Chaos Powered Grammatical Evolution

Ivan Zelinka[1], Petr Šaloun[1], and Roman Senkerik[2]

[1] VSB-Technical University of Ostrava, 17. listopadu 15,
708 33 Ostrava-Poruba, Czech Republic
{ivan.zelinka,petr.saloun}@vsb.cz
[2] Faculty of Applied Informatics, Tomas Bata University in Zlin
senkerik@fai.utb.cz

Abstract. In this paper we discuss alternative nonrandom generators for symbolic regression algorithms and compare its variants powered by classical pseudo-random number generator and chaotic systems. Experimental data from previous experiments reported for genetic programming and analytical programming is used. The selected algorithms are differential evolution and SOMA. Particle swarm, simulated annealing and evolutionary strategies are in process of investigation. All of them are mutually used in scheme Master-Slave meta-evolution for final complex structure fitting and its parameter estimation.

1 Introduction

The need of problem optimization is very old and covers various disciplines including engineering, economics, physics, biology and many others. In fact, many real-world problems can be defined as an optimization problem. The goal of a typical optimization is to maximize productivity or performance of some process or device or to minimize waste. Many more or less sophisticated optimization techniques have been developed over time. While the simplest problems involving functions of a single variable may be solved using basic math, many real-world problems require more complex tools. Evolutionary processes can be in general used for many practical tasks, like robot trajectory design, plasma or chemical reactor control, aircraft wings design, scheduling problems amongst the others.

For a long time in history optimization methods have been based on still more and more complicated methods usually involving exact mathematics. However, with increasing complexity of problems to be optimized need for more powerful and flexible optimization techniques arose. In mid-sixties, evolutionary algorithms were developed to address these demands. They are considered a powerful tool with many advantages over traditional optimization techniques.

One of the biggest advantages of evolutionary algorithms is that unlike many other traditional optimization techniques, they dont depend upon mathematical models of problems. Actually, the only precondition of using an evolutionary algorithm is the ability to evaluate candidate solution. In other words, the only thing that matter is whether or not it is possible to evaluate a solution once it is presented.

K. Saeed and V. Snášel (Eds.): CISIM 2014, LNCS 8838, pp. 455–464, 2014.

An interesting extension of evolutionary algorithms is so called symbolic regression, that uses evolution to synthesize complex structures (formulas, el. circuits, computer programs) from simple building blocks (mathematical functions, el. elements, programming commands). The initial idea of symbolic regression by means of a computer program was proposed in genetic programming (GP) [1,2]. The other approach of grammatical evolution (GE) was developed in [3], [20] and analytical programming (AP) in [4]. Another interesting investigation using symbolic regression were carried out in [5] on AIS and Probabilistic Incremental Program Evolution (PIPE), which generates functional programs from an adaptive probability distribution over all possible programs. Yet another new technique is the so called *Transplant Evolution*, see [6], [7] and [8] which is closely associated with the conceptual paradigm of AP, and modified for GE. GE was also extended to include DE by [9]. Generally speaking, it is a process which combines, evaluates and creates more complex structures based on some elementary and noncomplex objects, in an evolutionary way. Such elementary objects are usually simple mathematical operators $(+, -, \times, ...)$, simple functions (sin, cos, And, Not, ...), user-defined functions (simple commands for robots – MoveLeft, TurnRight, ...), etc. An output of symbolic regression is a more complex "object" (formula, function, command,...), solving a given problem like data fitting of the so-called Sextic and Quintic problem [10,11], randomly synthesized function [11], Boolean problems of parity and symmetry solution (basically logical circuits synthesis) [12,4], or synthesis of quite complex robot control command by [2,19]. Examples mentioned in [13] are just few samples from numerous repeated experiments done by symbolic regression, which are used to demonstrate how complex structures can be produced by symbolic regression in general for different problems, see [13].

This paper focuses on grammar evolution which is an advanced evolutionary technique suitable for symbolic regression. Classical grammar evolution uses genetic algorithms as a computational core that manipulates genetic information. Further parts of this paper deal with an idea of *replacing the genetic-based core by DE or SOMA and pseudorandom generators by chaotic systems*. There has been made an experiment that measures performance of these alternative versions compared to the classical GE.

2 Used Methods and Motivation

For our experiments described here standard hardware and algorithms have been used. All important information about algorithms used in our experiments is mentioned and referred here. Standard as well as modern evolutionary algorithms were used for our experimentation. Comparing to the previous method, in this research GE with evolutionary algorithms like differential evolution (DERand1Bin), [17] and SOMA (AllToOne), [18] are used. Application of alternative algorithms like Genetic Algorithms GA and Simulated Annealing (SA), ES and/or Swarm Intelligence is in process now.

The main difference comparing to similar experiments is the fact that we are using a) not only GE based on pseudorandom number generators (PRNG)

but also on chaotic systems instead of PRNG use and b) instead of GE based operations are used algorithms DE and SOMA. All experiments were done in Mathematica 9, on MacBook Pro, 2.8 GHz Intel Core 2 Duo.

The master-slave approach was used in our experimentation, i.e. one kind of evolutionary algorithm was used like Master (to estimate general structure for data fitting with non-estimated parameters - constants) and second one s a Slave (to estimate just mentioned parameters - constants in formulas from Master process). Based on principles of DE and SOMA, individuals were of integer structure in the Master evolution.

The main motivation is based on use of chaotic dynamics instead of PRNG in previous research papers of various researchers (including our own, [26]-[28]). Till now chaos was observed in plenty of various systems (including an evolutionary one) and in the last few years it was also used to replace pseudo-number generators (PRGNs) in evolutionary algorithms (EAs). Let's mention for example research papers like one of the first use of chaos inside EAs [25], [26]-[28] discussing use of deterministic chaos inside particle swarm algorithm instead of PRGNs, [34] - [37] investigating relations between chaos and randomness or the latest one [38], [39], or another using chaos with EAs in applications, like [40] and [41], amongst the others.

Another research joining deterministic chaos and pseudorandom number generator has been done for example in [34]. Possibility of generation of random or pseudorandom numbers by use of the ultra weak multidimensional coupling of p 1-dimensional dynamical systems is discussed there. Another paper [29] deeply investigates logistic map as a possible pseudo-random number generator and is compared with contemporary pseudo-random number generators. A comparison of logistic map results is made with conventional methods of generating pseudo-random numbers. The approach used to determine the number, delay, and period of the orbits of the logistic map at varying degrees of precision (3 to 23 bits).

Logistic map that we are using here was also used in [30] like chaos-based true random number generator embedded in reconfigurable switched-capacitor hardware. Another paper [35] proposed an algorithm of generating pseudorandom number generator, which was called (couple map lattice based on discrete chaotic iteration) and combined the couple map lattice and chaotic iteration. Authors also tested this algorithm in NIST 800-22 statistical test suits and was used in image encryption.

In [36] authors exploit interesting properties of chaotic systems to design a random bit generator, called CCCBG, in which two chaotic systems are cross-coupled with each other. For evaluation of the bit streams generated by the CCCBG, the four basic tests are performed: monobit test, serial test, auto-correlation, Poker test. Also the most stringent tests of randomness: the NIST suite tests have been used. A new binary stream-cipher algorithm based on dual one-dimensional chaotic maps is proposed in [37] with statistic proprieties showing that the sequence is of high randomness. Similar studies are also done in [31], [25], [32] and [33].

Mutual comparison is discussed at the end.

2.1 Experiment Design

Our experiments have been set so that GE powered by classical pseudorandom number generator and deterministic chaos generators (see for example [14], [15] and [16]), were used with above mentioned Master-Slave approach. Based on the fact that deterministic chaos generators were successfully used in the past [14], [15] and [16] for classical evolutionary algorithms, we have selected logistic equation (1), and data series generated by this equation with setting $A = 4$ as a random numbers to replace classical classical pseudorandom number generator in symbolic regression. Algorithms selected for our experiments were SOMA [18] and differential evolution (DERand1Bin) [17]. Another algorithms like, simulated annealing (SA) [22] and [23], Evolutionary strategies (ES) [21] and Particle Swarm (PSO) [24] are in process. Here we report results for GE with SOMA (GSOMA) and GE with DE (GDE) test results.

$$x_{n+1} = Ax_n\left(1 - x_n\right) \tag{1}$$

Both DE and SOMA are well regarded for their performance when applied on various optimization problems. Genetic algorithms, on the other hand, are considered less performing. That said there is an assumption that also the grammar-enabled variants of both (GDE and GSOMA) perform better than the classical GE. This experiment aims to measure the GE, GDE and SOMA performance and to compare it mutually.

Following paragraphs describe the problem of definition and control parameters that were used in the experiment. Each test was repeated 50 times for higher accuracy.

All three mentioned evolutionary techniques were used to optimize a function fitting problem. The goal was to find a function that describes given function as closely as possible. The function was discretized into 50 points on a given interval. Candidate solutions were evaluated as follows:

1. They were discretized the same way as the original function.
2. An absolute deviation was computed in each discretized point.
3. The cost value was determined as a sum of these absolute deviations.

An ideal solution would have a zero cost value while higher numbers mean that the fitting was not perfect. That said no conversion was needed to transform cost values into fitness values as lower values already represented better solution. Furthermore, the ideal solution is represented by zero which is the optimal case.

There were two functions used in this experiment to achieve more accurate results - formulas 3 and 4.

The cost function (2) has been defined according to Eq. 2 and the main aim of the used evolution was to find formula, that gives the smallest value of Eq. 2. To verify the functionality of AP more properly, set of comparative simulations based on selected examples from Koza's GP has been done. Simulations were focused on selected examples from [2] and [10], see its description in formulas 3 (sextic) and 4 (quintic).

$$f_{cost} = \sum_{1}^{n} |data_i - synthesized\_data_i| \tag{2}$$

$$x^6 - x^4 + x^2 \tag{3}$$

$$x^5 - 2x^3 + x \tag{4}$$

Different control parameters were used for each of the evolutionary techniques (GE, GDE and GSOMA). One reason is obvious each of these techniques uses a different set of control parameters. However, even if they would not, GE e.g. is known for requiring much bigger population sizes than DE to perform equally well so setting the same parameter values would not be fair even if it was technically possible. This is apparent with different GSOMA variants where highly different numbers of migration cycles are used, yet both variants evaluate the fitness function same times. To deal with this problem, we had to find three different sets of parameters that give the best results possible for each individual evolutionary technique. These parameter sets (described in tables 1, 2, 3, 4) were used for the experiment.

Table 1. GE Control Parameters

Parameter	**Value**
Cross Rate	0.85
Selection Strategy	Roulette-wheel selection
Population Size	300
Chromosome Length	50
Generation Count	500
Elitism Level	1

Table 2. GDE Control Parameters

Parameter	**Value**
Cross Rate	0.85
Mutation Constant	0.7
Population Size	50
Chromosome Length	50
Generation Count	20000

3 Results

This part contains graphs that visualize the experiment results in different ways. The first type of diagrams represent the evolution progress as it tries to approach the null fitness value. Many lines (specifically 50 but many of them overlap) can

Table 3. GSOMA All2One Control Parameters

Parameter	Value
Path Length	3.5
Step Size	0.11
Perturbation	0.3
Population Size	50
Dimension (Vector Length)	50
Migration Cycles	1000

Table 4. GSOMA All2All Adaptive

Parameter	Value
Path Length	3.5
Step Size	0.11
Perturbation	0.3
Population Size	7
Dimension (Vector Length)	50
Migration Cycles	20

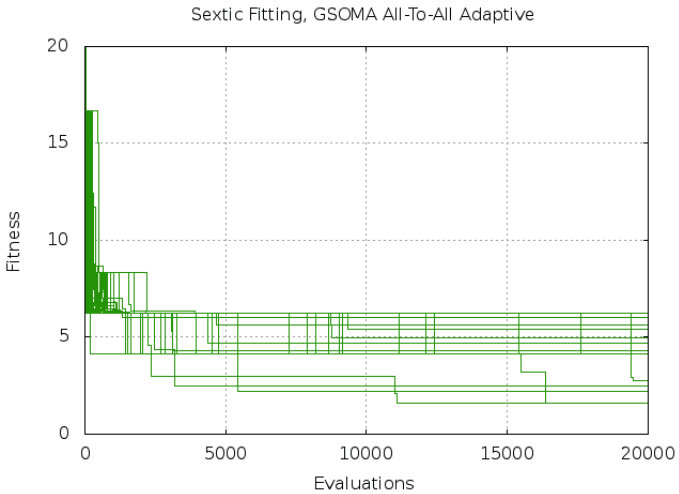
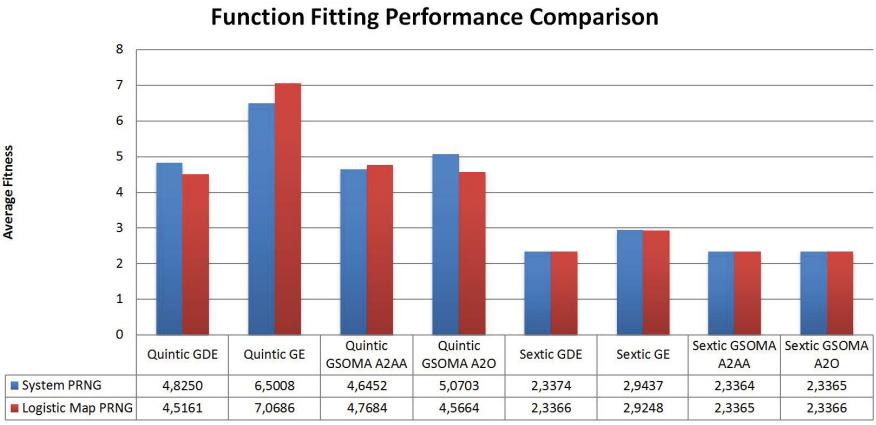


Fig. 1. An example of GSOMA

be seen in each of these diagrams, see Fig. 1. That is because each evolution has been run in 50 iterations as mentioned earlier and each chart visualizes these iterations all at once.

There were created diagrams for each evolutionary technique tested (GE, GDE and GSOMA) for each problem (Sextic, Quintic), see example at Fig. 2. That is obviously because two different problems were optimized. The diagrams are quite self-describing. There is a fitness value on the vertical axis and a number of fitness



	Quintic GDE	Quintic GE	Quintic GSOMA A2AA	Quintic GSOMA A2O	Sextic GDE	Sextic GE	Sextic GSOMA A2AA	Sextic GSOMA A2O
System PRNG	4,8250	6,5008	4,6452	5,0703	2,3374	2,9437	2,3364	2,3365
Logistic Map PRNG	4,5161	7,0686	4,7684	4,5664	2,3366	2,9248	2,3365	2,3366

Fig. 2. Function fitting performance comparison

function evaluations on the horizontal axis. The individual lines consequently represent how many times the fitness function had to be evaluated to find a solution having the particular fitness value.

The second type of diagram shows the overall performance of each evolutionary technique depending of the PRNG used, see example at Fig. 2. There is a serie for each evolution technique measured (each of them with both PRNGs) on the horizontal axis. The vertical axis shows the average fitness values of the best solution of each iteration that has been created using 20,000 fitness function evaluation at maximum.

4 Conclusion

Based on the presented results, the chaotic PRNG based on logistic map proved itself suitable for use with GE, GDE and both GSOMA variants. As it turned out, neither of the techniques is sensitive to the non-uniform numbers distribution of the logistic map PRNG. The difference in performance using either of generators is within statistical error. It is surprising that such simple dynamic system can produce non random numbers which allows evolutionary techniques to run with almost unaffected performance. The uniformity of distribution is of course not the only attribute of quality and logistic map. PRNG may have some other qualities that the system PRNG is missing, although it offers much better (i.e. more uniform) number distribution. It also does not seem that logistic map PRNG would work better with some evolutionary techniques that with the others. Results for GE, GDE and GSOMA are all very similar for both PRNG types. There are open research questions like what is the best combination of algorithm parameters in Master-Slave approach, etc. The solution of those and other questions is mater on the next research.

Acknowledgement. The following two grants are acknowledged for the financial support provided for this research: Grant Agency of the Czech Republic - GACR P103/13/08195S, partially supported by Grant of SGS No. SP2014/159, VŠB - Technical University of Ostrava, Czech Republic, and by European Regional Development Fund under the project CEBIA-Tech No. CZ.1.05/2.1.00/03.0089.

References

1. Koza, J.: Genetic Programming: A paradigm for genetically breeding populations of computer programs to solve problems, Stanford University, Computer Science Department, Technical Report, STAN-CS-90-1314 (1990)
2. Koza, J.: Genetic Programming. MIT Press (1998)
3. Ryan, C., Collins, J.J., Neill, M.O.: Grammatical Evolution: Evolving Programs for an Arbitrary Language. In: Banzhaf, W., Poli, R., Schoenauer, M., Fogarty, T.C. (eds.) EuroGP 1998. LNCS, vol. 1391, pp. 83–96. Springer, Heidelberg (1998)
4. Zelinka, I., Oplatkova, Z., Nolle, L.: Analytic programming – Symbolic regression by means of arbitrary evolutionary algorithms. Int. J. of Simulation, Systems, Science and Technology 6(9), 44–56 (2005)
5. Johnson, C.: Artificial immune systems programming for symbolic regression. In: Ryan, C., Soule, T., Keijzer, M., Tsang, E., Poliand, R., Costa, E. (eds.) EuroGP 2003. LNCS, vol. 2610, pp. 345–353. Springer, Heidelberg (2003)
6. Weisser, R., Osmera, P.: Two-Level Transplant Evolution for Optimization of General Controllers, New Trends in Technologies, Sciyo (2010)
7. Weisser, R., Osmera, P.: Two-level Tranpslant Evolution, 17th Zittau Fuzzy Colloquium, Zittau, Germany (2010)
8. Weisser, R., Osmera, P., Matousek, R.: Transplant Evolution with Modified Schema of Differential Evolution: Optimization Structure of Controllers. In: International Conference on Soft Computing MENDEL, Brno, Czech Republic (2010)
9. O'Neill, M., Brabazon, A.: Grammatical Differential Evolution. In: Proceedings of International Conference on Artificial Intelligence, pp. 231–236. CSEA Press (2006)
10. Koza, J., Bennet, F., Andre, D., Keane, M.: Genetic Programming III. Morgan Kaufmann, New York (1999)
11. Zelinka, I., Oplatkova, Z.: Analytic programming – Comparative study. In: Proceedings of Second International Conference on Computational Intelligence, Robotics, and Autonomous Systems, Singapore (2003)
12. Koza, J., Keane, M., Streeter, M.: Evolving inventions. Scientific American, 40–47 (2003)
13. Zelinka, I., Davendra, D., Senkerik, R., Jasek, R., Oplatkova, Z.: Analytical Programming - a Novel Approach for Evolutionary Synthesis of Symbolic Structures. In: Kita, E. (ed.) Evolutionary Algorithms, InTech (2011), `http://www.intechopen.com/books/evolutionary-algorithms/analytical-programming-a-novel-approach-for-evolutionary-synthesis-of-symbolic-structures` ISBN: 978-953-307-171-8, doi: 10.5772/16166
14. Zelinka, I., Senkerik, R., Pluhacek, M.: Do Evolutionary Algorithms Indeed Require Randomness? In: IEEE Congress on Evolutionary Computation, Cancun, Mexico, pp. 2283–2289 (2013)

15. Zelinka, I., Chadli, M., Davendra, D., Senkerik, R., Pluhacek, M., Lampinen, J.: Hidden Periodicity - Chaos Dependance on Numerical Precision. In: Zelinka, I., Chen, G., Rössler, O.E., Snasel, V., Abraham, A. (eds.) Nostradamus 2013: Prediction, Model. & Analysis. AISC, vol. 210, pp. 47–59. Springer, Heidelberg (2013)
16. Zelinka, I., Chadli, M., Davendra, D., Senkerik, R., Pluhacek, M., Lampinen, J.: Do evolutionary algorithms indeed require random numbers? Extended study. In: Zelinka, I., Chen, G., Rössler, O.E., Snasel, V., Abraham, A. (eds.) Nostradamus 2013: Prediction, Model. & Analysis. AISC, vol. 210, pp. 61–75. Springer, Heidelberg (2013)
17. Price, K.: An Introduction to Differential Evolution. In: Corne, D., Dorigo, M., Glover, F. (eds.) New Ideas in Optimization, pp. 79–108. McGraw-Hill, London (1999)
18. Zelinka, I.: SOMA – Self Organizing Migrating Algorithm. In: Babu, B.V., Onwubolu, G. (eds.) New Optimization Techniques in Engineering, pp. 167–218. Springer, New York (2004)
19. Oplatkova, Z., Zelinka, I.: Investigation on artificial ant using analytic programming. In: Proceedings of Genetic and Evolutionary Computation Conference, Seattle, WA, pp. 949–950 (2006)
20. O'Neill, M., Ryan, C.: Grammatical Evolution, Evolutionary Automatic Programming in an Arbitrary Language. Springer, New York (2003)
21. Beyer, H.-G.: Theory of Evolution Strategies. Springer, New York (2001)
22. Cern, V.: Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm. J. Opt. Theory Appl. 45(1), 41–51 (1985)
23. Kirkpatrick, S., Gelatt Jr., C.D., Vecchi, M.P.: Optimization by Simulated Annealing. Science 220(4598), 671–680 (1983)
24. Clerc, M.: Particle Swarm Optimization. ISTE Publishing Company (2006) ISBN 1905209045
25. Caponetto, R., Fortuna, L., Fazzino, S., Xibilia, M.: Chaotic sequences to improve the performance of evolutionary algorithms. IEEE Trans. Evol. Comput. 7(3), 289–304 (2003)
26. Pluhacek, M., Senkerik, R., Davendra, D., Kominkova Oplatkova, Z.: On the Behaviour and Performance of Chaos Driven PSO Algorithm with Inertia Weight. Computers and Mathematics with Applications (in print) ISSN 0898-1221
27. Pluhacek, M., Budikova, V., Senkerik, R., Oplatkova, Z., Zelinka, I.: Extended initial study on the performance of enhanced PSO algorithm with lozi chaotic map. In: Zelinka, I., Snasel, V., Rössler, O.E., Abraham, A., Corchado, E.S. (eds.) Nostradamus: Mod. Meth. of Prediction, Modeling. AISC, vol. 192, pp. 167–177. Springer, Heidelberg (2013)
28. Pluhacek, M., Senkerik, R., Zelinka, I.: Impact of Various Chaotic Maps on the Performance of Chaos Enhanced PSO Algorithm with Inertia Weight – an Initial Study. In: Zelinka, I., Snasel, V., Rössler, O.E., Abraham, A., Corchado, E.S. (eds.) Nostradamus: Mod. Meth. of Prediction, Modeling. AISC, vol. 192, pp. 153–166. Springer, Heidelberg (2013)
29. Persohn, K.J., Povinelli, R.J.: Analyzing logistic map pseudorandom number generators for periodicity induced by finite precision floating-point representation. Chaos, Solitons and Fractals 45, 238–245 (2012)
30. Drutarovsky, M., Galajda, P.: A robust chaos-based true random number generator embedded in reconfigurable switched-capacitor hardware. In: 17th International Conference Radioelektronika, April 24-25, vol. 1 and 2, pp. 29–34. Brno, Czech Republic (2007)

31. Bucolo, M., Caponetto, R., Fortuna, L., Frasca, M.,, R.: Does chaos work better than noise? IEEE Circuits and Systems Magazine 2(3), 4–19 (2002)
32. Hu, H., Liu, L., Ding, N.D.: Pseudorandom sequence generator based on the Chen chaotic system. Computer Physics Communications 184(3), 765–768 (2013), doi:10.1016/j.cpc.2012.11.017
33. Pluchino, A., Rapisarda, A., Tsallis, C.: Noise, synchrony, and correlations at the edge of chaos. Physical Review E 87(2) (2013), doi:10.1103/PhysRevE.87.022910
34. Lozi, R.: Emergence Of Randomness From Chaos. International Journal of Bifurcation and Chaos 22(2), 1250021 (2012), doi:10.1142/S0218127412500216
35. Wang, X.-Y., Qin, X.: A new pseudo-random number generator based on CML and chaotic iteration. Nonlinear Dynamics An International Journal of Nonlinear Dynamics and Chaos in Engineering Systems, Nonlinear Dyn. 70(2), 1589–1592 (2012), doi:10.1007/s11071-012-0558-0
36. Pareek, N.K., Patidar, V., Sud, K.K.: A Random Bit Generator Using Chaotic Maps. International Journal of Network Security 10(1), 32–38 (2010)
37. Wang, X.-Y., Yang, L.: Design Of Pseudo-Random Bit Generator Based On Chaotic Maps. International Journal of Modern Physics B 26(32), 1250208 (9 pages) (2012), doi:10.1142/S0217979212502086
38. Zhang, S.Y., Xingsheng, L.G.: A hybrid co-evolutionary cultural algorithm based on particle swarm optimization for solving global optimization problems. In: International Conference on Life System Modeling and Simulation / International Conference on Intelligent Computing for Sustainable Energy and Environment (LSMS-ICSEE), Wuxi, PR China, September 17-20 (2010)
39. Hong, W.-C., Dong, Y., Zhang, W.Y., Chen, L.-Y., Panigrahi, B.K.: Cyclic electric load forecasting by seasonal SVR with chaotic genetic algorithm. International Journal of Electrical Power and Energy Sysytems 44(1), 604–614, doi:10.1016/j.ijepes.2012.08.010
40. Chadli, M.: Unknown inputs observer design for fuzzy systems with application to chaotic system reconstruction. Computers and Mathematics with Applications 66(2), 147–154 (2013)
41. Zelinka, I., Chadli, M., Davendra, D., Senkerik, R., Jasek, R.: An investigation on evolutionary reconstruction of continuous chaotic systems. Mathematical and Computer Modelling 57(1-2), 2–15 (2013)