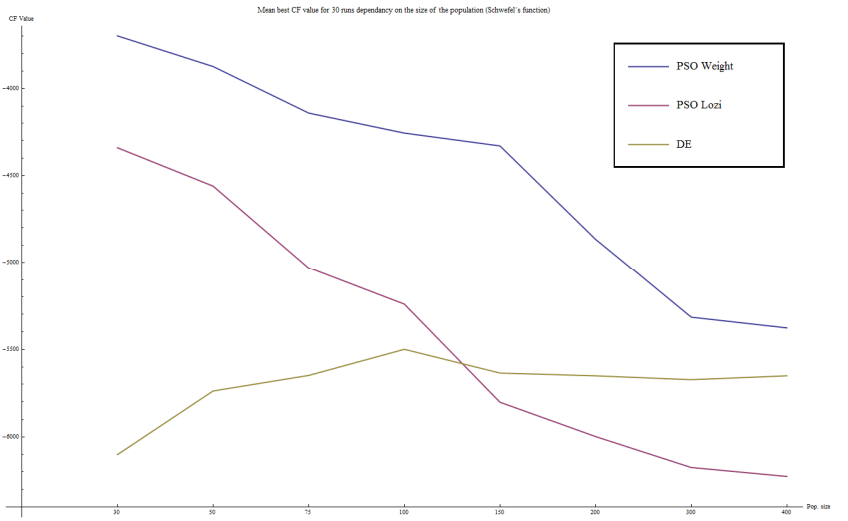

Fig. 3 PSO Weight, PSO Lozi and DE – influence of population size – Schwefel´s function

Table 10 Mean value for 30 runs; 1st De Jong`s function; dim = 20; generations = 200

NP:	150	200	300	400
PSO Weight	7.33004*10^-6	3.45288*10^-7	3.7368*10^-9	2.60345*10^-10
PSO Lozi	2.77017*10^-6	**4.12126*10^-8**	**3.66273*10^-11**	**1.03193*10^-14**
DE	**4.25565*10^-7**	4.94665*10^-7	6.11666*10^-7	7.07218*10^-7

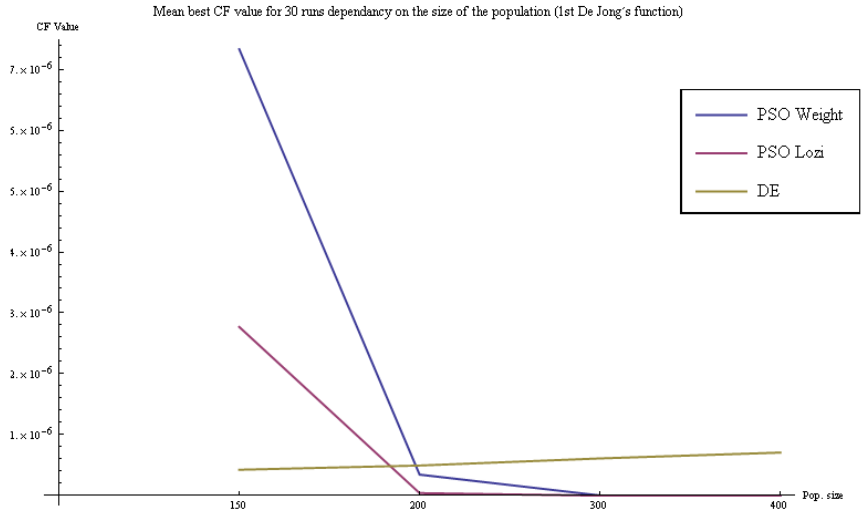

Fig. 4 PSO Weight, PSO Lozi and DE – influence of population size – 1st De Jong´s function

7 Conclusion

In this paper, the enhanced PSO algorithm with Lozi map chaos number generator was proposed and investigated. Four different test functions were used to show the performance of the proposed algorithm in comparison with classic non-chaotic version. The results obtained in this initial study seem to be promising and may indicate that chaotic Lozi map based random number generator can improve the performance of PSO algorithm. Furthermore the seemingly worse performance of chaotic PSO against DE in higher dimension was investigated. From this investigation it follows, that the population size has very significant influence on both algorithms. With increasing of size of the population DE is outperformed by PSO even though both algorithms perform the same amount of CF evaluations.

Future experiments should concentrate on tuning the control parameters of PSO and investigation on the performance of this algorithm on variety of problems. Overall, this study brought findings that chaotic Lozi map may improve the performance of PSO algorithm.

The primary aim of this work was not to develop a new type of random number generator, which should pass many statistical tests, but to try to use and test the implementation of natural chaotic dynamics into evolutionary algorithm as a chaotic random number generator.

Acknowledgements. This work was supported by European Regional Development Fund under the project CEBIA-Tech No. CZ.1.05/2.1.00/03.0089, by Internal Grant Agency of Tomas Bata University under the project No. IGA/FAI/2012/042 and by Internal Grant Agency of Tomas Bata University under the project No. IGA/FAI/2012/037.

References

1. Kennedy, J., Eberhart, R.: Particle Swarm Optimization. In: Proceedings of IEEE International Conference on Neural Networks, vol. IV, pp. 1942–1948 (1995)
2. Dorigo, M.: Ant Colony Optimization and Swarm Intelligence, Springer (2006)
3. Eberhart, R., Kennedy, J.: Swarm Intelligence. The Morgan Kaufmann Series in Artificial Intelligence. Morgan Kaufmann (2001)
4. Storn, R., Price, R.: Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces. Journal of Global Optimization 11, 341–359 (1997)
5. Goldberg, D.E.: Genetic Algorithms in Search Optimization and Machine Learning. Addison Wesley, p. 41 (1989) ISBN 0201157675
6. Davendra, D., Zelinka, I., Senkerik, R.: Chaos driven evolutionary algorithms for the task of PID control. Computers & Mathematics with Applications 60(4), 1088–1104 (2010) ISSN 0898-1221
7. Araujo, E., Coelho, L.: Particle swarm approaches using Lozi map chaotic sequences to fuzzy modelling of an experimental thermal-vacuum system. Applied Soft Computing 8(4), 1354–1364 (2008)
8. Nickabadi, A., Ebadzadeh, M.M., Safabakhsh, R.: A novel particle swarm optimization algorithm with adaptive inertia weight. Applied Soft Computing 11(4), 3658–3670 (2011) ISSN 1568-4946
9. Sprott, J.C.: Chaos and Time-Series Analysis. Oxford University Press (2003)