SYNTHESIS OF CONTROL LAW FOR CHAOTIC HENON SYSTEM PRELIMINARY STUDY

Zuzana Oplatkova, Roman Senkerik, Ivan Zelinka, Jiří Hološka
Faculty of Applied Informatics
Tomas Bata University in Zlin
Nad Stranemi 4511, 760 05 Zlin
Czech Republic
{Oplatkova, Senkerik, Zelinka}@fai.utb.cz

KEYWORDS
Control law, Henon system, synthesis, evolutionary computation, analytic programming.

ABSTRACT

The paper deals with a synthesis of control law for a discrete chaotic Henon system by means of analytic programming. This is a preliminary study in which the aim is to show that tool for symbolic regression – analytic programming - is possible to use for such kind of problems. The paper consists of description of analytic programming as well as chaotic Henon system. This article contents only 21 successful simulations in the result section and will be extended within future tests in this field. SOMA (Self-Organizing Migrating Algorithm) with analytic programming was used for experiments in this case.

INTRODUCTION

The interest about the control of chaotic systems is spread day by day. First steps were done in (Zelinka et al., 2006), (Zelinka et al., 2007), (Senkerik et al., 2006) where the control law was based on Pyragas method: Extended delay feedback control – ETDAS (Pyragas, 1995). That papers were concerned to tune several parameters inside the control technique for chaotic system. Compared to that, a presented paper shows a possibility how generate the whole control law (not only to optimize several parameters) for the purpose of stabilization of a chaotic system. The synthesis of control is inspired by the Pyragas's delayed feedback control technique (Just, 1999), (Pyragas, 1992). Unlike the original OGY control method (Ott, 1990) it can be simply considered as a targeting and stabilizing algorithm together in one package (Kwon, 1999). Another big advantage of Pyragas method is the amount of accessible control parameters.
Instead of evolutionary algorithms (EA) utilization, analytic programming (AP) is used here. AP is a superstructure of EAs and is used for synthesis of analytic solution according the required behavior. A control law from the proposed system can be viewed as a symbolic structure, which can be created according the requirements for the stabilization of a chaotic system. The advantage is that it is not necessary to have some "preliminary" control law and only to estimate its parameters. This system will generate the structure of the law also with suitable parameter values.
Firstly, a problem design is proposed. The next paragraph is focused on AP description. Results and conclusion follow afterwards.

PROBLEM DESIGN

The chosen example of chaotic system was the two dimensional Henon map in form (1).

$$\begin{aligned} x_{n+1} &= a - x_n^2 + by_n \\ y_{n+1} &= x_n \end{aligned} \tag{1}$$

This is a model invented with a mathematical motivation to investigate chaos. The Henon map is a discrete-time dynamical system, which was introduced as a simplified model of the Poincaré map for the Lorenz system. It is one of the most studied examples of dynamical systems that exhibit chaotic behavior and in fact it is also a two-dimensional extension of the one-dimensional quadratic map. The map depends on two parameters, a and b, which for the canonical Henon map have values of $a = 1.4$ and $b = 0.3$. For the canonical values the Henon map is chaotic (Hilborn, 2000).
The example of this chaotic behavior can be clearly seen from bifurcation diagram – Figure 1.
This figure shows the bifurcation diagram for the Henon map created by plotting of a variable x as a function of the one control parameter for the fixed second parameter.
This work is focused on explanation of AP application for synthesis of a whole control law instead of demanding tuning of EDTAS method control law to stabilize desired Unstable Periodic Orbits (UPO). As a study case a p-1 (a fixed point) desired UPO is used only in this preliminary study. Until today, 21 successful simulations out of 21 have been carried out and the others are running.
EDTAS method was obviously an inspiration for preparation of sets of basic functions and operators for AP.

Proceedings 24th European Conference on Modelling and Simulation ©ECMS Andrzej Bargiela, Sayed Azam Ali David Crowley, Eugène J.H. Kerckhoffs (Editors)
ISBN: 978-0-9564944-0-5 / ISBN: 978-0-9564944-1-2 (CD)

The original control method – ETDAS in the discrete form suitable for two-dimensional Henon map has the form (2).

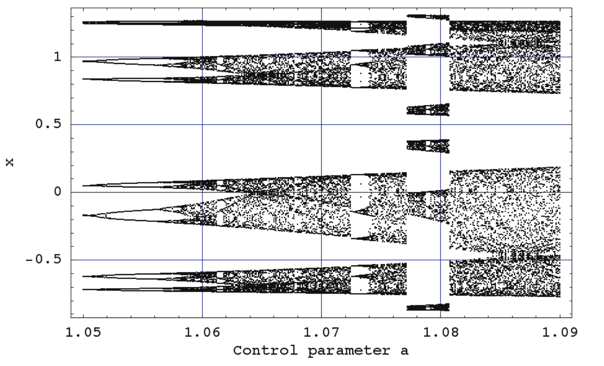


Figure 1: Bifurcation diagram of Henon Map

$$\begin{aligned} x_{n+1} &= a - x_n^2 + by_n + F_n \\ F_n &= K\left[(1-R)S_{n-m} - x_n\right] \\ S_n &= x_n + RS_{n-m} \end{aligned} \tag{2}$$

Where K and R are adjustable constants, F is the perturbation, S is given by a delay equation utilizing previous states of the system and m is the period of m-periodic orbit to be stabilized. The perturbation F_n in equations (2) may have arbitrarily large value, which can cause diverging of the system outside the interval {-1.5, 1.5}. Therefore, F_n should have a value between $-F_{\max}$, $F_{\max}$. In this preliminary study a suitable $F_{\max}$ value was taken from the previous research. To find the optimal value also for this parameter is in future plans.

COST FUNCTION FOR STABILIZATION TESTING

Proposal for the cost function comes from the simplest Cost Function (CF) presented in (Senkerik et al., 2008). The core of CF could be used only for the stabilization of p-1 orbit. The idea was to minimize the area created by the difference between the required state and the real system output on the whole simulation interval – τ_i.

But another cost function had to be used for stabilizing of higher periodic orbit. It was synthesized from the simple CF and other terms were added. In this case, it is not possible to use the simple rule of minimizing the area created by the difference between the required and actual state on the whole simulation interval – τ_i, due to the many serious reasons, for example: degrading of the possible best solution by phase shift of periodic orbit.

This CF is in general based on searching for desired stabilized periodic orbit and thereafter calculation of the difference between desired and found actual periodic orbit on the short time interval - τ_s (approx. 20 - 50 iterations) from the point, where the first min. value of difference between desired and actual system output is found. Such a design of CF should secure the successful stabilization of higher periodic orbit anywise phase shifted.

This CF can be also used for p-1 orbit. The CF_{Basic} has the form (3).

$$CF_{Basic} = penalization_1 + \sum_{t=\tau 1}^{\tau 2} \left| TS_t - AS_t \right| \tag{3}$$

where: TS - target state, AS - actual state

τ_1 - the first min. value of difference between TS and AS

τ_2 – the end of optimization interval (τ_1+ τ_s)

penalization$_1$= 0 if τ_i - $\tau_2 \geq \tau_s$;

penalization$_1$= 10*(τ_i - τ_2) if τ_i - τ_2 < τ_s (i.e. late stabilization)

ANALYTIC PROGRAMMING

Basic principles of the AP were developed in 2001 (Zelinka et al., 2005), (Zelinka et al., 2008), (Oplatkova et al., 2009). Until that time only genetic programming (GP) and grammatical evolution (GE) had existed. GP uses genetic algorithms while AP can be used with any evolutionary algorithm, independently on individual representation. To avoid any confusion, based on use of names according to the used algorithm, the name - Analytic Programming was chosen, since AP represents synthesis of analytical solution by means of evolutionary algorithms.

The core of AP is based on a special set of mathematical objects and operations. The set of mathematical objects is set of functions, operators and so-called terminals (as well as in GP), which are usually constants or independent variables. This set of variables is usually mixed together and consists of functions with different number of arguments. Because of a variability of the content of this set, it is called here "general functional set" – GFS. The structure of GFS is created by subsets of functions according to the number of their arguments. For example GFS_{all} is a set of all functions, operators and terminals, GFS_{3arg} is a subset containing functions with only three arguments, GFS_{0arg} represents only terminals, etc. The subset structure presence in GFS is vitally important for AP. It is used to avoid synthesis of pathological programs, i.e. programs containing functions without arguments, etc. The content of GFS is dependent only on the user. Various functions and terminals can be mixed together (Zelinka et al., 2005), (Zelinka et al., 2008), (Oplatkova et al., 2009).

The second part of the AP core is a sequence of mathematical operations, which are used for the program synthesis. These operations are used to transform an individual of a population into a suitable program. Mathematically stated, it is a mapping from an individual domain into a program domain. This mapping consists of two main parts. The first part is called discrete set handling (DSH) (Figure 2) (Zelinka et al., 2005), (Lampinen & Zelinka, 1999) and the second one stands for security procedures which do not allow synthesizing pathological programs. The method of DSH, when used, allows handling arbitrary objects including nonnumeric objects like linguistic terms {hot, cold, dark…}, logic terms (True, False) or other user defined functions. In the AP DSH is used to map an individual into GFS and together with security procedures creates the above mentioned mapping which transforms arbitrary individual into a program.

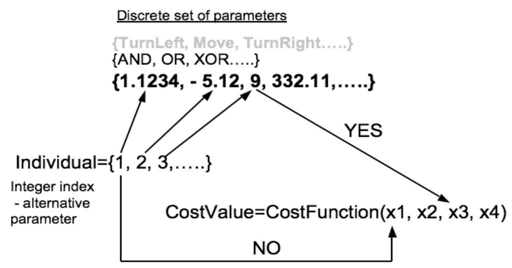


Figure 2: Discrete set handling

AP needs some evolutionary algorithm (Zelinka, 2004) that consists of population of individuals for its run. Individuals in the population consist of integer parameters, i.e. an individual is an integer index pointing into GFS. The creation of the program can be schematically observed in Figure 3. The individual contains numbers which are indices into GFS. The detailed description is represented in (Zelinka et al., 2005), (Zelinka et al., 2008), (Oplatkova et al., 2009).

AP exists in 3 versions – basic without constant estimation, AP_{nf} – estimation by means of nonlinear fitting package in Mathematica environment and AP_{meta} – constant estimation by means of another evolutionary algorithms; meta means metaevolution.

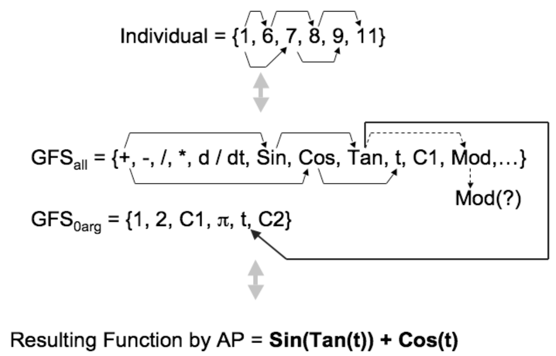


Figure 3: Main principles of AP

USED EVOLUTIONARY ALGORITHMS

Self Organizing Migrating Algorithm (SOMA) is a stochastic optimization algorithm that is modelled on the social behaviour of cooperating individuals (Zelinka, 2004). It was chosen because it has been proven that the algorithm has the ability to converge towards the global optimum (Zelinka, 2004). SOMA works on a population of candidate solutions in loops called *migration loops*. The population is initialized randomly distributed over the search space at the beginning of the search. In each loop, the population is evaluated and the solution with the highest fitness becomes the leader *L*. Apart from the leader, in one migration loop, all individuals will traverse the input space in the direction of the leader. Mutation, the random perturbation of individuals, is an important operation for evolutionary strategies (ES). It ensures the diversity amongst the individuals and it also provides the means to restore lost information in a population. Mutation is different in SOMA compared with other ES strategies. SOMA uses a parameter called PRT to achieve perturbation. This parameter has the same effect for SOMA as mutation has for genetic algorithms.

The novelty of this approach is that the PRT Vector is created before an individual starts its journey over the search space. The PRT Vector defines the final movement of an active individual in search space.

The randomly generated binary perturbation vector controls the allowed dimensions for an individual. If an element of the perturbation vector is set to zero, then the individual is not allowed to change its position in the corresponding dimension.

An individual will travel a certain distance (called the PathLength) towards the leader in *n* steps of defined length. If the PathLength is chosen to be greater than one, then the individual will overshoot the leader. This path is perturbed randomly.

RESULTS

As above mentioned, AP needs an evolutionary algorithm for its run. In this paper AP_{meta} version was used. It was easier to set all parameters than to use nonlinear fitting package, which was used with a big success in other cases.

SOMA algorithm (Zelinka, 2004) was used for both optimization tasks – to find a suitable solution of the control law and in metaevolution - to find suitable estimated values of constants in the obtained control law. Both settings were similar (Table 1 and Table 2).

Table 1: SOMA settings for AP

PathLength	3
Step	0.11
PRT	0.1
PopSize	50
Migrations	4
Max. CF Evaluations (CFE)	5345

Table 2: SOMA settings for meta evolution

PathLength	3
Step	0.11
PRT	0.1
PopSize	40
Migrations	5
Max. CF Evaluations (CFE)	5318

During all simulations, 21 successful results were obtained. A minimum number of cost function evaluations in the case of SOMA for AP was 100, maximum 3093. The average through all simulations was 1345. Simulations were stopped when CF was under 10^{-8}. These numbers were only for AP, if also second evolution for obtaining parameter values would be taken, each number has to be multiplied by 5318, i.e. total number of cost function evaluations was from 0.532 millions to 16.449 millions.

As was said the novelty of this approach represents the synthesis of feedback control law F_n (4) (perturbation) for the Henon system inspired by original ETDAS control method.

$$x_{n+1} = a - x_n^2 + by_n + F_n \tag{4}$$

Following control laws are examples of obtained results in version without Ks estimated from AP and the notation with simplification after estimation by means of second SOMA. The first case was stored for further processing or better tuning.

a) without estimation

$$F_n = \frac{x_{n-1}x_n}{x_{n-1}(2x_n - x_{n-1}) - \dfrac{x_n + K_1}{x_{n-1} - x_n}}$$

with Ks estimation

$$F_n = \frac{x_{n-1}x_n}{x_{n-1}(2x_n - x_{n-1}) - \dfrac{x_n + 0.0349}{x_{n-1} - x_n}}$$

In this case, number 2 inside is not supposed as some K but simplification of original formula $x_{n-1} + x_{n-1}$
Stabilization was reached in 33[th] step.

b) without estimation

$$F_n = \frac{x_n(x_n - x_{n-1})}{K_1}$$

with Ks estimation

$$F_n = 0.9203 * x_n(x_n - x_{n-1})$$

The system was stabilized in 25[th] step.

c) without estimation

$$F_n = K_1(x_n - x_{n-1})$$

with Ks estimation

$$F_n = 0.76899 * (x_n - x_{n-1})$$

Stabilization was reached in minimal number of steps (from all simulations) – in 20[th] step. Simulation output of the stabilization is depicted in Figure 4.

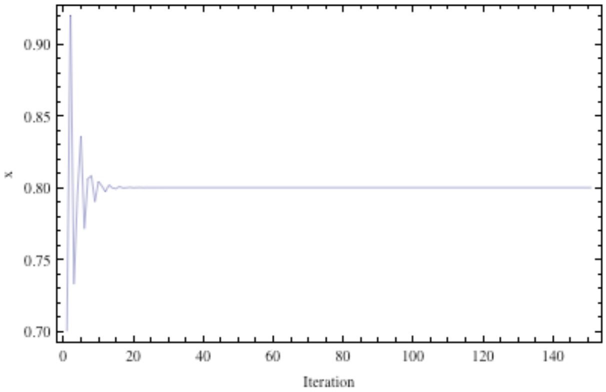


Figure 4: Example of result – stabilization of chaotic system by means of control law given in c)

d) without estimation

$$F_n = \frac{(x_{n-1} - x_n)K_1(-\dfrac{x_{n-1}}{K_2} - x_{n-1}x_n)}{x_n}$$

with Ks estimation

$$F_n = \frac{1.111(x_{n-1} - x_n)(0.011x_{n-1} - x_{n-1}x_n)}{x_n}$$

Stabilization was reached in maximal number of steps (from all simulations) - in 47[th] step. Simulation output of the stabilization is depicted in Figure 5.

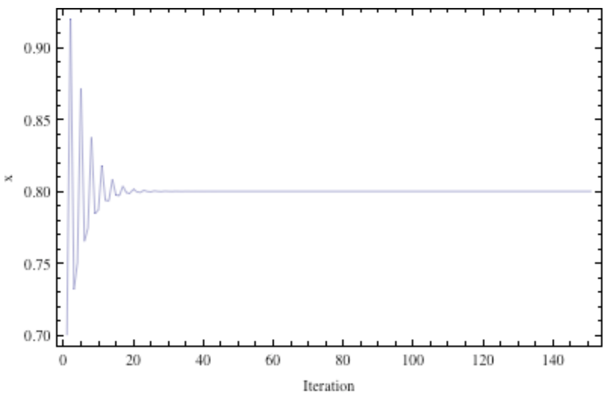


Figure 5: Example of result – stabilization of chaotic system by means of control law given in d)

e) without estimation

$$F_n = \frac{(x_{n-1} - x_n)}{x_{n-1}(x_n + K_1)}$$

with Ks estimation

$$F_n = \frac{(x_{n-1} - x_n)}{x_{n-1}(x_n - 2.2856)}$$

Stabilization was reached in maximal number of steps (from all simulations) - in 41th step. Simulation output of the stabilization is depicted in Figure 6.

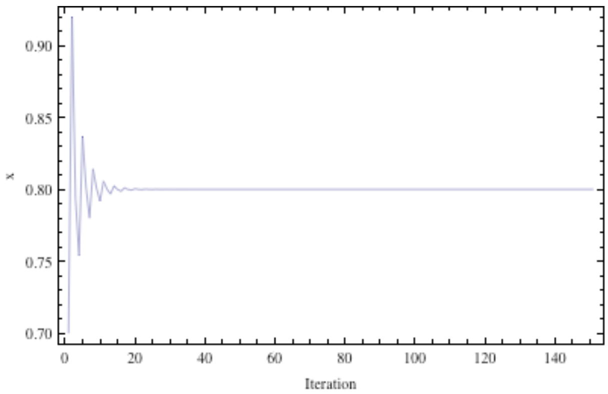


Figure 6: Example of result – stabilization of chaotic system by means of control law given in e)

Also an original ETDAS control law, which was tuned by means of evolutionary algorithms within (Senkerik et al., 2008) was found in the case of AP.
The quality of the solution was more or less the same. The cost function value is in order -15 or -16, i.e. almost zero. Thus the stabilization according to the cost value was very precise. The reached steps were counted as firstly observed required number rounded to 10^{-6}. According to this rule, the fastest stabilization was observed within 20 steps. The average was 34 steps.
The reached stabilization can be viewed also from the other side as a first minimal difference between reached and required solution. For the case with 20 steps it was reached 92 steps for successful stabilization according the second rule. The average amount of steps was 104.

CONCLUSION

This paper deals with a synthesis of a control law for stabilization of chaotic Henon system. In this case the analytic programming was used instead of tuning of parameters by means of evolutionary algorithms as in the previous research. AP is able to synthesize a symbolic notation for required behaviour of the system, EA tunes only parameters of expected law borrowed from ETDAS.
The presented results show that AP is able to solve problems of this kind and to produce the control law in a symbolic way. Within this preliminary study SOMA algorithm was used as an optimization algorithm for AP and also for estimating parameters in the second evolutionary process (meta-evolutionary approach).
Future plans are concerned to further tests to obtain more results for this chaotic system to reach a better statistics and also to use other chaotic systems. Further simulations will be considered also for higher orbit stabilization.

ACKNOWLEDGMENT

This work was supported by the grant NO. MSM 7088352101 of the Ministry of Education of the Czech Republic and by grants of Grant Agency of Czech Republic GACR 102/09/1680

REFERENCES

Hilborn R.C., 2000. Chaos and Nonlinear Dynamics: An Introduction for Scientists and Engineers, Oxford University Press, 2000, ISBN: 0-19-850723-2.

Just W., 1999, “Principles of Time Delayed Feedback Control”, In: Schuster H.G., Handbook of Chaos Control, Wiley-Vch, ISBN 3-527-29436-8.

Kwon O. J., 1999. “Targeting and Stabilizing Chaotic Trajectories in the Standard Map”, Physics Letters A. vol. 258, 1999, pp. 229-236.

Lampinen J., Zelinka I., 1999. *New Ideas in Optimization – Mechanical Engineering Design Optimization by Differential Devolution*, Volume 1. London: McGraw-hill, 1999, 20 p., ISBN 007-709506-5.

Oplatková, Z., Zelinka, I.: 2009. Investigation on Evolutionary Synthesis of Movement Commands, Modelling and Simulation in Engineering, Volume 2009 (2009), Article ID 845080, 12 pages, Hindawi Publishing Corporation, ISSN: 1687-559, e-ISSN: 1687-5605, doi:10.1155/2009/845080.

Ott E., C. Greboki, J.A. Yorke, 1990. “Controlling Chaos”, Phys. Rev. Lett. vol. 64, 1990, pp. 1196-1199.

Pyragas K., 1992, “Continuous control of chaos by self-controlling feedback”, Physics Letters A, 170, 421-428.

Pyragas K., 1995. “Control of chaos via extended delay feedback”, Physics Letters A, vol. 206, 1995, pp. 323-330.

Senkerik R., Zelinka I., Navratil E., 2006. “Optimization of feedback control of chaos by evolutionary algorithms”, in proc 1st IFAC Conference on analysis and control of chaotic systems, Reims, France, 2006.

Senkerik R., Zelinka I., Oplatkova Z., 2008. Evolutionary Techniques for Deterministic Chaos Control, CISSE’08, In Proc. IETA 2008, International Joint Conferences on Computer, Information, and Systems Sciences, and Engineering, 5-13 December 2008, ISBN 978-90-481-3655-1.

Zelinka I., 2004. “SOMA – Self Organizing Migrating Algorithm”, In: *New Optimization Techniques in Engineering*, (B.V. Babu, G. Onwubolu (eds)), chapter 7, 33, Springer-Verlag, 2004, ISBN 3-540-20167X.

Zelinka I.,Oplatkova Z, Nolle L., 2005. *Boolean Symmetry Function Synthesis by Means of Arbitrary Evolutionary Algorithms-Comparative Study*, International Journal of Simulation Systems, Science and Technology, Volume 6, Number 9, August 2005, pages 44 - 56, ISSN: 1473-8031, online http://ducati.doc.ntu.ac.uk/uksim/journal/Vol-6/No.9/cover.htm, ISSN: 1473-804x.

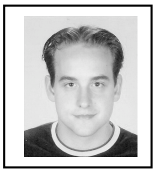
Zelinka I., Senkerik R., Navratil E., 2006. “Investigation on Real Time Deterministic Chaos Control by Means of Evolutionary Algorithms”, CHAOS’06, In Proc. 1st IFAC Conference on Analysis and Control of Chaotic Systems, Reims, France, 28-30 June 2006, pages 211-217

Zelinka I., Senkerik R., Navratil E., 2007. ”Investigation on Evolutionary Optimitazion of Chaos Control”, CHAOS, SOLITONS & FRACTALS (2007), doi:10.1016/j.chaos.2007.07.045.

Zelinka, I., Guanrong Ch., Celikovsky S., 2008. Chaos Synthesis by Means of Evolutionary algorithms, International Journal of Bifurcation and Chaos, Vol. 18, No. 4 (2008) 911–942.

AUTHORS BIOGRAPHIES

ZUZANA OPLATKOVA was born in Czech Republic, and went to the Tomas Bata University in Zlin, where she studied technical cybernetics and obtained her MSc. degree in 2003 and Ph.D. degree in 2008. She is a lecturer (Artificial Intelligence) at the same university. Her e-mail address is: oplatkova@fai.utb.cz



ROMAN SENKERIK was born in the Czech Republic, and went to the Tomas Bata University in Zlin, where he studied Technical Cybernetics and obtained his MSc degree in 2004 and Ph.D. degree in Technical Cybernetics in 2008. He is now a lecturer at the same university (Applied Informatics, Cryptology, Artificial Intelligence, Mathematical Informatics). Email address: senkerik@fai.utb.cz

IVAN ZELINKA was born in the Czech Republic, and went to the Technical University of Brno, where he studied Technical Cybernetics and obtained his degree in 1995. He obtained Ph.D. degree in Technical Cybernetics in 2001 at Tomas Bata University in Zlin. Now he is a professor (Artificial Intelligence, Theory of Information) and a head of the department. Email address: zelinka@fai.utb.cz .