Rigorous Study of the Chua's Circuit Spiral Attractor

Zbigniew Galias, *Member, IEEE,*

***Abstract*—It is shown that a certain set is positively invariant for the return map associated with the Chua's circuit. The set contains the intersection of the numerically observed spiral attractor and the planes defining the return map. A method for rigorous integration of piece-wise linear systems in regions containing trajectories tangent to hyperplanes separating the linear regions necessary to carry out the proof is developed.**

***Index Terms*—Chua's circuit, piece-wise linear system, trapping region, interval arithmetic.**

I. Introduction

The dynamics of the Chua's circuit is well understood in terms of geometrical models [1], [2]. There are however only few rigorous results concerning chaotic dynamics for this system. The existence of a homoclinic orbit for some unknown parameter value within a certain range was shown in [3]. The existence of a nontrivial symbolic dynamics embedded in the double-scroll attractor was proved in [4]. It was shown that the system is chaotic in the topological sense, i.e. that the topological entropy of the flow is positive. The lack of results concerning the whole attractor is caused by the fact that there are no general tools for the rigorous integration of piece-wise linear (PWL) systems.

In this work, we consider the Chua's circuit with parameter values for which one observes a spiral attractor containing trajectories tangent to hyperplanes separating linear regions (called in the following the C^0-hyperplanes). Integration of such trajectories poses a problem, since standard methods [5], [6] which work under the assumption that the vector field is smooth are not applicable.

When intersections of trajectories with the C^0-hyperplanes are transversal it is possible to extend methods developed for smooth systems to integration of PWL systems. This is achieved by using the C^0-hyperplanes as transversal sections. When a trajectory intersects a C^0-hyperplane, its intersection with the plane is computed and the result is used as a set of initial conditions for further computations. This approach has been successfully used to find the trapping region for the return map associated with the Chua's circuit for the case when the attractor does not contain trajectories tangent to the C^0-hyperplanes [7]. Clearly, this method will fail if some trajectories of interest are tangent to the C^0-hyperplanes.

This work was supported in part by the AGH University of Science and Technology, grant no. 11.11.120.611.

Z. Galias is with the AGH University of Science and Technology, Department of Electrical Engineering, 30–059 Krakow, Poland (e-mail: galias@agh.edu.pl).

In order to carry out the analysis of the dynamics over the whole spiral attractor we develop a method which can be used to rigorously integrate PWL systems also in the case of a tangency. This is an important tool in studies of PWL systems since it provides a general technique for computing enclosures of trajectories in such systems. The technique proposed is based on the theory of differential inclusions used to obtain estimates for solutions of perturbed continuous dynamical systems.

The paper is organized as follows. In Section II, the method for integration of PWL systems when intersections with the C^0-hyperplanes are transversal is recalled and then the procedure for integrating PWL systems for the tangent case is presented. A toy example of a two-dimensional system is considered to explain how the method works. In Section III, rigorous analysis of the Chua's circuit with the spiral attractor is performed. The case when some trajectories belonging to the attractor are tangent to the C^0-hyperplanes is considered. It is proved that a certain region is a positively invariant set for an associated return map. According to our knowledge this has never been described in the literature before. Graph representation of the dynamics is constructed and bounds for the average return time over the attractor are calculated. Discussion on the performance of the algorithms is presented.

The basic tool used to make the results of numerical computations rigorous is interval arithmetic [8]. In the following, boldface is used to denote intervals, interval vectors and matrices, and the usual math italics is used to denote point quantities. For a given interval $\mathbf{x} = [a, b]$ by $\underline{x}$ and $\overline{x}$ we denote its left and right end points respectively, i.e. $\underline{x} = a$ and $\overline{x} = b$. The diameter of the interval $\mathbf{x}$ is defined as $\mathrm{diam}(\mathbf{x}) = \overline{x} - \underline{x}$. The diameter of the interval vector $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n)$ is defined as the maximum of diameters $\mathrm{diam}(\mathbf{x}_k)$.

II. Rigorous integration of PWL systems

Let the piece-wise linear (PWL) system be defined by

$$\dot{x} = f(x), \tag{1}$$

where $f\colon \mathbb{R}^n \mapsto \mathbb{R}^n$ is a PWL continuous map. By $x(t) = \varphi(t, \hat{x})$ we denote the solution of (1) satisfying the initial condition $x(0) = \hat{x}$.

We assume that the state space $\mathbb{R}^n$ is composed of m linear regions $R_1, R_2, \ldots R_m$, and that in the region R_k the state equation (1) has the form

$$\dot{x} = A_k x + v_k, \tag{2}$$

where $A_k \in \mathbb{R}^{n\times n}$, and $v_k \in \mathbb{R}^n$. If A_k is invertible then in the linear region R_k solutions can be computed as

$$x(t) = \varphi_k(t, \hat{x}) = \mathrm{e}^{A_k t}(\hat{x} - w_k) + w_k, \tag{3}$$

where $w_k = -(A_k)^{-1} v_k$.

Let $\Sigma_1, \Sigma_2, \dots, \Sigma_p$ be hyperplanes separating the linear regions $R_1, R_2, \dots R_m$. They will be referred to as the C^0-hyperplanes.

In this section, we discuss how to calculate an enclosure of the set $\varphi(\mathbf{t}, \mathbf{x}) = \{\varphi(t, x) \colon t \in \mathbf{t}, x \in \mathbf{x}\}$ for a given interval $\mathbf{t} = [\underline{t}, \overline{t}]$ with $\underline{t} > 0$ and an interval vector $\mathbf{x} \subset \mathbb{R}^n$. We describe methods for finding enclosures for $\varphi(\mathbf{t}, \mathbf{x})$ in various cases. First, the case when trajectories for $t \in \mathbf{t}$ stay in a single linear region is considered.

A. Solutions enclosed in a single linear region

If all trajectories based at $\mathbf{x} \subset R_k$ remain in R_k for $s \in [0, \overline{t}]$ the problem is simple. The enclosure can be found by evaluating the formula (3) in interval arithmetic (for details see [4]). More precisely, one computes

$$\mathbf{y} = \mathrm{e}^{A_k \mathbf{t}}(\mathbf{x} - w_k) + w_k. \tag{4}$$

Since the calculations are performed in interval arithmetic it is ensured that the result $\mathbf{y}$ encloses the true solution. i.e. $\varphi_k(\mathbf{t}, \mathbf{x}) = \{\varphi_k(t, x) \colon t \in \mathbf{t}, x \in \mathbf{x}\} \subset \mathbf{y}$.

B. Transversal intersections

Another relatively easy case is when all trajectories based at $\mathbf{x} \subset R_k$ enter another linear region R_l through the hyperplane Σ_j, and intersections of trajectories with Σ_j are transversal. The first step is to find $s_1 > 0$ such that $\varphi_k([0, s_1], \mathbf{x}) \in R_k$ and $s_2 > s_1$ such that $\varphi_k(s_2, \mathbf{x}) \subset R_l$. In order to obtain a narrow enclosure, s_1 should be as large as possible while $s_2 > s_1$ should be as small as possible. s_1 and s_2 can be optimized using the bisection technique. Next, one evaluates $\mathbf{y} = \varphi_k(\mathbf{s}, \mathbf{x})$, where $\mathbf{s} = [s_1, s_2]$ and finally, the intersection $\mathbf{z} = \mathbf{y} \cap \Sigma_j$ is computed. The intersection $\mathbf{z}$ serves as a set of initial conditions for further computations. The problem of computing $\varphi(\mathbf{t}, \mathbf{x})$ has been reduced to the problem of computing $\varphi(\mathbf{t} - \mathbf{s}, \mathbf{z})$. Let us note that since $\mathbf{z} \subset \Sigma_j \subset R_l$, the hyperplane Σ_j has been crossed, and we can start computations in the next linear region.

The procedure for computing an enclosure for $\varphi(\mathbf{t}, \mathbf{x})$ utilizing this method of crossing a C^0-hyperplane is presented as the Algorithm 1. The Algorithm works as long as trajectories of interest transversally intersect the C^0-hyperplanes. It has been successfully applied to the analysis of the Chua's circuit for parameter values, for which the attractor does not contain trajectories tangent to the C^0-hyperplanes (see [7]).

This method fails if for $x \in \mathbf{x}$, a trajectory $\varphi([0, \overline{t}], x)$ is tangent to a hyperplane separating linear regions. This case is handled in the following sections.

Algorithm 1 Computation of $\varphi(\mathbf{t}, \mathbf{x})$, transversal case

```
loop
  $k \Leftarrow$ index such that $\mathbf{x} \subset R_k$
  find $s_1 > 0$ such that $\varphi_k([0, s_1], \mathbf{x}) \subset R_k$
  if $s_1 > \overline{t}$ then
    return $\mathbf{y} = \varphi_k(\mathbf{t}, \mathbf{x})$
  end if
  find $s_2 > s_1$ such that $\varphi_k(s_2, \mathbf{x}) \cap R_k = \emptyset$
  $\mathbf{s} \Leftarrow [s_1, s_2]$
  $\mathbf{y} \Leftarrow \varphi_k(\mathbf{s}, \mathbf{x})$
  $l \Leftarrow$ index such that $\varphi_k(s_2, \mathbf{x}) \subset R_l$
  $j \Leftarrow$ index such $\Sigma_j$ separates $R_k$ and $R_l$
  if the vector field $f$ over $\mathbf{y}$ is not transversal to $\Sigma_j$ then
    return ERROR
  end if
  $\mathbf{x} \Leftarrow \mathbf{y} \cap \Sigma_j$
  $\mathbf{t} \Leftarrow \mathbf{t} - \mathbf{s}$
end loop
```

C. Integration of perturbed dynamical systems

Let us start by formulating a theoretical result which allows one to compute enclosures of solutions of perturbed continuous dynamical systems. Let us consider an ordinary differential equation

$$\dot{x} = f(x), \tag{5}$$

where $x \in \mathbb{R}^n$ and $f \colon \mathbb{R}^n \mapsto \mathbb{R}^n$. Let us assume that we know how to integrate

$$\dot{x} = g(x), \tag{6}$$

which is a perturbation of (5). The following theorem provides bounds on solutions of (5) based on solutions of (6).

Theorem 1: Let $x(t)$ and $y(t)$ be solutions of (5) and (6), respectively. Let us assume that the map g is C^1, $x(0) = y(0)$, and $x(t), y(t) \in D \subset \mathbb{R}^n$ for $t \in [0, h]$, where the set D is a bounded, closed, and convex. Then, for $t \in [0, h]$

$$|y_i(t) - x_i(t)| \leq \Delta_i, \tag{7}$$

where

$$\Delta = \int_0^t \mathrm{e}^{B(t-s)} c \mathrm{d}s, \tag{8}$$

$$B_{ij} \geq \begin{cases} \sup_{x \in D} \left| \frac{\partial g_i}{\partial x_j}(x) \right|, & \text{for } i \neq j, \\ \sup_{x \in D} \frac{\partial g_i}{\partial x_j}(x), & \text{for } i = j, \end{cases} \tag{9}$$

and

$$c_i \geq |g_i(x(t)) - f_i(x(t))|, \quad \text{for } t \in [0, h]. \tag{10}$$

The above theorem is a conclusion from the results on integration of differential inclusions developed in [9], [10].

D. Tangent intersections

Here, it is shown how to use Theorem 1 for integration of PWL systems in regions where trajectories are tangent to the C^0-hyperplanes. A simplified version of this method was given in [11]. Here, we present a more detailed explanation together with several improvements, which make the method applicable to study of the Chua's circuit.

Let us assume that $\hat{x} \in R_k$, the trajectory $\varphi([0,\tau),\hat{x}) \subset R_k$, $\varphi(\tau,\hat{x}) \in \Sigma_j$, where Σ_j is the hyperplane separating the linear regions R_k and R_l. Further, we assume that the trajectory $\varphi(t,\hat{x})$ is tangent to Σ_j at the intersection point $\varphi(\tau,\hat{x})$. The goal is to compute an enclosure of the set $\varphi(\mathbf{t},\mathbf{x}) = \{\varphi(t,x) \colon x \in \mathbf{x}, t \in \mathbf{t}\}$, for a given interval vector $\mathbf{x}$ containing $\hat{x}$ and $\mathbf{t} = [\underline{t},\overline{t}]$ such that $\underline{t} > \tau$.

To solve the problem we consider the PWL system (1) as a perturbation of the linear system:

$$\dot{x} = g(x) = A_k x + v_k. \tag{11}$$

In this case the elements of the matrix B defined in the Theorem 1 can be computed as $B_{ij} = |(A_k)_{ij}|$ for $i \neq j$ and $B_{ii} = (A_k)_{ii}$. One can easily show that

$$\Delta = \int_0^t \mathrm{e}^{B(t-s)} c \mathrm{d}s = \int_0^t \mathrm{e}^{Bs} c \mathrm{d}s = t \sum_{i=0}^{\infty} \frac{(Bt)^i}{(i+1)!} \cdot c. \tag{12}$$

When B is invertible the above formula reduces to

$$\Delta = \int_0^t \mathrm{e}^{B(t-s)} c \mathrm{d}x = B^{-1}\left(\mathrm{e}^{Bt} - I\right) c. \tag{13}$$

The difference between g and f is zero over the region R_k and for the region R_l can be computed as:

$$g(x) - f(x) = (A_k - A_l)x + v_k - v_l. \tag{14}$$

From the continuity of the vector field f it follows that when the trajectory remains close to the hyperplane Σ_j the difference (14) is small.

The procedure starts by finding $s_1 > 0$ such that $\varphi_k([0,s_1],\mathbf{x}) \subset R_k$. The set $\mathbf{u} = \varphi_k(s_1,\mathbf{x})$ serves as an initial condition for integration along the tangency. To reduce overestimation s_1 should be as large as possible. It can be optimized using the bisection method. In the second part of the procedure, the PWL system is treated as a perturbed linear system. We select s_2, compute enclosure $\mathbf{v}$ of the solution $\varphi_k([0,s_2],\mathbf{u})$ of the linear system (11). Next, the set $\mathbf{v}$ is inflated to form the interval vector $\mathbf{w} \supset \mathbf{v}$, which serves as a guess of the set containing the solution $\varphi([0,s_2],\mathbf{u})$ of the nonlinear system. Next, one computes the vector $c = \sup_{x\in\mathbf{w}} |g(x) - f(x)|$. It is found by evaluating in interval arithmetic the formula (14) over the set $\mathbf{w} \cap R_l$, taking absolute value of each element and selecting the right end-points of the interval vector as a result. Finally, the vector Δ if found using formula (12) or (13). If $\mathbf{v} + [-1,1]\Delta \subset \mathbf{w}$ we know from the Theorem 1 that the solution of the PWL system is enclosed in $\mathbf{v} + [-1,1]\Delta$. It follows that $\varphi(s_2,\mathbf{u}) \subset \mathbf{z} = \varphi_k(s_2,\mathbf{u}) + [-1,1]\Delta$. If $\mathbf{z} \cap \Sigma_j = \emptyset$ and the vector field f over the set $\mathbf{z}$ points away from the hyperplane Σ_j we can break the computation along the tangency and continue integration in the next linear region.

Algorithm 2 Computation of $\varphi(\mathbf{t},\mathbf{x})$, general case

```
loop
  $k \Leftarrow$ index such that $\mathbf{x} \subset R_k$
  find $s_1$ such that $\varphi_k([0,s_1],\mathbf{x}) \subset R_k$
  if $s_1 > \overline{t}$ then
    return  $\mathbf{y} = \varphi_k(\mathbf{t},\mathbf{x})$
  end if
  $\mathbf{u} \Leftarrow \varphi_k(s_1,\mathbf{x})$
  $\mathbf{t} \Leftarrow \mathbf{t} - s_1$
  select $s_2 > 0$
  $l \Leftarrow$ index such that $\varphi_k(s_2,\mathbf{u}) \subset R_k \cup R_l$
  $j \Leftarrow$ index such $\Sigma_j$ separates $R_k$ and $R_l$
  SectionPassed $\Leftarrow$ false
  repeat
    $\mathbf{v} \Leftarrow \varphi_k([0,s_2],\mathbf{u})$
    select $\mathbf{w} \supset \mathbf{v}$
    repeat
      compute $c = \sup_{x\in\mathbf{w}} |g(x) - f(x)|$ using (14)
      compute $\Delta$ using (12) or (13)
      $\mathbf{z} \Leftarrow \varphi_k(s_2,\mathbf{u}) + [-1,1]\Delta$
      if $\mathbf{v} + [-1,1]\Delta \subset \mathbf{w}$ and $\mathbf{z} \cap \Sigma_j = \emptyset$ then
        $\mathbf{x} \Leftarrow \mathbf{z}$
        $\mathbf{t} \Leftarrow \mathbf{t} - s_2$
        SectionPassed $\Leftarrow$ true
      end if
      select larger $\mathbf{w}$
    until SectionPassed or $\mathrm{diam}(\mathbf{w}) > d_{\max}$
    select larger $s_2$
  until SectionPassed
end loop
```

The procedure for passing C^0-hyperplanes based on integration of a perturbed linear system is utilized in the Algorithm 2 for finding an enclosure for $\varphi(\mathbf{t},\mathbf{x})$. In the inner repeat loop the conditions $\mathbf{v} + [-1,1]\Delta \subset \mathbf{w}$ and $\mathbf{z} \cap \Sigma_j = \emptyset$ are verified. If at least one of them is not satisfied, the interval vector $\mathbf{w}$ is inflated and the inner cycle is repeated. When the size of $\mathbf{w}$ exceeds a predefined value $d_{\max}$ the inner loop is left and s_2 is increased.

Let us note that in the algorithm it is never assumed that the intersection is tangent. Therefore, the Algorithm 2 is general in the sense that it can be also used to cross a C^0-hyperplane in a transversal case.

Let us also note that for the transversal case we have a choice of using either the simple method to cross the section described in the Algorithm 1 or the perturbation based method implemented in the Algorithm 2. These two options are compared in the Section III. It will be shown that the perturbation based method offers superior performance.

E. An illustrative example

To illustrate how the method works let us consider a two-dimensional PWL system:

$$\begin{pmatrix} \dot{x}_1 \\ \dot{x}_2 \end{pmatrix} = f\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} a_{11}x_1 + a_{12}x_2 + (|x_1 - 1| - 1)e \\ a_{21}x_1 + a_{22}x_2 \end{pmatrix}. \tag{15}$$

There are two linear regions $R_1 = \{x\colon x_1 \le 1\}$ and $R_2 = \{x\colon x_1 \ge 1\}$ separated by the line $\Sigma_1 = \{x\colon x_1 = 1\}$. The only point at which trajectories are tangent to Σ is $(x_1, x_2) = (1, e - a_{11}/a_{12})$. This can be found by solving the equations $x_1 = 1$, $\dot{x}_1 = 0$.

To find solutions of the nonlinear system (15) we treat it as a perturbation of the linear system:

$$\begin{pmatrix}\dot{x}_1\\ \dot{x}_2\end{pmatrix} = g\begin{pmatrix}x_1\\ x_2\end{pmatrix} = \begin{pmatrix}a_{11}x_1 + a_{12}x_2 + (x_1-2)e\\ a_{21}x_1 + a_{22}x_2\end{pmatrix}. \quad (16)$$

For $x_1 > 1$ vector fields (15) and (16) are equal.

Hence, we can get bounds for the solution $x(t)$ of (15) from the solution $y(t)$ of (16) using Theorem 1 with

$$B = \begin{pmatrix}a_{11}+e & |a_{12}|\\ |a_{21}| & a_{22}\end{pmatrix}, c = \begin{pmatrix}\sup_{x\in\mathbf{w}} |(|x_1-1| - x_1 + 1)e|\\ 0\end{pmatrix}.$$

Let us select $a_{11} = 2$, $a_{12} = a_{21} = a_{22} = 1$, and $e = 2$. In this case the point of tangency is $(x_1, x_2) = (1, 0)$. The integration procedure is tested using the set of initial conditions $\mathbf{x} = ([1.004, 1.0045], [-0.099, -0.091]) \subset U_2$ and the integration time $t = 0.2$. The interval vector $\mathbf{x}$ and three trajectories starting from the center and two corners of $\mathbf{x}$ are plotted in Fig. 1. One can see that the trajectory based at the center of $\mathbf{x}$ passes close to the point of tangency, the trajectory based at the top-right corner of $\mathbf{x}$ does not leave the linear region R_2, while the trajectory based at the bottom-left corner intersects the line $x = 1$ two times. No matter how we divide $\mathbf{x}$ into smaller rectangles there will always be at least one interval vector containing all three types of trajectories (tangent to Σ, with no intersections with Σ, and with two intersections). The tangent case is therefore unavoidable if we want to integrate the system for the whole $\mathbf{x}$.

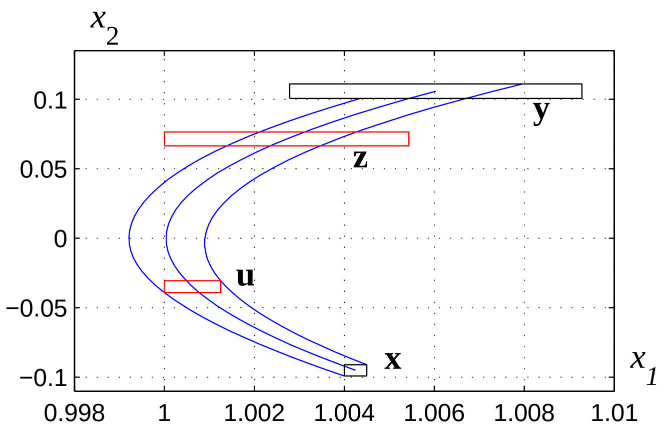

Fig. 1. Two-dimensional PWL system, integration along the tangency

Results of applying the Algorithm 2 for the computation of an enclosure of $\varphi(0.2, \mathbf{x})$ are plotted in Fig. 1. The intermediate enclosures $\mathbf{u}$, $\mathbf{z}$, and the final result $\mathbf{y}$ are shown. The set $\mathbf{u} = \varphi_2(s_1, \mathbf{x})$ encloses the solution set at time s_1 when all trajectories are just before intersection with Σ. The rectangle $\mathbf{z} = \varphi_2(s_2, \mathbf{u}) + \Delta$ contains the solution set $\varphi(s_2, \mathbf{u})$ when all trajectories has already passed the tangency area.

In this case $s_1 \approx 0.064$, and one can see that $\mathbf{u} = \varphi_k(s_1, \mathbf{x})$ is a very narrow enclosure of the set of true trajectories. The set $\mathbf{u}$ is relatively large and in consequence the time $s_2 = 0.1044$ needed to pass the tangency for all initial conditions is also large. This results in a considerable overestimation, the diameter of the set $\mathbf{z}$ is much larger than the diameter of the true solution set. After the set $\mathbf{z}$ is found the final result $\mathbf{y} = \varphi_2(0.2 - s_1 - s_2, \mathbf{z})$ is computed using formulas for solutions of linear systems. The diameters of the initial set and the result are $\mathrm{diam}(\mathbf{x}) = (0.0005, 0.008)$, $\mathrm{diam}(\mathbf{y}) = (0.0065, 0.0104)$.

When the diameter of the initial set is reduced to $\mathrm{diam}(\mathbf{x}) = (10^{-5}, 10^{-5})$ the time s_2 needed to pass along the tangency is significantly smaller $s_2 = 0.0215$ and in consequence the overestimation is also reduced. In this case the diameter of the result is $\mathrm{diam}(\mathbf{y}) = (6.63 \cdot 10^{-5}, 1.92 \cdot 10^{-5})$.

III. Rigorous analysis of the Chua's circuit spiral attractor

Let us consider the Chua's circuit [1], a third-order PWL system described by the following set of ordinary differential equations:

$$\begin{aligned} C_1\dot{x}_1 &= (x_2 - x_1)/R - g(x_1),\\ C_2\dot{x}_2 &= (x_1 - x_2)/R + x_3, \quad (17)\\ L\dot{x}_3 &= -x_2 - R_0x_3, \end{aligned}$$

where $g(z) = G_bz + 0.5(G_a - G_b)(|z+1| - |z-1|)$ is a three segment PWL characteristics.

For this system there are three linear regions $R_1 = \{x \in \mathbb{R}^3\colon x_1 \le -1\}$, $R_2 = \{x\colon |x_1| \le 1\}$ and $R_3 = \{x\colon x_1 \ge 1\}$ separated by planes $\Sigma_1 = \{x\colon x_1 = -1\}$ and $\Sigma_2 = \{x\colon x_1 = 1\}$.

The circuit is studied with the following parameter values (after appropriate parameter rescaling): $C_1 = 1$, $C_2 = 8.3$, $G_a = -3.4429$, $G_b = -2.1849$, $L = 0.06913$, $R = 0.33065$, $R_0 = 0.00036$.

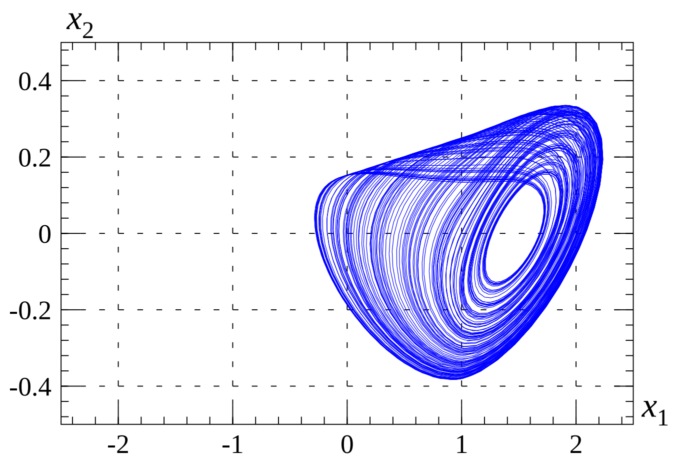

Fig. 2. Computer generated spiral attractor for the Chua's circuit

For the parameter values considered the spiral attractor is observed in computer simulations (compare Fig. 2). Let us note that the attractor does not intersect Σ_1. Trajectories are tangent to Σ_2 at $x_1 = 1$, $x_2 = 1 + RG_a = -0.138394885$. Note that some trajectories turn close to the plane Σ_2, i.e. intersections with Σ_2 are not always transversal. This means that the return map associated with the plane Σ_2 is not continuous over a neighborhood of the attractor.

A. *Return map*

Let P be the return map defined by the plane $S = \{x\colon x_1 = 1.5\}$. This is a reasonable choice, since the attractor intersects this plane transversally (compare Fig. 2), and therefore one can expect that the return map is continuous in a neighborhood of the intersection of the attractor and the plane S.

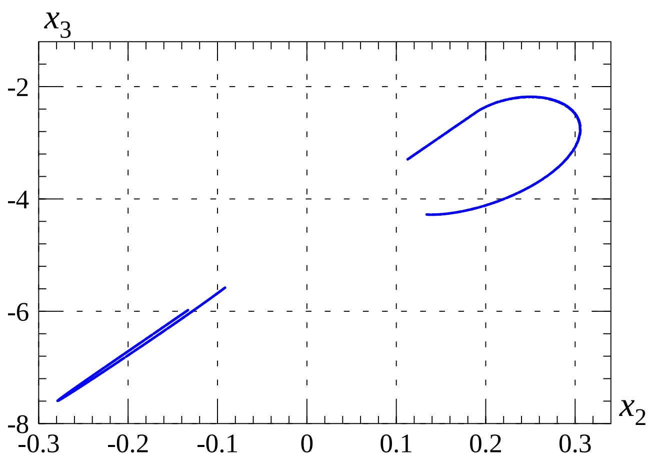

Fig. 3. A trajectory of the return map P defined by $S = \{x\colon x_1 = 1.5\}$

A 10000 points trajectory of the return map is shown in Fig. 3. The plot is composed of two separated parts corresponding to different directions with which trajectories of the continuous system intersect the plane S.

B. *Comparison of algorithms*

In order to compare the two algorithms presented in Section II let us compute an enclosure for the image of the interval vector $\mathbf{x} = (1.5, -0.2212^{2}_{0}, -6.89^{80}_{78})$ under the return map P. The results obtained are $\mathbf{y} = (1.5, 0.2^{872}_{799}, -2.^{320}_{233})$ for the Algorithm 1 and $\mathbf{y} = (1.5, 0.283^{961}_{104}, -2.2^{815}_{716})$ for the Algorithm 2. The second algorithm produces a much narrower enclosure, with the diameter more than 8 times smaller than for the first algorithm.

For $\mathbf{x} = (1.5, -0.282236^{4}_{3}, -7.6119^{6}_{5})$ the difference in performance is even larger. The second algorithm produces the result 80 times smaller in each direction.

For $\mathbf{x} = (1.5, -0.16^{3}_{2}, -6.2^{66}_{52})$ the first algorithm fails. This is due to the fact that some trajectories based at $\mathbf{x}$ are tangent to the plane Σ_2. The second algorithms returns the result $\mathbf{y} = (1.5, 0.1^{945}_{857}, -2.^{509}_{385})$.

Let us now compare the two algorithms for the computation of longer trajectories. Let us choose $x = (1.5, -0.2212, -6.8978)$ as the initial point. Results on finding enclosures of the trajectory $\varphi(t, x)$ are shown in Fig. 4. For each algorithm the diameter of the interval vector returned as a function of the integration time is plotted.

For the trajectory $\varphi(t, x)$ intersections with the plane Σ_2 happen approximately at $t = 0.8$, $t = 2.9$, $t = 6.9$, $t = 11.9$, $t = 28.3$, $t = 29.6$, $t = 33.9$, $t = 37.3$, $t = 41.2$, and so on. One can clearly see that the intersection times correspond to sudden increases in the size of the result for the Algorithm 1. This is due to the fact that at each intersection the Algorithm 1 computes the times when all trajectories are before and after intersection. As one can see, this leads to a huge overestimation. The Algorithm 2 handles intersections via integration of perturbed dynamical systems and as one can see this approach reduces overestimation.

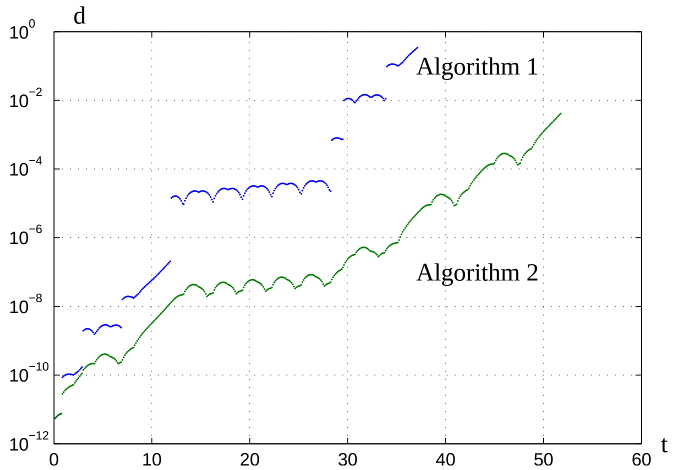

Fig. 4. Comparison of the two algorithms for finding enclosures of trajectories. The diameter d of the result as a function of the integration time t

From the results presented in this section it follows that in general one should use the Algorithm 2 also to handle the transversal case.

C. *A trapping region*

Construction of a trapping region containing the attractor is often the first step in the analysis of dynamical systems, since it limits the interesting behavior to a bounded set. We say that a set Ω is a trapping region for the map P if it is positively invariant under the action of this map, i.e. $P(x) \in \Omega$ for every $x \in \Omega$. Clearly, each trajectory based in a trapping region Ω stays in it forever.

We will prove that there exist a trapping region for the return map P defined previously. The trapping region found encloses the attractor observed numerically. Let us stress that this would not be possible using solely the Algorithm 1.

The procedure starts with the construction of a candidate set. A candidate is constructed using the following two properties. First, it should enclose an observed attractor, so that the dynamics is captured by the set selected. Second, the image of its border (computed non-rigorously for now) should be enclosed within this set.

The set Ω selected as a candidate is shown in Fig. 5. It has been constructed by drawing two polygons enclosing a numerically observed trajectory and then manually adjusting positions of its corners so that $P(x_k) \in \Omega$, where x_k for $k = 1, 2, \ldots, p$ are points chosen along the border of Ω. For now the evaluation of P is not performed rigorously. The set Ω is composed of two polygons Q_1, Q_2. The definition of Ω is given in the Appendix.

In order to prove that Ω is a trapping region one has to show that $P(x) \in \Omega$ for each $x \in \Omega$. This can be done by

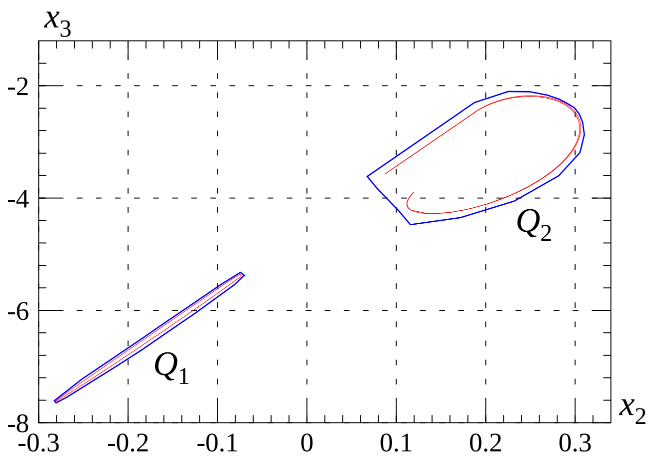

Fig. 5. The set Ω and the image of its border computed non-rigorously

covering the set Ω by a number of interval vectors $\mathbf{x}_k$ (called also boxes), computing enclosures $\mathbf{y}_k$ of images $P(\mathbf{x}_k)$ and verifying that $\mathbf{y}_k \subset \Omega$ for each k. Since it is not known a priori what should be the size of covering rectangles, and perhaps using a uniform size is not the best choice, in practice one uses a subdivision technique. First, interval vectors covering one polygon each are found. The interval vectors are added to the list of boxes to be verified. For each box in the list one computes its image under P. If the image can be found, and it is enclosed in Ω then the box is removed from the list. Otherwise, the box is split into smaller boxes and is removed from the list. Smaller boxes which have non-empty intersection with the set Ω are added to the list, while other boxes are skipped. Computations are continued until the list of boxes is empty. On completion of the procedure, the proof is finished.

It is possible to speed up the procedure described above by using the fact that the map P is one-to-one, which is a consequence of the uniqueness of solutions of the dynamical system considered. Instead of verifying that $P(\Omega) \subset \Omega$ it is sufficient to verify that (a) the image of the border $\partial\Omega$ is enclosed in Ω and (b) the map P is well defined on Ω. If these two conditions hold then from the uniqueness of solutions and continuity of P it follows immediately that $P(x) \in \Omega$ for each $x \in \Omega$. For more details see [4]. Both conditions can be verified using the subdivision technique. For the first condition one starts with a set of boxes each covering a single edge of $\partial\Omega$ and the procedure is continued in the same way as before. For the existence condition the difference is that one only checks if the image $\mathbf{y}_k$ of a box $\mathbf{x}_k$ can be computed, and the enclosure condition $\mathbf{y}_k \subset \Omega$ is not verified. It will be shown below that this modification drastically reduces the computation time.

Using the procedure described above, it has been verified that $P(\partial\Omega) \subset \Omega$. The set of boxes $\mathbf{x}_k$ covering $\partial\Omega$ for which the condition $P(\mathbf{x}_k) \subset \Omega$ has been verified, are shown in Fig. 6(a), while enclosures of their images are shown in Fig. 6(b). There are 3694 boxes in the covering. The total number of boxes for which the image was evaluated was 4363, and the total computation time was 177 seconds on a single-core 2.40 GHz computer. It is interesting to note that only for two out of 3694 boxes the intersection with the plane Σ_2 was not transversal. These boxes are plotted in a different color in Fig. 6. Trajectories based at the remaining boxes have either two or none intersections with the plane Σ_2 before returning to the plane S.

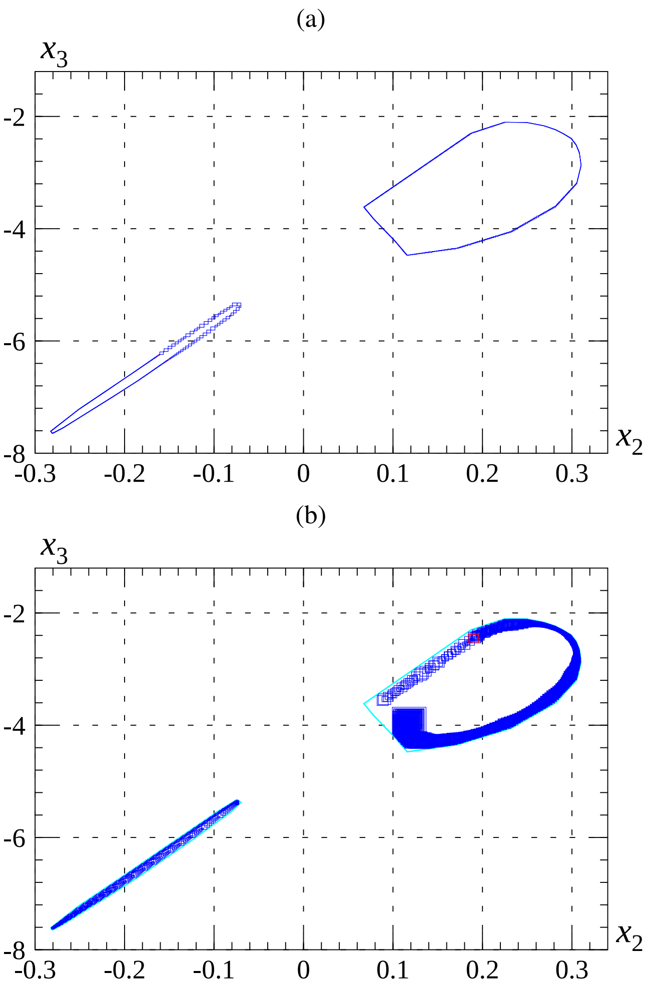

Fig. 6. (a) the covering of the set $\partial\Omega$ by 3694 boxes for the proof of the condition $P(\partial\Omega) \subset \Omega$, (b) the image of the covering enclosed in the set Ω

The covering of the set Ω composed of 1893 boxes for the proof of the existence of P is shown in Fig. 7. The total number of evaluations of the map P was 2670 and the computation time was 772 seconds. Therefore, the total time necessary to prove that $P(\Omega) \subset \Omega$ was less than 16 minutes. It is interesting to note that the time needed to prove that Ω is a trapping region without treating the border and interior differently resulted in a covering composed of 406727 boxes and the computation time of more than 6 hours (approximately 23 times longer).

D. Narrower enclosures of the attractor

The procedure used for the proof of the condition $P(\partial\Omega) \subset \Omega$ can be modified to get a narrower approximation of the attractor. This is achieved by adding an additional condition under which the box should be split — a box is split also if

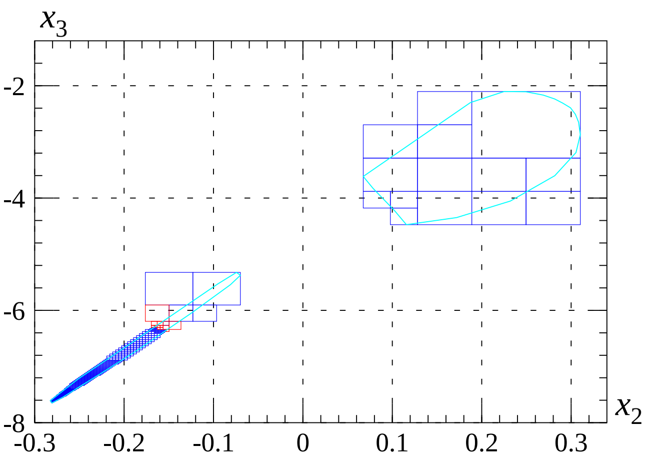

Fig. 7. The covering of the set Ω by 1893 boxes for the proof of existence of $P(\Omega)$

the diameter of its image is above a given threshold. Using the maximum diameter of an image equal to 0.01 required the computation time of 35 minutes and resulted in the covering of $\partial\Omega$ by 39613 boxes. The enclosure of the image of the covering is shown in Fig 8.

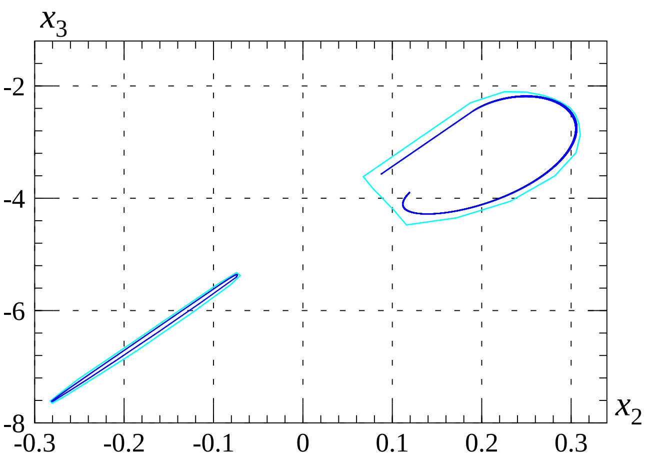

Fig. 8. The enclosure of the image of the set $\partial\Omega$ under the map P

The narrow covering obtained in this way can be further used as a new trapping region. Computing its image allows one to obtain even better approximations of the attractor.

E. Graph representation of the dynamics of P

Once, there is a trapping region for the map P enclosing the chaotic attractor one can study global dynamics of the system over the attractor. One of the frequently used methods is the construction of a directed graph representing the dynamics of the map (compare [12], [13]). Graph representation helps in finding narrow enclosures of the invariant part of the trapping region, helps is locating periodic orbits, and so on.

To construct the graph, first, the trapping region is covered by boxes of a specified size. Here, we use the so-called ε–boxes of the form $\mathbf{v} = [k_1\varepsilon_1, (k_1+1)\varepsilon_1] \times [k_2\varepsilon_2, (k_2+1)\varepsilon_2]$, where $k_{1,2}$ are integer numbers defining the position of the box, and $\varepsilon_{1,2}$ are fixed real numbers defining the ε–box size. These boxes are graph vertices. Next, non-forbidden transitions between boxes are found in order to define graph edges. For each box $\mathbf{v}_k$ enclosure $\mathbf{y}_k$ of its image under P is computed. The boxes $\mathbf{v}_j$ which have non-empty intersection with $\mathbf{y}_k$ define graph edges starting at vertex $\mathbf{v}_k$, i.e. $(\mathbf{v}_k, \mathbf{v}_j)$ is an edge in the graph if and only if $\mathbf{v}_j \cap \mathbf{y}_k \neq \emptyset$, where $\mathbf{y}_k$ is an enclosure of $P(\mathbf{v}_k)$.

The trapping region, constructed in the previous section has been covered by 13446 ε–boxes with $(\varepsilon_1, \varepsilon_2) = (1/2000, 1/200)$ (see Fig. 9(a)). Using the method described above, it has been shown that there are 473848 non-forbidden transitions between the boxes.

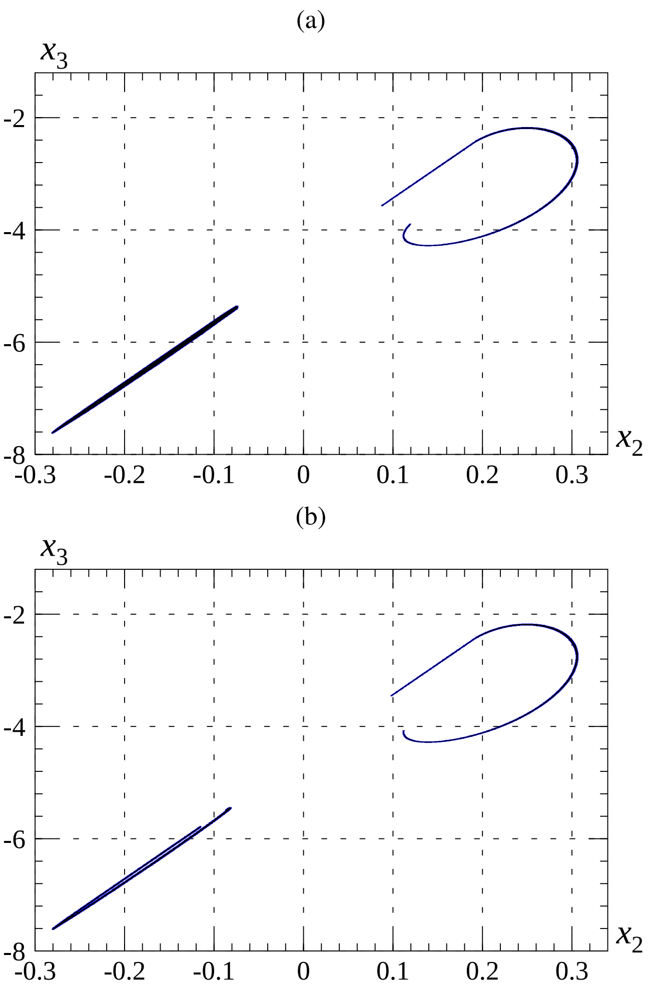

Fig. 9. Graph representation of the dynamics of P using ε–boxes with $(\varepsilon_1, \varepsilon_2) = (1/2000, 1/200)$, (a) initial graph representation with 13446 boxes, (b) reduced representation with 10896 boxes

Interesting dynamics happens in the invariant part of the trapping region. For a trapping region Ω its invariant part under the action of P is defined as $\mathrm{Inv}(\Omega) = \bigcap_{k\geq 0} P^k(\Omega)$.

The graph representation can be reduced if we remove boxes which have empty intersection with the invariant part. A box is outside the invariant part (and hence can be removed from the graph) if there are no edges starting at this box or if there are no edges ending in this box. The procedure is continued until no more boxes can be removed. This procedure leads to a graph with 11124 boxes and 399252 connections.

Further reduction can be obtained by computing narrower enclosures of box images. The simplest method is to use the subdivision technique. A box is divided into several smaller boxes and their images are computed. In this way a graph of 10896 boxes and 352706 connections has been constructed (compare Fig. 9(b)). The number of graph vertices have been reduced by 19%. Note that the finer structure of the covering reveals that the left part has two leaves.

The same procedure has been carried out for ε–boxes with $(\varepsilon_1, \varepsilon_2) = (1/4000, 1/400)$. For the computation of box images a subdivision technique has been used. Each box was split into not more than 32×32 smaller boxes (when the image was small enough the splitting was stopped). The smallest boxes for which images were calculated were of the size $(1/(4000 \cdot 32), 1/(400 \cdot 32))$. The computations took several hours. After removing boxes not belonging to the invariant part a graph with 12679 boxes and 61376 non-forbidden connections has been obtained. The ε–boxes covering the attractor are plotted in Fig. 10. The area of the covering has been reduced by 71%, when compared to the case with larger ε–boxes. Note, that the representation obtained is a very accurate enclosure of the attractor of P shown in Fig. 3.

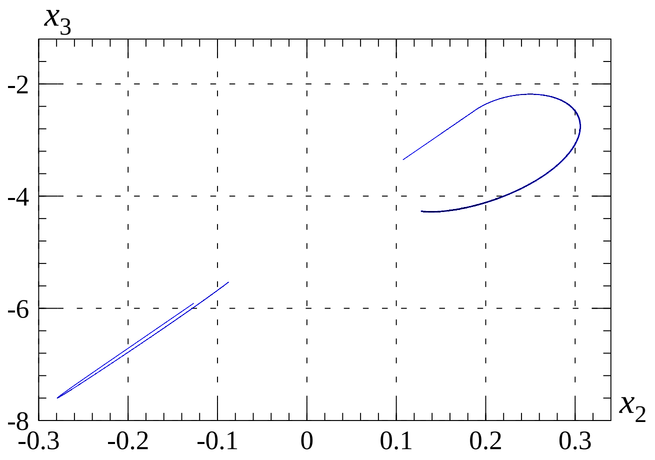

Fig. 10. Graph representation of the dynamics of P using ε–boxes with $(\varepsilon_1, \varepsilon_2) = (1/4000, 1/400)$, there are 12679 boxes and 61376 non-forbidden connections in the graph representation

F. A bound for the average return time

Enclosures for the return times for individual boxes have been found during the construction of the graph. Using this information one can obtain a bound for the return time for the map P over the attractor. The procedure is described below. Let us denote by $\mathbf{t}_k$ an enclosure of return times for points belonging to $\mathbf{v}_k$. The hull of intervals $\mathbf{t}_k$ is enclosed in the interval $\mathbf{T} = [2.345, 6.563]$. It follows that each point belonging to the attractor has the return time belonging to the interval $\mathbf{T}$.

Using the information on non-forbidden connections one can obtain bounds $\mathbf{T}_k$ for the return time of the k-th iterate of P. $\mathbf{T}_k$ can be computed as a union of intervals $\mathbf{t}_{i_1} + \mathbf{t}_{i_2} + \cdots + \mathbf{t}_{i_k}$ over all sequences $(i_1, i_2, \ldots, i_k)$ such that $(\mathbf{v}_{i_1}, \mathbf{v}_{i_2})$, $(\mathbf{v}_{i_2}, \mathbf{v}_{i_3})$,…,$(\mathbf{v}_{i_{k-1}}, \mathbf{v}_{i_k})$ are graph edges. The results are collected in Table I.

k	$\mathbf{T}_k$	$\mathbf{T}_k/k$
1	[2.345,6.563]	[2.345,6.563]
2	[5.338,9.417]	[2.669,4.709]
3	[8.353,12.985]	[2.784,4.329]
4	[11.329,15.996]	[2.832,3.999]
5	[14.339,20.087]	[2.867,4.018]
6	[17.425,23.046]	[2.904,3.841]
7	[20.415,26.910]	[2.916,3.845]
8	[24.582,29.895]	[3.072,3.737]
9	[27.264,33.893]	[3.029,3.766]
10	[31.123,36.870]	[3.112,3.687]
100	[331.76,348.87]	[3.3176,3.4887]
1000	[3338.4,3469.0]	[3.3384,3.4690]
10000	[33404,34670]	[3.3404,3.4670]

TABLE I
BOUNDS $\mathbf{T}_k$ FOR THE RETURN TIME OF THE k-TH ITERATE OF P, BOUNDS $\mathbf{T}_k/k$ FOR THE AVERAGE RETURN TIME

$\mathbf{T}_k/k$ is an enclosure for the average return time for all points belonging to the attractor. For $k = 10000$ we obtain the bound $[3.3404, 3.4670]$ for the average return time (compare Table I).

IV. Conclusion

It has been shown that there exists a trapping region for the return map associated with the Chua's circuit spiral attractor. A graph representation of the dynamics of the spiral attractor has been constructed, and a bound for the average return time has been found.

The algorithm for rigorous integration of piece-wise linear systems necessary to carry out computer assisted proofs has been developed. The algorithm is general in the sense that it also handles the case of trajectories tangent to hyperplanes separating linear regions. It has been shown that the algorithm provides narrower enclosures of the true trajectories also in the case of transversal intersections. The proposed algorithm may help to understand the dynamics of the double-scroll attractor, which also contains trajectories tangent to the C^0-hyperplanes.

The methods can be used without major modifications for the integration of piece-wise smooth systems, i.e. systems with a continuous vector field, where the state space can be divided into regions where the vector field is C^1. The only difference when compared to piece-wise linear systems is that one cannot use exact formulas for the integration in smooth regions and standard techniques for integration of nonlinear systems have to be employed.

Acknowledgment

The author would like to acknowledge fruitful discussions with Prof. P. Zgliczyński.

Appendix
Definition of the trapping region Ω

The trapping region Ω is the union of two convex polygons Q_1, Q_2:

$$\begin{aligned}
Q_1 = (&(-0.28252, -7.60875), (-0.28060, -7.64744),\\
&(-0.26857, -7.54685), (-0.21417, -7.00526),\\
&(-0.18461, -6.70352), (-0.15415, -6.36824),\\
&(-0.12497, -6.05103), (-0.08107, -5.54296),\\
&(-0.06994, -5.37532), (-0.07429, -5.32374),\\
&(-0.09528, -5.53006), (-0.12254, -5.82665),\\
&(-0.15658, -6.19545), (-0.18909, -6.55135),\\
&(-0.22160, -6.90210), (-0.25120, -7.21944)),\\
Q_2 = (&(0.06747, -3.61542), (0.07835, -3.82577),\\
&(0.10256, -4.22553), (0.11576, -4.47425),\\
&(0.17186, -4.34798), (0.23204, -4.05334),\\
&(0.28173, -3.60182), (0.30541, -3.19039),\\
&(0.31028, -2.87016), (0.30833, -2.64891),\\
&(0.30477, -2.50918), (0.29926, -2.39273),\\
&(0.29045, -2.30838), (0.28160, -2.23569),\\
&(0.26880, -2.16689), (0.24946, -2.10743),\\
&(0.22509, -2.10315), (0.18741, -2.29958)).
\end{aligned}$$

References

[1] L. Chua and G. Lin, "Canonical realisation of Chua's circuit family," *IEEE Trans. Circ. Syst.*, vol. CAS–37, no. 7, pp. 885–902, Jul. 1990.

[2] L. Chua, M. Komuro, and T. Matsumoto, "The double scroll family," *IEEE Trans. Circ. Syst.*, vol. CAS–33, pp. 1037–1118, Nov. 1986.

[3] T. Matsumoto, L. Chua, and K. Ayaki, "Reality of chaos in the double scroll circuit: a computer-assisted proof," *IEEE Trans. Circ. Syst.*, vol. CAS–35, no. 7, pp. 909–925, Jul. 1988.

[4] Z. Galias, "Positive topological entropy of Chua's circuit: A computer assisted proof," *Int. J. Bifurcation and Chaos*, vol. 7, no. 2, pp. 331–349, 1997.

[5] R. Lohner, "Enclosing the solutions of ordinary initial and boundary value problems," in *Computerarithmetic, Scientific Computation and Programming Languages*. Stuttgart: Teubner, 1987, pp. 225–286.

[6] Z. Galias, "Counting low-period cycles for flows," *Int. J. Bifurcation and Chaos*, vol. 16, no. 10, pp. 2873–2886, 2006.

[7] ——, "On rigorous study of Poincaré maps defined by piecewise linear systems," in *Proc. IEEE Int. Symposium on Circuits and Systems, ISCAS'05*, Kobe, Japan, May 2005, pp. 3407–3410.

[8] R. Moore, *Interval Analysis*. Englewood Cliffs, NJ: Prentice Hall, 1966.

[9] E. Hairer, S. Nørsett, and G. Wanner, *Solving ordinary differential equations. I. Nonstiff problems*. New York: Springer Verlag, 1993.

[10] P. Zgliczyński and T. Kapela, "A Lohner-type algorithm for control systems and ordinary differential inclusions," *Discrete and Continuous Dynamical Systems B*, vol. 11, pp. 365–385, 2009.

[11] Z. Galias, "On rigorous integration of piece-wise linear continuous systems," in *Proc. IEEE Int. Symposium on Circuits and Systems, ISCAS'11*, Rio de Janeiro, May 2011, pp. 1339–1342.

[12] M. Dellnitz, A. Hohmann, O. Junge, and M. Rumpf, "Exploring invariant sets and invariant measures," *Chaos: and Interdisciplinary Journal of Nonlinear Science*, vol. 7, no. 2, pp. 221–228, 1997.

[13] Z. Galias, "Interval methods for rigorous investigations of periodic orbits," *Int. J. Bifurcation and Chaos*, vol. 11, no. 9, pp. 2427–2450, 2001.