*$6^{th}$ International Conference on Applied Informatics*
*Eger, Hungary, January 27–31, 2004.*

# A verified computational technique to locate chaotic regions of Hénon systems*

**Balázs Bánhelyi[1] and Tibor Csendes[1]**

[1]University of Szeged, Institute of Informatics
e-mail: {banhelyi | csendes}@inf.u-szeged.hu

**Abstract**

We present a computer assisted proof for the existence of a horseshoe of the 7-th iterate classical Hénon map ($\mathcal{H}(x, y) = (1 + y - \alpha x^2, \beta x)$). An earlier, published theorem gives three geometrical conditions to be fulfilled by all points of the solution region, given by 2 parallelograms. We analyze these conditions separately and in case when all of them hold true, the proof is ready. The method applies interval arithmetic and recursive subdivision. This verified technique proved to be fast on the investigated problem instances.

**Categories and Subject Descriptors:** Computer Algorithms and other Fields of Applied Informatics

**Key Words and Phrases:** Chaos, Hénon-map, Verified Method

## 1 Introduction

We study verified computational methods to check and locate regions containing horseshoes, the prototype of chaotic behaviour. The investigated Hénon map was $\mathcal{H}(x, y) = (1 + y - \alpha x^2, \beta x)$. The paper [4] considered the $\alpha = 1.4$ and $\beta = 0.3$ values and some regions of the two dimensional Euclidean space: $E_1 = \{(x, y) \mid x \geqslant 0.4,\ y \geqslant 0.28\}$, $E_2 = \{(x, y) \mid x \leqslant 0.64,\ |y| \leqslant 0.01\}$, $E = E_1 \cup E_2$, $O_1 = \{(x, y) \mid x < 0.4,\ y > 0.01\}$, $O_2 = \{(x, y) \mid y < 0\}$.

The sets $Q_1, Q_2 \subset \{(x, y) \mid 0.01 = y_0 \leqslant y \leqslant y_1 = 0.28\}$ are parallelograms connecting $E_1$ and $E_2$ with horizontal sides and sides parallel to the line of equation $y = x \tan\varphi$, where $\varphi = \tan^{-1} 2$. The $x$ coordinates of the lower vertices of the parallelograms are $x_a$, $x_b$, $x_c$, and $x_d$. The nonhorizontal sides of the parallelograms $Q_1$ resp. $Q_2$ are detoned

*This work was supported by the Grants OMFB D-30/2000, OMFB E-24/2001, OTKA T 032118 and T 034350. The authors are grateful to Barnabás Garay (BME, Budapest, Hungary) and Mihály Görbe (GAMF, Kecskemét, Hungary) for their collaboration.

by $a, b$ resp. $c, d$, where

$$\sigma = \left\{(x_\sigma + t, y_0 + t\ \tan\varphi) \mid t \in [0, \frac{y_1 - y_0}{\tan\varphi}]\right\}, \sigma = a, b, c, d$$

and $x_a < x_b < x_c < x_d < 0.64$. Note that $\varphi$ is the lower left angle of the parallelograms and the $\tan\varphi$ is the tangent of the lower left angle of the sides. This notations are demonstrated on Figure 1.

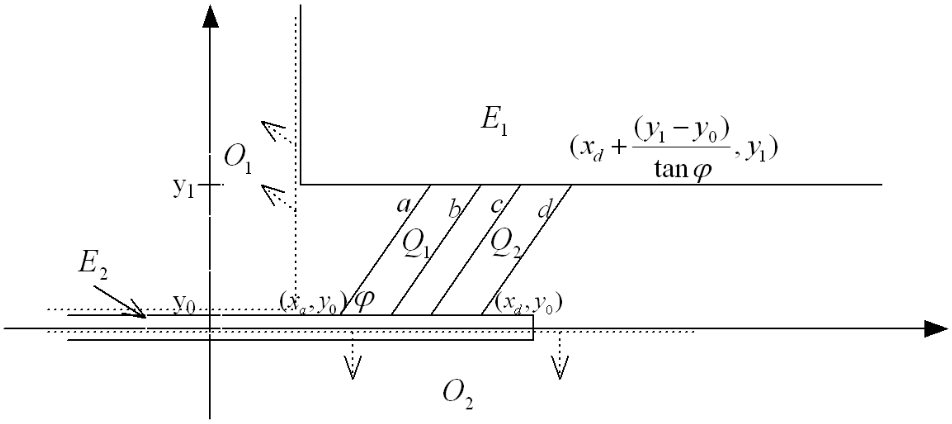


Figure 1: The considered parallelograms and other notations.

For the earlier described settings this statement was proven:

**Theorem 1.1** *Assume that, for some positive integer $k$, the conditions:*

$$\mathcal{H}^k(a \cup d) \subset O_2,$$

$$\mathcal{H}^k(b \cup c) \subset O_1,$$

$$\mathcal{H}^k(Q_1 \cup Q_2) \subset \mathbb{R}^2 \setminus E,$$

*hold true. Then $\mathcal{H}^k$ has a horseshoe in $Q_1 \cup Q_2$.*

According to [4, 9] Theorem 1.1 ensures the chaotic behaviour for the points of the parallelograms $Q_1$ and $Q_2$ with parallel sides with the $x$ axis (for $y = 0.01$ and $y = 0.28$, respectively), with the common tangent of 2, and lower vertices of 0.460, 0.556; and 0.558, 0.620, respectively. You can see this in Figure 2.

Zgliczynski's proof is based on the fixed point index with computer assisted rigorous computations. He used a Lipschitz map to eliminate the errors caused by the floating point arithmetic. Instead of this we used interval arithmetic. We have substituted real numbers by intervals. If an obtained number is not representable, then it is rounded outward.

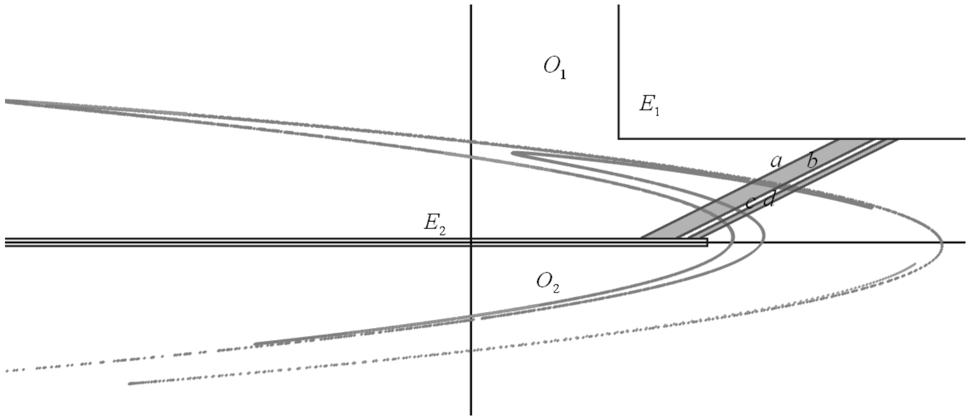


Figure 2: Illustration of the Hénon map with original parameters.

**Definition 1.1** *The interval arithmetic is defined by:*

$$A \circ B = \{a \circ b \mid a \in A \text{ and } b \in B\}, \quad A, B \in \mathbb{I}$$

($\mathbb{I}$ is the set of $[i, j]$ intervals, where $i, j \in \mathbb{R}$, and $i < j$).

*These were implemented as:*

$$[a, b] + [c, d] = [a + c, b + d],$$
$$[a, b] - [c, d] = [a - d, b - c],$$
$$[a, b] * [c, d] = [\min(ac, ad, bc, bd), \max(ac, ad, bc, bd)],$$
$$[a, b]/[c, d] = [a, b] * [1/d, 1/c] \text{ if } 0 \notin [c, d].$$

More on interval based inclusion funcions can be read in [3, 5, 8]

# 2 New algorithms for checking the chaotic behaviour and for locating regions containing such points

In the proof the earlier conditions are analyzed separately, and in the case when all of them hold true, the proof is complete. The adaptive subdivision technique is executed three times independently.

The subdivision technique encloses first the sets $Q_1$ and $Q_2$ in a 2-dimensional interval. Hence the starting interval is the 2-dimensional interval:

$$[x_a, x_d + \frac{(y_1 - y_0)}{\tan \varphi}][y_0, y_1].$$

Then an adaptive subdivision technique generates such a subdivision of the starting interval that either:

- for all subintervals all the 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 contradicts the given condition.

To realize this algorithm we have applied a modified version of an earlier procedure to solve constrained nonlinear optimization problems with tolerance [2, 6, 7].

The algorithm is a subdivision technique:

**Step 1.** Calculate the initial interval that contains the regions of interest.

**Step 2.** Push the initial interval into the stack.

**Step 3.** Pop an interval $v$ out of the stack.

**Step 4.** Measure the width of $v$, and determine the widest coordinate direction.

**Step 5.** Use the actual $\mathcal{H}^k$ transformation to $v$ to obtain the interval $w$.

**Step 6.** If the interval $v$ has at least one point from the argument set,

and the condition does not hold for $w$, then

If the width of interval $v$ is less than the user set limit given, then

print $v$ and STOP,

else bisect $v$ along the widest side,

push the subintervals into the stack and go to Step 3.

**Step 7.** If the stack is empty then print that the condition is proven and STOP, else go to Step 3.

One step of the algorithm is demonstrated on Figure 3.

For the numerical experiments we have applied the C-XSC programming language [3, 5, 8] supporting interval arithmetic. The results were obtained both in Linux and in the Cygwin environment, on an average personal computer. The necessary CPU times were seconds.

# 3 Results and summary

In the first experiment we have checked whether the earlier published regions are in fact chaotic. The result file:

```
Checking the conditions of chaos for a Henon system

epsilon =   0.00000000010000000000

exponent=   7
```

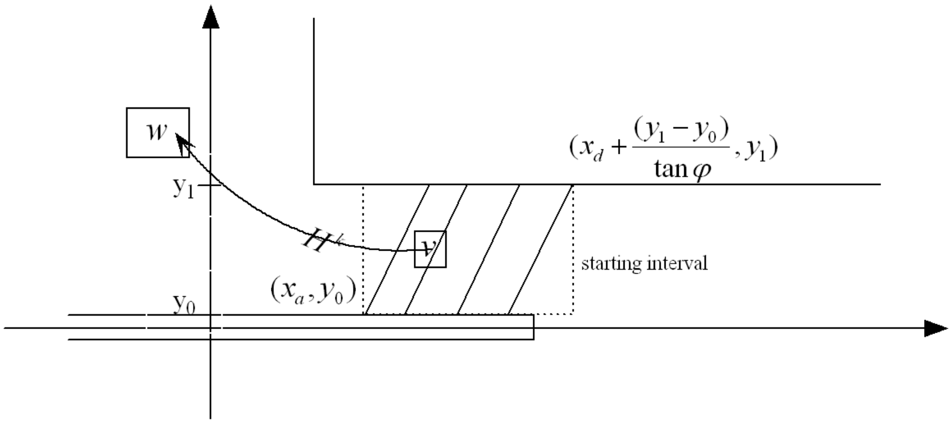


Figure 3: One example for $v_i$ and $\mathcal{H}^k(v_i)$.

```
alpha   =   1.39999999999999991118
beta    =   0.29999999999999998890

x_a     =   0.46000000000000001998
x_b     =   0.55600000000000004974
x_c     =   0.58799999999999996714
x_d     =   0.61999999999999999556
y_0     =   0.01000000000000000021
y_1     =   0.28000000000000002665
tangent =   2.00000000000000000000

Starting interval:
[ 0.46000000000000001998, 0.75500000000000000445]
[ 0.01000000000000000020, 0.28000000000000002665]


1. condition: successful proof!
Number of function evaluations: 273
Largest depth of the stack: 11

2. condition: successful proof!
Number of function evaluations: 523
Largest depth of the stack: 13

3. condition: successful proof!
Number of function evaluations: 1613
Largest depth of the stack: 14
```

In this way we were able to prove with a short computation that the published system is chaotic in the given regions. Those intervals calculated by the program are given in Figure 4.

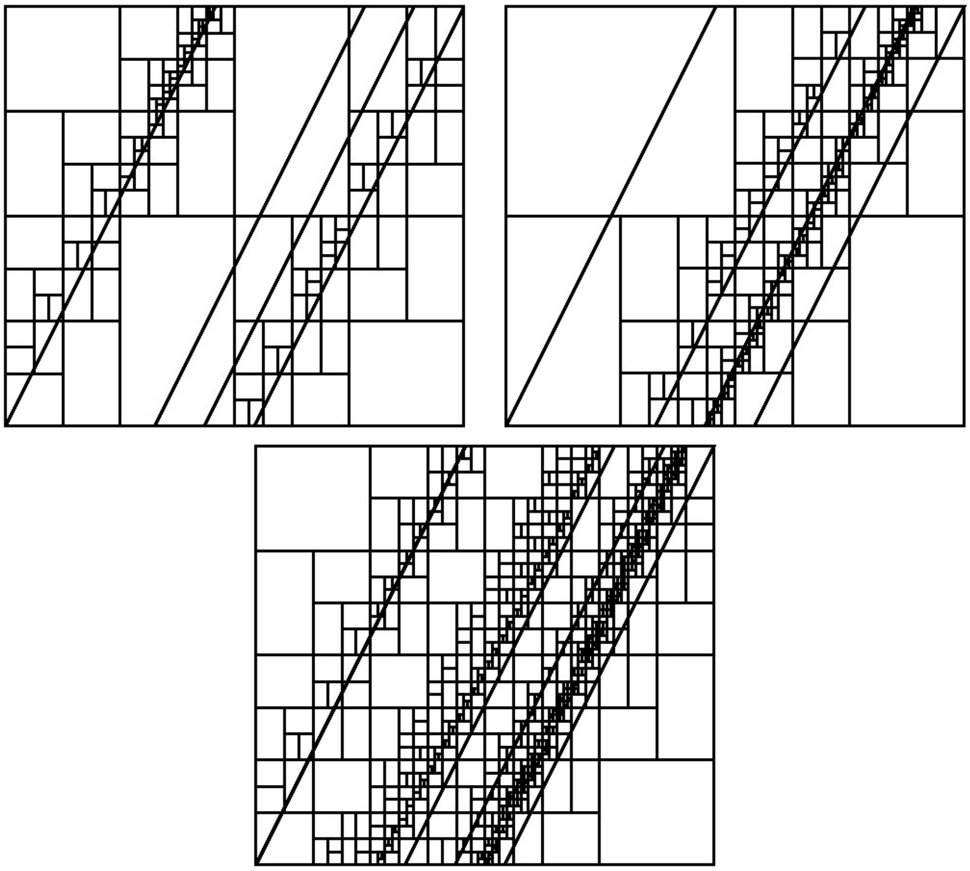

Figure 4: Verified intervals for conditions I-II-III, respectively.

As a next experiment we have extended the parameters of the Hénon map, whether the chaotic behaviour still holds. The obtained result for the $\mathcal{H}^5$ case with the same parameter setting and with the same starting region:

```
Checking the conditions of chaos for a H\'enon system

epsilon =   0.00000000010000000000

exponent=   5

alpha   =   1.39999999999999991118
beta    =   0.29999999999999998890
```

```
x_a     =   0.46000000000000001998
x_b     =   0.55600000000000004974
x_c     =   0.58799999999999996714
x_d     =   0.61999999999999999556
y_0     =   0.01000000000000000021
y_1     =   0.28000000000000002665
tangent =   2.00000000000000000000

Starting interval:
[ 0.46000000000000001998, 0.75500000000000000445]
[ 0.01000000000000000020, 0.28000000000000002665]

A small interval not complying with the conditions:
[ 0.75499999993131505782, 0.75500000000000000445]
[ 0.27999999993713575729, 0.28000000000000002665]


1. condition: Unsuccessful proof!
Number of function evaluations: 64
Largest depth of the stack: 65

2. condition: successful proof!
Number of function evaluations: 261
Largest depth of the stack: 12

3. condition: successful proof!
Number of function evaluations: 137
Largest depth of the stack: 8
```

In other words we could find such a very small interval that has points inside the examined region, which does not fulfill the first condition. Notice that also the starting interval and 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 $7.0 \cdot 10^{-11}$.

Summarizing the paper we can conclude that our program has been implemented in C, and the verification of Theorem 1.1 for $k = 7$ required the calculation of $\mathcal{H}^7$ for 2409 intervals. This took around one second on an IBM PC PIV. Thus we can use this technique with some modification in other problems. We solved some other problems with the described method:

- locate a chaotic region for a Hénon 5-th iterate,
- determine a whole set of those parameter values that ensure the chaotic behavior for the $\mathcal{H}^7$ transformation with the same parallelograms.

You can read more on these problems in [1].

# References

[1] Balázs Bánhelyi's Hénon home page:
`http://www.inf.u-szeged.hu/~banhelyi/Henon`

[2] Csendes, T., Z.B. Zabinsky, and B.P. Kristinsdottir: Constructing large feasible suboptimal intervals for constrained nonlinear optimization. Annals of Operations Research, 58(1995) 279-293.

[3] C-XSC Languages home page:
`http://www.math.uni-wuppertal.de/org/WRST/index_en.html`

[4] Galias, Z. and P. Zgliczynski. Computer assisted proof of chaos in the Lorenz equations. Physica D, 115(1998) 165-188.

[5] Klatte, R., U. Kulisch, A. Wiethoff, C. Lawo, and M. Rauch: C-XSC — A C++ Class Library for Extended Scientific Computing, Springer-Verlag, Heidelberg, 1993.

[6] Kristinsdottir, B.P., Z.B. Zabinsky, T. Csendes, and M.E. Tuttle: Methodologies for tolerance intervals. Interval Computations, 3(1993) 133-147.

[7] Kristinsdottir, B.P., Z.B. Zabinsky, M.E. Tuttle, and T. Csendes: Incorporating manufacturing tolerances in optimal design of composite structures, Engineering Optimization 26(1996) 1-23.

[8] Mischaikow, K. and M. Mrozek , Chaos in the Lorenz equations: A computer assisted proof. Part II: Details, Mathematics of Computation, 67 (1998) 1023-1046

[9] Zgliczynski, P.: Computer assisted proof of the horseshoe dynamics in the Hénon map. Random & Computational Dynamics 5(1997) 1-17.

# Postal addresses

**Balázs Bánhelyi and Tibor Csendes**
*University of Szeged,*
*Institute of Informatics,*
*H-6701 Szeged, P.O. Box 652,*
*Hungary*