Journal of Global Optimization (2006) 35: 145–160
DOI 10.1007/s10898-005-1509-9

A Verified Optimization Technique to Locate Chaotic Regions of Hénon Systems

TIBOR CSENDES[1], BARNABÁS M. GARAY[2] and BALÁZS BÁNHELYI[1]
[1]*Institute of Informatics, University of Szeged, H-6701 Szeged, P.O. Box 652, Hungary (e-mails:{csendes, banhelyi}@inf.u-szeged.hu)*
[2]*Institute of Mathematics, Budapest University of Technology, H-1521 Budapest, Hungary (e-mail: garay@math.bme.hu)*

(Received 30 May 2005; accepted 1 August 2005)

Abstract. We present a new verified optimization method to find regions for Hénon systems where the conditions of chaotic behaviour hold. The present paper provides a methodology to verify chaos for certain mappings and regions. We discuss first how to check the set theoretical conditions of a respective theorem in a reliable way by computer programs. Then we introduce optimization problems that provide a model to locate chaotic regions. We prove the correctness of the underlying checking algorithms and the optimization model. We have verified an earlier published chaotic region, and we also give new chaotic places located by the new technique.

Key words: chaos, global optimization, Hénon-map, verified optimization method.

1. Introduction

Computer-assisted proofs for the existence of chaos are important for the understanding of dynamic properties of the solutions of differential equations. These techniques have been intensively investigated recently (see e.g. Neumaier and Rage, 1993; Rage et al., 1994; Zgliczynski, 1997, 2003; Galias and Zgliczynski, 2001). A detailed summary on the set oriented (Lipschitz constant based) numerical methods for dynamical systems is given in Dellnitz and Junge (2002).

We study verified computational methods to check and locate regions the points of which fulfill the conditions of chaotic behaviour. The investigated Hénon mapping is $\mathcal{H}(x, y) = (1 + y - Ax^2, Bx)$. The paper (Zgliczynski, 1997) considered the $A = 1.4$ and $B = 0.3$ values and some regions of the two-dimensional Euclidean space: $E = E_1 \cup E_2 = \{(x, y) | x \geqslant 0.4, y \geqslant 0.28\} \cup \{(x, y) | x \leqslant 0.64, |y| \leqslant 0.01\}$, $\mathcal{O}_1 = \{(x, y) | x < 0.4, y > 0.01\}$, $\mathcal{O}_2 = \{(x, y) | y < 0\}$.

According to Zgliczynski (1997) Theorem 1 ensures the chaotic behaviour for the points of the parallelograms Q_0 and Q_1 with parallel sides with the x axis (for $y_0 = 0.01$ and $y_1 = 0.28$, respectively), with the common tangent of 2, and x coordinates of the lower vertices are $x_a = 0.460$, $x_b = 0.556$; and $x_c = 0.558$, $x_d = 0.620$, respectively. The mapping and

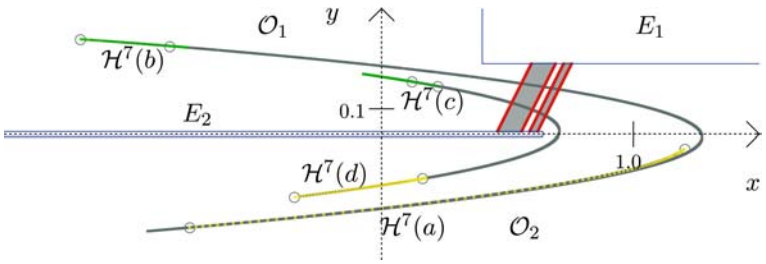

Figure 1. Illustration of the H^7 transformation for the classic Hénon parameters $A = 1.4$ and $B = 0.3$ together with the chaotic region of two parallelograms. The a, b, c, and d sides of the parallelograms are depicted on the upper left picture of Figure 2.

the problem details (such as the transformed sides of the parallelograms, $H^7(a), H^7(b), H^7(c)$, and $H^7(d)$) are illustrated on Figure 1.

THEOREM 1. *Assume that the following relations hold for the given particular Hénon mapping:*

$$\mathcal{H}^7(a \cup d) \subset \mathcal{O}_2, \tag{1}$$

$$\mathcal{H}^7(b \cup c) \subset \mathcal{O}_1, \tag{2}$$

$$\mathcal{H}^7(Q_0 \cup Q_1) \subset \mathbb{R}^2 \backslash E, \tag{3}$$

then chaotic trajectories belong to the starting points of the regions Q_0 and Q_1.

The present paper provides a methodology to verify chaos for certain mappings and regions. We discuss first how to check the set theoretical conditions of the above theorem in a reliable way by computer programs. Then we introduce optimization problems that provide a model to locate chaotic regions. We have checked the correctness of the earlier published chaotic region, the correctness of the underlying checking algorithms and the optimization model is proven. We also give new chaotic places located by the new technique. The paper (Bánhelyi et al., 2005 submitted) provides additional new chaotic regions located by the present method.

2. New Techniques for Checking the Chaotic Behaviour and for Locating Regions Containing such Points

2.1. A NEW ALGORITHM TO CHECK SUBSET RELATIONS

The main difficulty of checking conditions (1)–(3) is that one has to prove these for a continuum of points. In Zgliczynski (1997) the author calculated the Lipschitz constant, gave an upper bound for the rounding error committed and thus reduced the whole task to investigating a finite number

of points of a dense enough grid. This method works basically only with human interaction. To search chaotic regions an automated checking routine is more appropriate. The technique to be introduced combines interval arithmetic and adaptive branch-and-bound subdivision of the region of interest.

The applied algorithm first encloses the sets Q_0 and Q_1 in a two-dimensional closed interval I, the starting interval. Then to prove subset relations an adaptive branch-and-bound technique generates such a subdivision of the starting interval that either:

- for all subintervals the given conditions of chaos hold – in case they contain points of the respective sets, or
- it is shown that a small subinterval (of a user set size) exists that contains at least one point of the respective set, and it contradicts at least one of the relations.

Now the sets $\mathcal{O}_1, \mathcal{O}_2$, and $R^2 \backslash E$ in the conditions (1)–(3) are all open sets, and the union of a finite number of closed sets is closed as well. It is why the algorithm should check whether the transformed subintervals are subsets of the respective sets. For the following we assume that the conditions of chaos are expressed in a similar form, i.e. the transformed sets must be subsets of certain open sets.

To realize this algorithm we have applied a modified version of an earlier procedure to solve constrained nonlinear optimization, problems with tolerance (Kristinsdottir et al., 1993, 1996; Csendes et al., 1995).

The present Algorithm 1 is capable to recognize that a region satisfies the conditions of chaos in the sense given in Theorem 2. Denote the respective mapping by $\mathcal{T}$, the argument interval to be checked by Q, and the set in which the transformed set must be contained by $\mathcal{O}: \mathcal{T}(Q) \subset \mathcal{O}$.

The original mapping, $\mathcal{T}$ can be applied directly only to points. To build an adaptive subdivision algorithm we need a representation that enables us to describe a transformed set (in contrast to point to point mappings). The new technique uses an interval arithmetic based inclusion function (Ratschek and Rokne, 1988) for this purpose.

A mapping $F: \mathbb{I}^n \to \mathbb{I}^m$ is an *inclusion mapping* of the mapping $f: \mathbb{R}^n \to \mathbb{R}^m$ if for $\forall Y \in \mathbb{I}^n$ and $\forall y \in Y \;\; f(y) \in F(Y)$, where $\mathbb{I}$ stands for the set of all closed real intervals.

The lower and upper bounds of an interval $Y \in \mathbb{I}^n$ are denoted by $\underline{Y}$ and $\overline{Y}$, respectively. The width of an interval is $w(Y) = \overline{Y} - \underline{Y}$ if $Y \in \mathbb{I}$, and $w(Y) = \max_i(\overline{Y}_i - \underline{Y}_i)$ if $Y \in \mathbb{I}^n$ is an n-dimensional interval vector (also called a box). $\mathbb{I}(X)$ stands for all intervals in X. F is said to be an *isotone inclusion mapping* over X if for $\forall Y, Z \in \mathbb{I}(X), Y \subseteq Z$ implies $F(Y) \subseteq F(Z)$.

ALGORITHM 1. The checking routine

Inputs: – ε: the user set limit size of subintervals,
– Q: the argument set to be proved,
– $\mathcal{O}$: the aimed set for which $\mathcal{T}(\mathrm{Q}) \subset \mathcal{O}$ is to be checked.

1. Calculate the initial interval I, that contains the regions of interest
2. Push the initial interval into the stack
3. **while** the stack is nonempty
4. Pop an interval v out of the stack
5. Calculate the width of v
6. Determine the widest coordinate direction
7. Calculate the transformed interval $w = T(v)$
8. **if** $v \cap Q \neq \emptyset$, and the condition $w \subset \mathcal{O}$ does not hold, **then**
9. **if** the width of interval v is less than ε **then**
10. **print** that the condition is hurt by v and **stop**
11. **else** bisect v along the widest side: $v = v_1 \cup v_2$
12. push the subintervals into the stack
13. **endif**
14. **endif**
15. **end while**
16. **print** that the condition is proven and **stop**

We say that the inclusion mapping F has the *zero convergence property*, if $w(F(Z_i)) \to 0$ holds for all the $\{Z_i\}$ interval sequences for which $Z_i \subseteq X$ for all $i = 1, 2, \ldots$ and $w(Z_i) \to 0$.

One can easily compose an inclusion mapping by substituting the real operations and standard functions by their interval analogues. This procedure is called natural interval extension. The inclusion mapping provided by natural interval extension is an isotone inclusion mapping that has the zero convergence property (Ratschek and Rokne, 1988). We assume for the following that either the expression of the mapping is known, or a computer program is given that calculates it. To enclose the rounding errors, and to provide verified numerical results it suffices then to use the so called outside rounding that gives computer representable result intervals containing all the points of the real operations.

In rare cases it may happen that one of the transformed subintervals fits to the border of the aimed open set, as e.g. the interval $T(I) = [0.0, 0.4] \times [0.01, 0.02]$ fits $\mathcal{O}_1$. A careful implementation with the compulsory outside rounding is needed for the distance of points or sets that enables to recognize that $T(I)$ has a zero measure set of points that do not meet the condition $T(I) \subset \mathcal{O}_1$.

THEOREM 2. *Assume that the underlying mapping $\mathcal{T}$ is given by an inclusion mapping T and the algorithm returns that the checked condition $\mathcal{T}(Q) \subset$*

O is fulfilled. Then Algorithm 1 *generates a subdivision of the initial interval I around the search region in such a way that for all subintervals either*

(i) *the subinterval does not contain a point of the argument region Q, or*
(ii) *the transformed subinterval is a subset of the respective set of the condition O.*

Proof. In case the algorithm returned that the checked condition was fulfilled, it must have stopped in Step 16. It means that the starting interval I has possibly been subdivided into subintervals that covered I, nevertheless for none of these subintervals was the condition hurt (since the algorithm has not terminated in Step 10).

On the other hand Step 16 can only be reached if all the subintervals generated during the execution of Algorithm 1 were either further subdivided in Step 11, or skipped from further subdivision since the condition in Step 8 was not fulfilled.

Now since Step 16 was reached, the starting interval I was fully covered by subintervals $\{I_k\}_{k=1}^{n}$ that fit the above mentioned cases (i) and (ii). The inclusion property of mapping implies that then the original condition

$$\mathcal{T}(Q) \subset \bigcup_{l=1}^{m} T(I_l) \subset \mathcal{O}$$

holds, where $\cup_{l=1}^{m} T(I_l)$ is the union of those transformed subintervals that no $I_j \in \{(I_l)\}_{l=1}^{m}$ is disjunct of Q. In this sense Algorithm 1 is correct. □

After proving the correctness the following theorem discusses which property of the inclusion mapping is required to ensure that Algorithm 1 can recognize that a condition of chaotic behaviour is fulfilled.

THEOREM 3. *Assume that the underlying mapping $\mathcal{T}$ is given by an inclusion function T that has the zero convergence property, $\varepsilon = 0$, and $\mathcal{T}(Q) \subset \mathcal{O}$ holds. Then Algorithm* 1 *concludes after a finite number of iteration steps that the condition of chaotic behaviour is fulfilled.*

Proof. Assume that the statement of the theorem is false. Then there exist an infinite number of subintervals generated by the algorithm. Select an arbitrary infinite subsequence $\{(I_i)\}_{i=1}^{\infty}$ of these subintervals that converges to a point: $\lim_{i=1}^{\infty}(I_i) = x$. It is obviously possible by compactness. Two cases must be distinguished.

1. Consider first the case when the sequence of subintervals converges to such a point x, that $x \in Q$ holds. Due to the zero convergence property of the inclusion mapping $\lim_{i=1}^{\infty} w(T(I_i)) = 0$, i.e. $\lim_{i=1}^{\infty} T(I_i) = \mathcal{T}(x)$. We have assumed that $\mathcal{T}(Q) \subset \mathcal{O}$ holds, hence $\mathcal{T}(x) \in \mathcal{O}$. This fact and

that $\mathcal{O}$ is an open set, imply that $T(I_i) \subset \mathcal{O}$ holds for all i indices that are larger than a certain threshold index K. For the latter subintervals, however, Algorithm 1 does not make further subdivision, and it contradicts that the subinterval sequence is infinite as assumed.

2. Now when $\lim_{i=1}^{\infty} I_i = x \notin Q$, then, again we find that in Step 8 of Algorithm 1 the condition of further subdivision cannot be fulfilled, since after a certain iteration index L for all subintervals I_i, $I_i \not\subset Q$ holds. This fact however implies that the condition of subdivision cannot be satisfied for $i > L$. In other words, Algorithm 1 cannot generate an infinite subinterval sequence that converges to a point outside Q.

Now since $\varepsilon = 0$, thus Algorithm 1 cannot terminate in Step 10, only in Step 16. This concludes the proof. □

Consider now the case when the algorithm has to prove that the studied region does not meet the criteria of chaos.

THEOREM 4. *Assume that the underlying mapping $\mathcal{T}$ is given by an inclusion function T, $\varepsilon = 0$, and there exists a point $x \in Q$ such that $\mathcal{T}(x) \notin \mathcal{O}$. Then Algorithm* 1 *cannot conclude after a finite number of iteration steps whether the condition of chaotic behaviour is fulfilled.*

Proof. Consider the subintervals generated by Algorithm 1 that contain the point x for which $\mathcal{T}(x) \notin \mathcal{O}$ holds. For such an I subinterval $T(I)$ must contain $\mathcal{T}(x)$, and in this way $T(I)$ cannot be a subset of $\mathcal{O}$.

It is why at least one subinterval containing x always remains in the stack, they cannot be skipped in Step 14 (as they do fit to the **then** branch of Step 8). On the other hand these subintervals can neither be deleted in Step 10, since the widths of generated subintervals are positive. □

In other words, the main conclusion of Theorem 4 is that although Algorithm 1 can prove the chaotic behaviour of a mapping in a finite number of iteration steps, to prove that the respective conditions do not hold the algorithm is less suitable. When ε is set to zero, then the algorithm will not terminate. When we set ε to a positive value the user will obtain a result interval of width just below ε, for which the conditions of chaos could not be proven (although they may hold).

The checking routine has also been successfully used for a different chaos related problem (Csendes et al., 2005, submitted).

2.2. A GLOBAL OPTIMIZATION MODEL FOR LOCATING CHAOTIC REGIONS

Once we have a reliable computer procedure to check the conditions of chaotic behaviour of a mapping it is straightforward to set up an

optimization model that transforms the original chaos location problem to a global optimization problem.

The chaotic regions have several parameters that identify them. In the early phase of our investigation we have restricted the search to locate two parallelograms similar to that used in the article (Zgliczynski, 1997): we have allowed to change the vertical and horizontal positions and also the common tangent, but the parallelograms always had two sides parallel to the x axis. It is also possible to find fitting parameter values for the Hénon mapping, i.e. for the mapping parameters A and B, and furthermore also for parameters of the aimed sets of the underlying theorem, e.g. the border coordinates of the set E.

The search for a chaotic region was modelled as a constrained global optimization problem, subsequently the constraints were represented by a penalty function approach. The original objective function was constant, still the possibility exists to extend it to a more complex form that expresses further aims, e.g. to locate a second chaotic region, different from the known one.

The key question for the successful application of a global optimization algorithm is how to compose the penalty functions. On the basis of earlier experiences collected solving similar constrained problems, we have decided to add a nonnegative value proportional to how much the given condition was hurt, plus a fixed penalty term in case at least one of the constraints was not satisfied.

As an example, consider the case when one of the conditions for the transformed region was hurt, e.g. when (2), i.e. the relation

$$\mathcal{H}^k(b \cup c) \subset \mathcal{O}_1$$

does not hold for a given kth iterate, and for a region of two parallelograms. For such a case the checking routine will provide a subinterval I that contains at least one point of the investigated region, and which contradicts the given condition. Then we have calculated the Hausdorff distance of the transformed subinterval $H^k(I)$ to the set $\mathcal{O}_1$ of the right side of the condition,

$$\max_{x \in H^k(I)} \inf_{y \in \mathcal{O}_1} d(z, y),$$

where $d(z, y)$ is a given metric, a distance between two two-dimensional points. Notice that the use of maximum in the expression is crucial, with minimization instead our optimization approach could provide (and has provided) result regions that do not fulfill the given conditions of chaotic behaviour. On the other hand, the minimal distance according to points of the aimed set (this time $\mathcal{O}_1$) is satisfactory, since it enables the technique to

push the search into proper directions. In cases when the checking routine answered that the investigated subinterval has fulfilled the given condition, we have not changed the objective function.

Summing it up, we have considered the following bound constrained problem for the T inclusion function of the mapping $\mathcal{T}$:

$$\min_{x \in X} g(x), \tag{4}$$

where

$$g(x) = f(x) + p\left(\sum_{i=1}^{m} \max_{z \in T(I)} \inf_{y \in S_i} d(z, y)\right),$$

X is the n-dimensional interval of admissible values for the parameters x to be optimized, $f(x)$ the original, nonnegative objective function, and $p(y) = y + C$ if y is positive, and $p(y) = 0$ otherwise. C is a positive constant, larger than $f(x)$ for all the feasible x points, m is the number of conditions to be fulfilled, and S_i is the aimed set for the ith condition.

In this discussion I is the subinterval returned by the checking routine in Step 10 (or the empty set). The interval I depends implicitly on the parameter x to be optimized. At this point we do not have any assumption on the order of the subintervals handled by the checking routine. It may be advantageous to set the priority of the stack in such a way that we obtain that infeasible subinterval that has the largest distance to the aimed set. The function $f(x)$ can be used to find parameters implying certain properties. In our present numerical study we have set $f(x) = 0$, i.e. we simply aimed to prove subset relations according to Theorem 1. Notice also that when $f(x)$ is constant then the problem (4) is basically a constraint satisfaction problem (Neumaier, 2004).

For more complicated cases the fixed sets given in Theorem 1 should also be changed subject to certain structural constraints, e.g. the x_a, x_b, x_c, and x_d coordinates of the parallelograms have to follow this order. These new conditions can also be represented in a similar way, following the penalty function approach of (4).

We have followed a stepwise approach, i.e. first we have formulated a simple, low-dimensional optimization problem, and we have increased the number of parameters to be optimized only if the earlier try was not successful.

The most important properties of the bound constrained global optimization problems related to chaos verification are summarized in the following statement.

THEOREM 5. *For the bound constrained global optimization problem defined in* (4) *the following properties hold:*

1. *In case a global optimization algorithm finds a point for which the objective function g has a value below C, i.e. when each penalty term* $\max_{z\in T(I)} \inf_{y\in s_i} d(z, y)$ *is zero, then all the conditions of chaos are fulfilled by the found region represented by the corresponding optimal parameters x found. At the same time, the checking routine provides a guaranteed reliability computational proof of the respective subset relations.*
2. *In case the given problem does not have a parameter set within the bounds of the parameters to be optimized such that the corresponding region would fulfill the criteria of chaos, then the optimization of* $g(x)$ *cannot result in an approximate optimizer point with an objective function value below C.*

Proof. Consider first the case when a parameter set has been found for which $g(x) < C$ for the problem of the form (4). Then, due to the definition of g we have

$$p\left(\sum_{i=1}^{m} \max_{z\in T(I)} \inf_{y\in S_i} d(z, y)\right) = 0,$$

that is

$$\max_{z\in T(I)} \inf_{y\in S_i} d(z, y) = 0 \tag{5}$$

holds for each $i = 1, \ldots, m$. Now (5) can hold only when $I = \emptyset$, since the checking routine returns a subinterval I only when $T(I)$ has not been contained in the respective set of the checked condition. In addition to that, the checking routine applies a safe, interval arithmetic-based adaptive subdivision scheme, and no returned subinterval means that a computational proof has been completed that the checked region was covered by such subintervals that all fit into the aimed set of the condition (after the respective transformation). This proves the first part of Theorem 5.

Assume now that we have a mapping for which our conditions of chaos are not satisfied. In other words it means that for all possible regions there exists a subinterval I that contains at least one point of the investigated region q, and the transformed set of which has a point outside the set O on the right hand side of a condition. Now the checking routine cannot accept the given region, since then all points of the transformed I interval should have been feasible. Then the subinterval returned by the checking routine must have a point with a positive distance to all points of the

aimed set of the condition, i.e. the argument of the penalty function p cannot be zero. Hence $g(x) > C$, and that was to be proved. Notice that here this is true even for the inclusion X of the parameter vector x. □

The cases not covered by the two statements of Theorem 5 are those when the checking was inconclusive, e.g. since the excess width of the interval inclusion functions did not allow to cover the examined region with out of question intervals ($I \cap Q = \emptyset$), and with fully feasible subintervals of size larger than the user set threshold value. Although more precise results can be achieved at the cost of higher computational complexity and memory requirements, due to the high degree of nonlinearity involved, one cannot expect that all problems can be solved in reasonable time. On the other hand, substantial problem understanding can be utilized by the proper setting of the parameter bounds – as it is also demonstrated by our computational results summarized in the next section.

3. Numerical Results

For the computational experiments we have applied the C-XSC programming language (CXSC, 2005; Klatte et al., 1993) supporting interval arithmetic. The results were obtained both in Linux and in the Cygwin environment, on an average personal computer. In the present paper we just provide some demonstrative examples for the functioning of the suggested and introduced technique. First we have checked the reported chaotic region (Zgliczynski, 1997) by our checking routine (Algorithm 1).

3.1. Proving the chaotic behaviour for the 7th iterate Hénon mapping

We have investigated the seventh iterate of the Hénon mapping with the parameters of $A = 1.4$ and $B = 0.3$. The checked region consists of two parallelograms with sides parallel to the x-axis, the first coordinates of the lower corner points were 0.460, 0.556, 0.588, and 0.620, while the second coordinates were the same, 0.01. The common y coordinate for the upper corner points was 0.28, and the tangent of the sides was 2. We have set the ε threshold value for the checking routine to be 10^{-10}.

First the algorithm determined the starting interval, that contains the region to be checked:

$$[0.46000000000, 0.75500000000] \times [0.01000000000, 0.28000000000].$$

Then the three conditions were checked one after the other. All of these proved to be valid – as expected. The amount of function evaluations (for the transformation, i.e. for the seventh iterate of the Hénon mapping

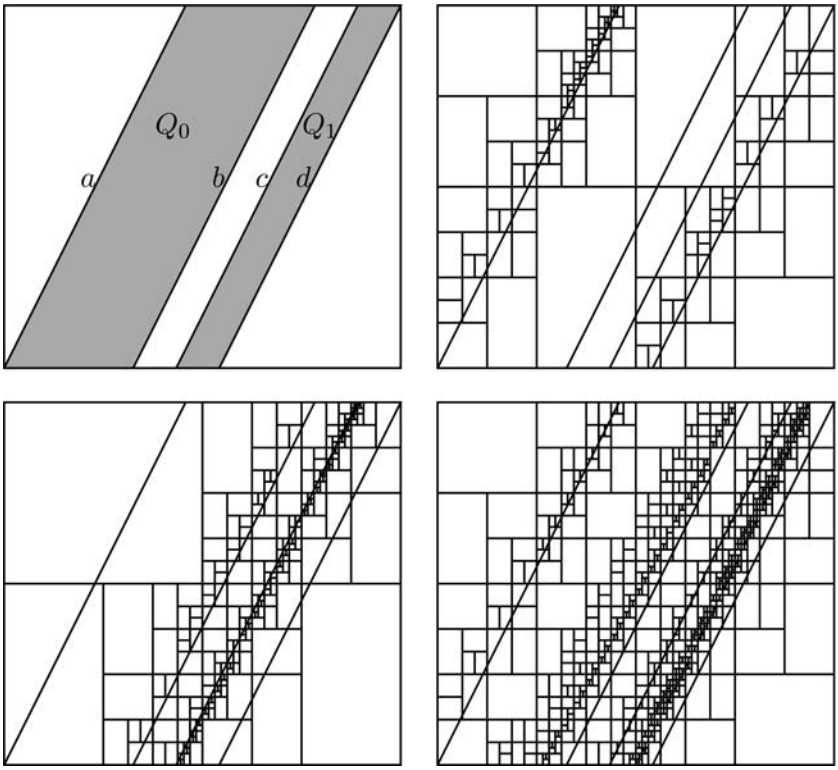

Figure 2. The parallelograms and the starting interval covered by the verified subintervals for which the given condition holds (in the order of mentioning in Theorem 1).

in each case) were 273, 523, and 1613, respectively. The algorithm stores those subintervals for which it was impossible to prove directly whether the given condition holds, these required further subdivision to achieve a conclusion. The depth of the stack necessary for the checking was 11, 13, and 14, respectively. The CPU time used proved to be negligible, only a few seconds.

The obtained results are demonstrated also on Figure 2 (together with the parallelograms). The density of the subintervals indicates that in the related subregion the given condition was just fulfilled, the overestimation involved in the interval calculations required much refinement.

Summarizing the results, we were able to prove with an acceptable amount of computation and human overhead that the published system is chaotic in the given, known regions. This confirms the result of Zgliczynski.

As a second step, randomly chosen A and B values were checked close to the classical parameters. The following ones ensured chaos for the $\mathcal{H}^7$ Hénon system with unchanged other region and algorithm parameters:

A	B
1.3555400848181643,	0.32668379383472889
1.3465721096594685,	0.32450555140362324
1.4403201855906845,	0.22585009468060412
1.4136297518450903,	0.26880306437090162
1.3702743902664050,	0.30756016043366862

Notice that without our automatic checking of the conditions for chaos it could have been very difficult when not even impossible to arrive at the above results, since the human interaction and insight necessary plus the required overhead could be prohibitive.

As a third way of applying the checking routine, we have determined parameter intervals around $A = 1.4$ and $B = 0.3$ for which mapping $\mathcal{H}^7$ still has chaos on the same pair of parallelograms. The obtained intervals were $A \in [1.377599, 1.401300]$ and $B \in [0.277700, 0.310301]$. Notice that these intervals do not contain all the A, B pairs given above.

The technique with which this result was obtained is an earlier interval optimization procedure able to solve tolerance optimization problems (Kristinsdottir et al., 1993, 1996; Csendes et al., 1995). The key feature necessary for this algorithm is that the checking routine can accept interval valued parameters for the calculated mapping.

3.2. LOCATING CHAOTIC REGIONS FOR THE 5TH HÉNON ITERATE

The most interesting question is obviously whether the interval arithmetic based reliable checking routine together with the penalty function approach applies to finding chaotic regions for different mappings. Since the lower the iterate, the more difficult to find a chaotic region, we have studied the fifth Hénon iterate (Figure 3). The reason why the sixth iterate was disregarded is that, assuming $B > 0$, even iterates of the Hénon mapping preserve whereas add iterates change the orientation. In particular, the geometry of H^6 on the plane differs considerably from those of H^5 and of H^7.

First we have repeated the run of the checking routine for $\mathcal{H}^5$ with the same parameter setting and region that was successful for $\mathcal{H}^7$. The result for the $\mathcal{H}^5$ case was that the conditions (1) and (2) were fulfilled, and the necessary number of function evaluations and the depths of the stack were 261 and 12 for (1), and 137 and 8 for (2), respectively. However, for the condition (3) we have obtained the small interval, the upper right corner of the starting interval

$$[0.46000000000, 0.75500000000] \times [0.01000000000, 0.28000000000]:$$

```
[0.75499999993131505782,   0.75500000000000000445]
[0.27999999993713575729,   0.28000000000000002665]
```

not complying with condition (3) after having spent 64 function evaluations and at the stack level of 65. The latter figure means that it was a depth first like search.

In other words we could find such a very small interval that has points inside the examined region, and which does not fulfill the third condition. Notice that the result small interval has coordinates the digits of which indicate the limitation of machine representation (here double precision). The width of the contradicting interval is less than $7.0 \cdot 10^{-11}$.

This phenomenon remained also when we have decreased the set stopping parameter ε from the default value of 10^{-10}. Neither were we able to find a chaotic region by changing those coordinates of the candidate region that caused seemingly the contradiction. Actually it was the experience that moved us to pose the chaotic region location problem as a constrained global optimization problem, since the latter seemed to be an easy, straightforward way to handle the highly nonlinear behaviour of the mapping together with the large dimensional parameter space of the region coordinates.

Turning to the location of a chaotic region for the fifth iterate Hénon mapping, we have used the constrained optimization model discussed in detail in Section 2.2. To solve the related bound constrained global optimization problem (involving the penalty functions), we have applied GLOBAL, a clustering stochastic global optimization algorithm (Csendes, 1988). It is capable to find the global optimizer points of moderate dimensional global optimization problems, when the relative size of the region of attraction of the global minimizer points is not very small. This global

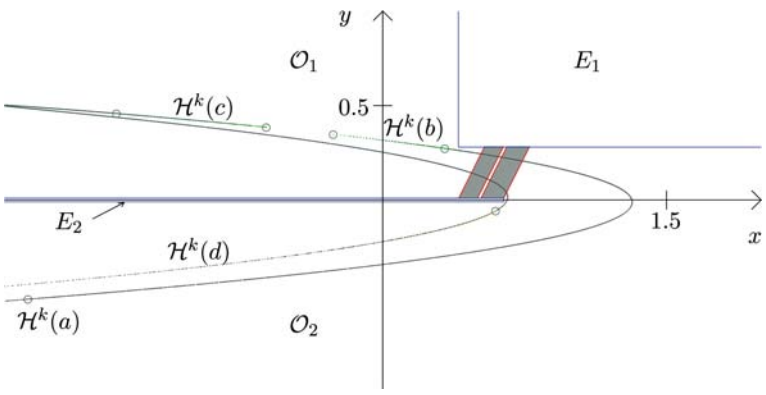

Figure 3. Illustration of the found chaotic H^5 transformation.

optimization method was successfully applied recently in the solution of difficult circle packing problems (Markót and Csendes, 2005).

The optimization model allowed to change the following parameters of the problem: first of all, the parameters of the mapping: A and B, then the coordinates of the parallelograms: x_a, x_b, x_c, x_d; the tangent of the lower left angle in the parallelograms; and finally also the description parameters of the sets of the conditions (1)–(3). In some cases we have decreased the number of parameters to be optimized, when there seemed to be a reasonable chance to obtain a positive result.

The location of a chaotic region for the 5th iterate Hénon mapping could not be accomplished by a single run of the global optimization algorithm. The typical runs of this experimental kind of work for 6–10 parameters required 10^4–10^5 function evaluations, and seconds to minutes of CPU time. For more difficult problems the number of function evaluations spent in the global phase (to find good starting points for the local phase) were closely equal to the needs of the local searches. For easier problems the cost of the global phase was larger.

Finally a found parameter set ensuring chaos for the $\mathcal{H}^5$ system:

$$
\begin{aligned}
A &= 1.7414857,\\
B &= 0.38127953,\\
x_a &= 0.40090670,\\
x_b &= 0.50113505,\\
x_c &= 0.51875196,\\
x_d &= 0.63916903,
\end{aligned}
$$

while all other parameters were set as in the original problem (e.g. the common tangent was again 2). More solved chaotic region location problems are reported with technical details in Bánhelyi et al. (2005).

4. Summary

We have introduced a reliable, interval arithmetic based checking algorithm for proving subset relations, and furthermore a constrained global optimization model for the location of regions that have chaotic properties for some particular Hénon mappings. We have proven that the algorithm and the model are correct and capable to find chaotic regions. We have checked chaos for an earlier investigated seventh iterate Hénon mapping. A tolerance optimization algorithm provided a positive width interval for those mapping parameters that ensure the conditions of chaos with the same region. We presented chaotic regions obtained by the proposed optimization technique for the more difficult fifth iterate of the Hénon mapping.

One of the main advantage of the presented approach is that without much understanding of the structure of the underlying mapping and with a modest overhead, one can locate chaotic regions of dynamic systems. The tolerance optimization method provided large sets of the parameters that allow chaos around known chaotic regions.

Acknowledgments

This work was supported by the Grants OMFB D-30/2000, OMFB E-24/2001, OTKA T 034350, T 048377, and T 046822. The authors are grateful to Mihály Görbe (GAMF, Kecskemét, Hungary) for coding some of the related programs.

References

1. Bánhelyi, B., Csendes, T. and Garay B.M., Optimization and the Miranda approach in detecting horseshoe-type chaos by computer. Manuscript, submitted for publication. Available at www.inf.u-szeged.hu/∼csendes/henon2.pdf
2. Csendes, T. (1988), Nonlinear parameter estimation by global optimization – efficiency and reliability. *Acta Cybernetica*, 8, 361–370.
3. Csendes, T., Bánhelyi B. and Hatvani L., Towards a computer-assisted proof for chaos in a forced damped pendulum equation. Manuscript, submitted for publication. Available at www.inf.u-szeged.hu/∼csendes/jcaminga.pdf
4. Csendes, T., Zabinsky, Z.B. and Kristinsdottir, B.P. (1995), Constructing large feasible sub-optimal intervals for constrained nonlinear optimization. *Annals of Operations Research*, 58, 279–293.
5. C-XSC Languages home page (2005), http://www.math.uni-wuppertal.de/org/WRST/index_en.html
6. Dellnitz, M. and Junge, O. (2002), Set oriented numerical methods for dynamical systems. Handbook of dynamical systems, Vol. 2, North-Holland, Amsterdam, pp. 221–264.
7. Galias, Z. and Zgliczynski, P. (2001), Abundance of homoclinic and heteroclinic orbits and rigorous bounds for the topological entropy for the Hénon map. *Nonlinearity*, 14, 909–932.
8. Klatte, R., Kulisch, U., Wiethoff, A., Lawo, C., and Rauch, M. (1993), *C-XSC - A C++ Class Library for Extended Scientific Computing*, Springer-Verlag, Heidelberg.
9. Kristinsdottir, B.P., Zabinsky, Z.B., Csendes, T., and Tuttle, M.E. (1993), Methodologies for tolerance intervals. *Interval Computations*, 3, 133–147.
10. Kristinsdottir, B.P., Zabinsky, Z.B., Tuttle, M.E., and Csendes, T. (1996), Incorporating manufacturing tolerances in optimal design of composite structures, *Engineering Optimization* 26, 1–23.
11. Markót, M.C. and Csendes, T. A New Verified Optimization Technique for the "Packing Circles in a Unit Squre" Problems. Accepted for publication in the SIAM J. on Optimization. Available at www.inf.u-szeged.hu/∼csendes/publ.html
12. Neumaier, A. (2004), Complete search in continuous global optimization and constraint satisfaction, *Acta Numerica*, 271–369.
13. Neumaier, A. and Rage, T. (1993), Rigorous chaos verification in discrete dynamical systems, *Physica D* 67, 327–346.

14. Rage, T., Neumaier, A. and Schlier, C. (1994), Rigorous verification of chaos in a molecular model, *Physical Review E*. 50, 2682–2688.
15. Ratschek H. and Rokne J., (1988), *New Computer Methods for Global Optimization*, Ellis Horwood, Chichester.
16. Zgliczynski, P. (1997), Computer assisted proof of the horseshoe dynamics in the Hénon map. *Random & Computational Dynamics* 5, 1–17.
17. Zgliczynski, P. (2003), On smooth dependence on initial conditions for dissipative PDEs, an ODE-type approach. *Journal of Differential Equations* 195, 271–283.