Simulations of Memristors using the Poincaré Map Approach

Zbigniew Galias, *Senior Member, IEEE*

***Abstract*—In many memristor models the internal variable is related to the width of a conductive layer and is therefore limited by the physical dimensions of the device. The necessity of keeping the internal variable within allowable limits may introduce discontinuities in right hand sides of differential equations describing dynamics of circuits with memristors. Such discontinuities are difficult to handle using standard numerical integration methods. An approach based on the concept of a Poincaré map is proposed to solve these difficulties. Two examples are discussed to show the usefulness of the proposed technique.**

***Index Terms*—memristor, numerical integration methods, Poincaré map.**

I. Introduction

In several memristor models, the internal variable represents the width of the region with higher (or lower) concentration of dopants and is limited by the physical dimensions of the device [1], [2], [3], [4], [5]. In these models it is usually assumed that the internal variable w belongs to the interval $[0, D]$, where D is the physical width of the device. To make sure that $w \in [0, D]$ the concept of a window function is introduced. The right hand side of the equation defining the dynamics of w is multiplied by a function which is zero outside the interval $[0, D]$. This may introduce discontinuities in right hand sides of equations defining the behavior of circuits with memristors. Standard numerical integration are not designed to handle such equations. Due to numerical errors introduced by integration methods, it may happen that w gets outside the allowed interval $[0, D]$ and stays there for ever. None of the approaches which have been proposed to solve this problem is completely satisfactory.

In this work, the concept of Poincaré map is proposed to solve the problem of discontinuities in simulations of circuit with memristors. Ordinary differential equations (ODEs) with discontinuous right hand sides (RHSs) are usually referred to as Filippov systems [6]. Systems with vector fields which have discontinuous higher derivatives are called nonsmooth. In the proposed approach, the state space is divided into regions where the dynamics is smooth. Instead of integrating a single ODE with a discontinuous (or nonsmooth) RHS, a sequence of Poincaré maps is evaluated. Poincaré maps are defined by surfaces in which the original ODE is discontinuous or nonsmooth. A preliminary version of this work has been presented in [7].

This work was supported by the National Science Centre, Poland, grant no. 2015/17/B/ST7/03763.

The author is with the Department of Electrical Engineering, AGH University of Science and Technology, al. Mickiewicza 30, 30–059, Kraków, Poland, (e-mail: galias@agh.edu.pl).

A simplified version of this approach has been introduced in [8]. The authors propose to use the 'Events' option with the standard integration method `ode23` in the MATLAB environment to accurately detect time instants when the internal variable of a memristor reaches a boundary. This idea is applied in [8] to a sinusoidally driven series connection of a memristor and a linear resistance.

An alternative approach to analyze a class of memristor circuits is the flux–charge analysis method [9], [10]. In this method vector fields describing circuits with memristors become smoother and in some cases the problem with discontinuous vector fields may disappear.

The layout of the manuscript is as follows. In Section II several memristor models and window functions are recalled. The Poincaré map approach is presented in Section III and two simulation examples are discussed in Section IV.

II. Simulations of Circuits with Memristors

An ideal current-controlled memristor [11] is described by the equations $v(t) = R(q)i(t)$ and $\frac{\mathrm{d}q}{\mathrm{d}t} = i$, where q is the charge and $R(q)$ is the resistance of the device. An ideal voltage-controlled memristor is described by $i(t) = G(\varphi)v(t)$ and $\frac{\mathrm{d}\varphi}{\mathrm{d}t} = v$, where $G(\varphi)$ is the conductance of the device controlled by the flux φ.

In [12], the concept of ideal memristors has been extended to what is now referred to as extended memristors. An extended current-controlled memristor is described by

$$v(t) = R(w, i)i(t), \quad \frac{\mathrm{d}w}{\mathrm{d}t} = F(w, i), \tag{1}$$

where $w \in \mathbb{R}^n$ is a vector of internal state variables. In the following, we consider the case $w \in \mathbb{R}$. In the dual case, an extended voltage controlled memristor is described by

$$i(t) = G(w, v)v(t), \quad \frac{\mathrm{d}w}{\mathrm{d}t} = F(w, v), \tag{2}$$

where $G(w, v)$ is the conductance of the device.

In the seminal paper [1], the linear ion drift model has been proposed to describe the behavior of certain nanoscale devices:

$$\frac{\mathrm{d}w}{\mathrm{d}t} = \mu_v \frac{R_{\mathrm{on}}}{D} i(t), \tag{3}$$

$$v(t) = \left(R_{\mathrm{on}} \frac{w}{D} + R_{\mathrm{off}} \left(1 - \frac{w}{D} \right) \right) i(t). \tag{4}$$

The linear ion drift model belongs to the class of ideal generic current-controlled memristors (compare [13]). It is assumed that $w \in [0, D]$, where D is the width of the device. When w reaches D (or 0) it cannot grow (decrease) any more. In

simulations, to achieve this effect one may multiply the right hand side of (3) by a window function $f(w)$ which takes the value zero for $w \notin [0, D]$. Various window functions have been proposed in the literature. The simplest one is the ideal rectangular window function $f_{\mathrm{R}}(w)$ for which $f_{\mathrm{R}}(w) = 1$ when $w \in [0, D]$ and $f_{\mathrm{R}}(w) = 0$ when $w \notin [0, D]$ (see Fig. 1(a)). The ideal rectangular window function can be defined as

$$f_{\mathrm{R}}(w) = \mathcal{H}(w)\mathcal{H}(D - w), \tag{5}$$

where $\mathcal{H}(x)$ is the unit step function satisfying $\mathcal{H}(x) = 1$ for $x \geq 0$ and $\mathcal{H}(x) = 0$ for $x < 0$.

In theory, this window function limits the value of the internal variable w to the interval $[0, D]$. The problem appears when one applies standard numerical integration methods to simulate the behavior of the element. The ideal rectangular window function introduces discontinuities in the definition of the derivative $\dot{w}$ which are difficult to handle numerically. In numerical simulations it may happen that w leaves the interval $[0, D]$. This is a consequence of using non-zero time steps and errors introduced by numerical integration methods. Once $w \notin [0, D]$ the derivative $\dot{w}$ becomes zero and the memristor holds its state forever.

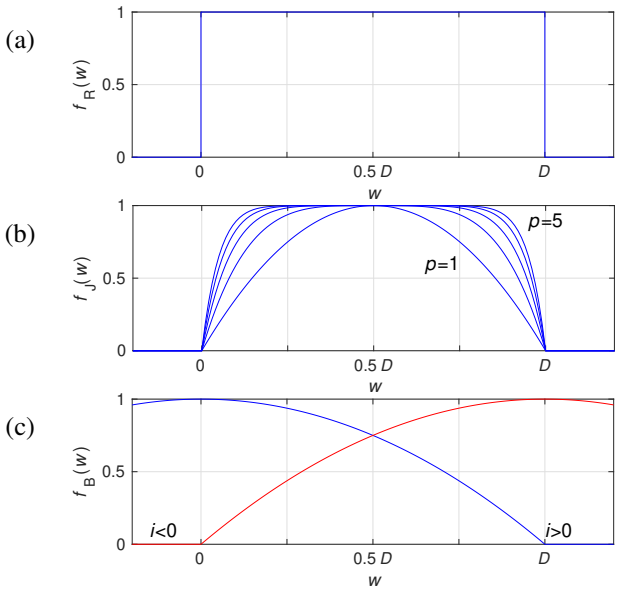

Fig. 1. Window functions: (a) ideal rectangular window function, (b) Joglekar window function for $p = 1, 2, \ldots, 5$, (c) Biolek window function for $p = 1$.

In [3], the Joglekar window function has been proposed:

$$f_{\mathrm{J}}(w) = 1 - (2w/D - 1)^{2p}\,, \tag{6}$$

where p is a positive integer. For this window function the derivative $\dot{w}$ close to the window endpoints decreases to zero in a continuous way (compare Fig. 1(b)). This window function may suffer from similar problems as the previous one. If we set the memristor to the state $R = R_{\mathrm{on}}$ or $R = R_{\mathrm{off}}$ which corresponds to $w = D$ and $w = 0$, respectively, then no external stimulus can change its state. The memristor behaves as a standard resistor. In theory, from the initial state $w \in (0, D)$, the lower (upper) boundary state $w = 0$ ($w = D$) may only be attained asymptotically. However, numerical errors may easily lead to a state $w \notin (0, D)$ from which there is no way back to the normal memristor behavior.

To solve the modeling problems at the boundaries the Biolek window function has been proposed in [2]

$$f_{\mathrm{B}}(w, i) = 1 - (w/D - \mathcal{H}(-i))^{2p}\,. \tag{7}$$

where $\mathcal{H}$ is the unit step function. This window function for the case $p = 1$ is plotted in Fig. 1(c). The result of applying the Biolek window function depends on the sign of the current. For $i > 0$ the RHS of (3) is positive and $f_{\mathrm{B}}(w, i > 0)$ changes from $f_{\mathrm{B}}(0, i > 0) = 1$ to $f_{\mathrm{B}}(D, i > 0) = 0$. The growth rate is maximal at the left endpoint of the window $[0, D]$ and decreases to zero at the right endpoint. It follows that it is easy to escape from the boundary state $w = 0$ and theoretically that there is an infinite time to reach the boundary $w = D$. For $i < 0$ the RHS of (3) is negative and the window function changes from $f_{\mathrm{B}}(0, i < 0) = 0$ to $f_{\mathrm{B}}(D, i < 0) = 1$. It is now easy to escape from the state $w = D$ and in theory the time needed to reach the boundary $w = 0$ is infinite. This model can be used in numerical simulation of electronic circuits with memristors and has been implemented as a SPICE model [2]. From the mathematical point of view, this model can be viewed as an artificial method to solve the problems related to integration of systems with discontinuous right hand sides.

The next model we consider is the boundary condition-based (BC) model [4] defined by

$$\frac{\mathrm{d}x}{\mathrm{d}t} = i_0^{-1} W(x(t))v(t) f_{\mathrm{BC}}(x(t), v(t)), \tag{8}$$

$$i(t) = W(x(t))v(t), \tag{9}$$

where $W(x) = G_{\mathrm{on}}G_{\mathrm{off}}\left(G_{\mathrm{on}} - (G_{\mathrm{on}} - G_{\mathrm{off}})x\right)^{-1}$ is the memductance. The window function f_{BC} is defined as:

$$f_{\mathrm{BC}}(w, v) = \begin{cases} 0, & w = 0 \text{ and } v \leq v_{th,0}, \\ 0, & w = D \text{ and } v \geq -v_{th,1}, \\ 1, & \text{otherwise.} \end{cases} \tag{10}$$

In the following, we consider the case $v_{th,0} = v_{th,1} = 0$. Similarly to the Biolek model, the BC model is designed in such a way that the internal variable x is limited to the interval $[0, D]$ and can easily leave the state $x = 0$ or $x = D$ once the voltage v has a proper sign. We show that using this model with standard numerical integration methods may introduce computational errors.

The last model, which we discuss, is the VTEAM model [14]. The VTEAM model with a linear dependence between the resistance $R(w)$ and the internal variable w is defined as:

$$\frac{\mathrm{d}w}{\mathrm{d}t} = \begin{cases} k_{\mathrm{off}}\left(v/v_{\mathrm{off}} - 1\right)^{\alpha_{\mathrm{off}}} f_{\mathrm{off}}(w), & v > v_{\mathrm{off}}, \\ 0, & v \in [v_{\mathrm{on}}, v_{\mathrm{off}}], \\ k_{\mathrm{on}}\left(v/v_{\mathrm{on}} - 1\right)^{\alpha_{\mathrm{on}}} f_{\mathrm{on}}(w), & v < v_{\mathrm{on}}, \end{cases} \tag{11}$$

$$i(t) = \left(R_{\mathrm{on}} + \frac{w - w_{\mathrm{on}}}{w_{\mathrm{off}} - w_{\mathrm{on}}}(R_{\mathrm{off}} - R_{\mathrm{on}})\right)^{-1} v(t), \tag{12}$$

where $v_{\mathrm{on}} < 0$ and $v_{\mathrm{off}} > 0$ are threshold voltages, $k_{\mathrm{off}} > 0$, $k_{\mathrm{on}} < 0$ and α_{off}, α_{on} are positive integer numbers. The resistance of the device may change only when $v_{\mathrm{off}} < v$ or

$v < v_{\rm on}$. The window functions $f_{\rm off}(w)$, and $f_{\rm on}(w)$ constrain the internal variable w to the interval $[w_{\rm on}, w_{\rm off}]$. We consider the ideal rectangular window function:

$$f_{\rm off}(w) = f_{\rm on}(w) = \mathcal{H}(w - w_{\rm on})\mathcal{H}(w_{\rm off} - w), \qquad (13)$$

where $\mathcal{H}$ is the unit step function.

The right hand side of (11) is discontinuous at $w = w_{\rm on}$ and $w = w_{\rm off}$, and is nonsmooth at $v = v_{\rm on}$ and $v = v_{\rm off}$. It follows that using standard numerical integration method to compute trajectories passing close to the planes $w = w_{\rm on}$, $w = w_{\rm off}$, $v = v_{\rm on}$ and $v = v_{\rm off}$ may lead to numerical errors and produce unexpected results.

III. The Poincaré Map Approach

In this section, we present an approach to compute trajectories of electronic systems containing memristors with bounded internal variables based on the concept of the Poincaré map [15].

Let us first recall the notion of a Poincaré map. We consider a dynamical system in $\mathbb{R}^n$ defined $\dot{x} = F(x,t)$, where $x \in \mathbb{R}^n$. Let us select a subset $\Sigma \subset \mathbb{R}^n$. Σ is usually a plane defined by $g(x) = a_1x_1 + \ldots + a_nx_n - b = 0$. In a more general setting, Σ may be a surface of dimension $n-1$ defined by $g(x) = 0$, where g is a possibly nonlinear function or a union of such surfaces. The Poincaré map $P\colon \mathbb{R}^n \mapsto \Sigma$ is defined as

$$P(x) = \varphi(\tau(x), x), \qquad (14)$$

where $\varphi(t,x)$ is the trajectory of the dynamical system passing through x for $t = 0$ (i.e., $\varphi(0,x) = x$) and $\tau(x) > 0$ is the smallest positive time after which the trajectory intersects Σ, i.e. $\varphi(\tau(x),x) \in \Sigma$ and $\varphi((0,\tau(x)),x) \cap \Sigma = \emptyset$. In the following, $g(x) = 0$ is called the exit condition.

In the first step of analysis, we divide the state space $\mathbb{R}^n$ into m regions $\mathcal{R}_1, \mathcal{R}_2, \ldots, \mathcal{R}_m$ in such a way that in each region the function $F(x,t)$ is smooth. The regions are separated by planes at which the right hand side is discontinuous or nonsmooth. For example, in the case of the linear ion drift model the planes separating smooth regions are $w = 0$ and $w = D$, while for the VTEAM model there are two conditions for which the right hand side is discontinuous ($w = w_{\rm on}$ and $w = w_{\rm off}$) and two conditions for which the right hand side is not smooth ($v = v_{\rm off}$, $v = v_{\rm on}$). Let us denote by $\mathcal{B}_k$ for $k = 1, 2, \ldots, m$ the boundary of the region $\mathcal{R}_k$. The Poincaré map defined by the boundary $\mathcal{B}_k$ is denoted as P_k.

In the Poincaré map approach the dynamical system is integrated within a given region $\mathcal{R}_k$ and at the same time the exit condition $\varphi(t,x) \in \mathcal{B}_k$ is monitored. This is equivalent to the evaluation of the Poincaré map P_k. The computation of a trajectory is carried out as the evaluation of a sequence of Poincaré maps (P_{i_k}) for $k = 0, 1, 2, \ldots$, where the symbol sequence $(i_0, i_1, i_2, \ldots)$ is defined by exit conditions. Let us explain this approach in more detail. We want to compute the trajectory $\varphi(t,x)$ with the initial condition x. Let us denote $x_0 = x$. The initial symbol i_0 is defined by the condition $x_0 \in \mathcal{R}_{i_0}$. In the kth step of the computation procedure ($k = 0, 1, 2, \ldots$) the Poincaré map P_{i_k} at x_k is evaluated and the exit point $x_{k+1} = P_{i_k}(x_k)$ is found. During the evaluation of P_{i_k} the trajectory based at x_k is found using a standard numerical integration method. In each integration step, the exit condition is verified and when it is detected then the return time and the exit point x_{k+1} are calculated. These calculations are carried out using a root finding technique, for example the Newton method. The next symbol i_{k+1} is defined by the condition $x_{k+1} \in \mathcal{R}_{i_{k+1}}$.

In the approach presented above, we always integrate a smooth dynamical system. This way we eliminate unexpected errors which may be introduced by using standard numerical integration methods to handle discontinuous or nonsmooth systems. This is the main advantage of the proposed method. Another advantage is the possibility of reducing the dimension of the system under study. As it is shown in the following section in certain regions the dynamics is defined by less than n variables. This permits faster computations.

Let us note that the Poincaré map approach cannot be used in an arbitrary system with a discontinuous RHS. In some cases the so called sliding motion may exist [6]. This happens if the vector field on both sides of a discontinuity surface points towards it. The effect is that the motion is restricted to the discontinuity surface. However, in memristor models with bounded states these effects are not observed.

There is a number of software packages that are capable of evaluating Poincaré maps. One of them is MATCONT, a MATLAB numerical continuation package for the interactive bifurcation analysis of dynamical systems [16]. In MATLAB, one may also use the 'Events' options in standard integration methods like `ode23` or `ode45` to obtain the required functionality. This approach is explored in the following section. The 'Events' procedure is used to define the exit condition $g(x) = 0$. Events are detected by finding sign crossings of g between steps ($g(x(t_k)g(x(t_{k+1})) < 0$). The event instance t satisfying conditions $g(x(t)) = 0$ and $t \in [t_k, t_{k+1}]$ is found by employing a root finding mechanism applied to a polynomial approximation of the solution $x(t)$ (compare [17]). Note that an event can be missed if there is an even number of sign crossings between steps.

Another software package which has the desired functionality is the CAPD library [18], a collection of C++ modules for numerical studies of dynamical systems. It provides algorithms for the evaluation of Poincaré maps and its derivatives. Intersections with Σ are detected in the same way as in the MATLAB software. The one-dimensional Newton method with a Taylor expansion of the solution is employed to find the return time. In the CAPD library it is also possible to evaluate Poincaré maps in a rigorous way. This is achieved by monitoring derivatives of g, using rigorous integration methods and carrying out all computations in interval arithmetic.

In the remaining part of this section, we discuss how to construct smooth regions for various memristor models.

Let us first consider the linear ion drift model. In this case there are three smooth regions separated by conditions $w = 0$, $w = D$, and $i = 0$. Description of regions $\mathcal{R}_k$ for this model is presented in Table I. For each region, we report its definition $\mathcal{R}_k$, the rate $\dot{w}$ of change of the internal variable, the boundary $\mathcal{B}_k$ presented as the list of exit conditions, and for each exit condition the corresponding next region $\mathcal{R}_l$.

TABLE I
SMOOTH REGIONS FOR THE LINEAR ION DRIFT MODEL.

k	$\mathcal{R}_k$	$\dot{w}$	$\mathcal{B}_k$	$\mathcal{R}_l$
1	$0 < w < D$	$\mu_v \frac{R_{\rm on}}{D} i(t)$	$w = 0$ $w = D$	$\mathcal{R}_2$ $\mathcal{R}_3$
2	$w = 0, i < 0$	0	$i = 0$	$\mathcal{R}_1$
3	$w = D, i > 0$	0	$i = 0$	$\mathcal{R}_1$

Note that $\dot{w} = 0$ for the regions $\mathcal{R}_2$ and $\mathcal{R}_3$. It follows that when solving ODEs in these regions the order is reduced by one. In the regions $\mathcal{R}_2$, and $\mathcal{R}_3$ there is a single exit condition $i = 0$. The Poincaré map for the region $\mathcal{R}_1$ is defined by two planes. From the region $\mathcal{R}_1$ a trajectory may go to one of the regions $\mathcal{R}_2$ and $\mathcal{R}_3$ depending on which of the conditions $w = 0$ or $w = D$ is satisfied at the exit point. From the regions $\mathcal{R}_2$ and $\mathcal{R}_3$ a trajectory returns to the region $\mathcal{R}_1$ when $i(t)$ changes its sign.

From the mathematical point of view the BC window is equivalent to the rectangular window and hence the Poincaré map approach leads to the same set of smooth regions. For the Biolek model one may also use the set of smooth regions presented in Table I. Using the Poincaré map approach in this case helps to avoid errors caused by using standard numerical methods for systems with bounded variables.

From the theoretical point of view applying the Poincaré map approach to the Joglekar window does not make sense. In this case the regions $w = 0$, $w \in (0, D)$, and $w = D$ are invariant. A trajectory initiated in one of the regions cannot enter another region. Leaving the region $w \in (0, D)$ may be only a results of numerical errors. On the other hand, to permit an escape from the regions $w = 0$ or $w = D$ one should change the derivative at the borders to some nonzero values, as it is done in the Biolek window.

Finally, let us consider the VTEAM model. Description of smooth regions for this model is presented in Table II. In this case, there are five smooth regions separated by the conditions: $w = w_{\rm off}$, $w = w_{\rm on}$, $v = v_{\rm off}$, and $v = v_{\rm on}$. In the last three regions the derivative $\dot{w}$ is zero. The transition table between regions is more complex than for the previous models.

TABLE II
SMOOTH REGIONS FOR THE VTEAM MODEL.

k	$\mathcal{R}_k$	$\dot{w}$	$\mathcal{B}_k$	$\mathcal{R}_l$
1	$v > v_{\rm off}$ $w < w_{\rm off}$	$k_{\rm off}\,(v/v_{\rm off} - 1)^{\alpha_{\rm off}}$	$v = v_{\rm off}$ $w = w_{\rm off}$	$\mathcal{R}_3$ $\mathcal{R}_4$
2	$v < v_{\rm on}$ $w > w_{\rm on}$	$k_{\rm on}\,(v/v_{\rm on} - 1)^{\alpha_{\rm on}}$	$v = v_{\rm on}$ $w = w_{\rm on}$	$\mathcal{R}_3$ $\mathcal{R}_5$
3	$v_{\rm on} < v < v_{\rm off}$ $w_{\rm on} < w < w_{\rm off}$	0	$v = v_{\rm off}$ $v = v_{\rm on}$	$\mathcal{R}_1$ $\mathcal{R}_2$
4	$v > v_{\rm on}$, $w = w_{\rm off}$	0	$v = v_{\rm on}$	$\mathcal{R}_2$
5	$v < v_{\rm off}$, $w = w_{\rm on}$	0	$v = v_{\rm off}$	$\mathcal{R}_1$

IV. SIMULATION EXAMPLES

In this section, we present simulation results obtained using standard numerical integration methods and the Poincaré map approach. All the simulations are carried out in the MATLAB environment.

As a first example, let us consider a series connection of a memristor and a linear resistor driven by a sinusoidal waveform. We assume that the memristor is described by the VTEAM model (11) with the ideal rectangular window function (13) and the linear dependence between the resistance and the internal variable (12).

The parameters of the VTEAM model are $\alpha_{\rm off} = \alpha_{\rm on} = 3$, $v_{\rm on} = -0.20$ V, $v_{\rm off} = 0.15$ V, $R_{\rm on} = 100\ \Omega$, $R_{\rm off} = 1000\ \Omega$, $k_{\rm on} = -8{\times}10^{-6}$ m/s, $k_{\rm off} = 4{\times}10^{-6}$ m/s, $w_{\rm on} = 0$, $w_{\rm off} = 10$ nm. The linear resistance is $R = 200\ \Omega$. The input voltage is $v_{\rm in}(t) = V_{\rm in}\sin(2\pi t/T)$, where $V_{\rm in} = 0.35$ V and $T = 10$ ms. As the initial value we use $w(0) = 6$ nm.

The dynamical system under study can be described by the following set of equation

$$\frac{{\rm d}w}{{\rm d}t} = \begin{cases} k_{\rm off}\,(v/v_{\rm off} - 1)^{\alpha_{\rm off}}, & v > v_{\rm off}, w \in [w_{\rm on}, w_{\rm off}], \\ k_{\rm on}\,(v/v_{\rm on} - 1)^{\alpha_{\rm on}}, & v < v_{\rm on}, w \in [w_{\rm on}, w_{\rm off}], \\ 0, & \text{otherwise}, \end{cases}$$
$$v(t) = R_m(w(t))v_{\rm in}(t)/(R_m(w(t)) + R), \qquad (15)$$
$$R_m(w) = R_{\rm on} + (R_{\rm off} - R_{\rm on})(w - w_{\rm on})/(w_{\rm off} - w_{\rm on}).$$

First, we present solutions obtained using the `ode45` procedure from the MATLAB package. The right hand side of the derivative $\dot{w}$ in (15) is defined as the procedure `vtRHS` (see the Appendix).

The results obtained by applying the procedure `ode45(@vtRHS,t,ws,op,p)` with the integration time `t=[0,0.03]`, the memristor initial condition `ws=6e-9`, and the options `op=odeset('MaxStep',1e-4)` are shown in Fig. 2(a). One can see that at a certain point due to numerical errors introduced by the integration procedure, the trajectory reaches the region $w > w_{\rm off}$ and stays there until the end of the integration time.

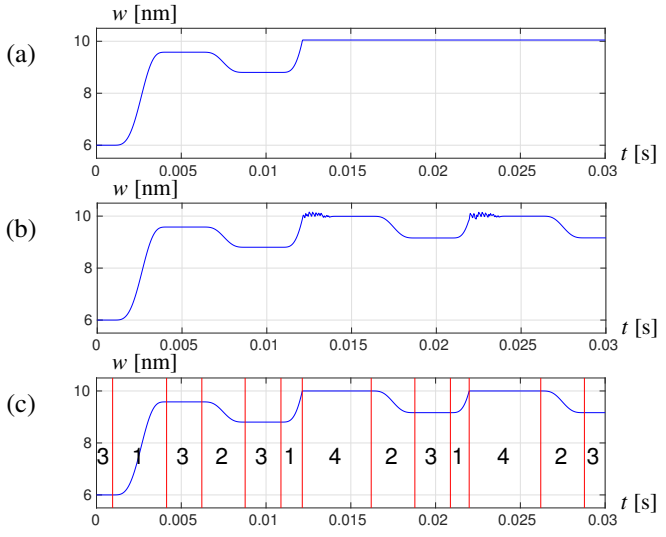

Fig. 2. Solutions for the system (15) found using various approaches; (a) the `ode45` procedure with the RHS being zero when $w \notin [w_{\rm on}, w_{\rm off}]$, (b) the `ode45` procedure with the RHS forcing solutions to converge to the interval $[w_{\rm on}, w_{\rm off}]$, (c) the Poincaré map approach (the `ode45` procedure with the `events` option).

A common solution to overcome this problem is to force the variable w to come back to the interval $[w_{\rm on}, w_{\rm off}]$ by modifying the vector field outside $[w_{\rm on}, w_{\rm off}]$ to push trajectories towards this region. This can be done by adding the following two lines of code:

```
if w>p.woff; rhs=-1e-6; end
if w<0; rhs=+1e-6; end
```

to the procedure `vtRHS`. The results are shown in Fig. 2(b). One can see that this solutions works better. The variable w is not stuck in the region $w > w_{\rm off}$. However, one can see high frequency oscillations close to the border $w = w_{\rm off}$. These oscillations are a consequence of the fact that the vector field (15) is discontinuous at $w = w_{\rm off}$.

Let us now apply the Poincaré map approach. For the VTEAM model there are five smooth regions. The dynamics for these regions is defined by $\dot{w} = k_{\rm off}\,(v/v_{\rm off} - 1)^{\alpha_{\rm off}}$ for $\mathcal{R}_1$ ($v > v_{\rm off}$, $w < w_{\rm off}$), $\dot{w} = k_{\rm on}\,(v/v_{\rm on} - 1)^{\alpha_{\rm on}}$ for $\mathcal{R}_2$ ($v < v_{\rm on}$, $w > w_{\rm on}$) and $w(t) = \text{const}$ for $\mathcal{R}_3$, $\mathcal{R}_4$, and $\mathcal{R}_5$. The definition of the derivative $\dot{w}$ for these five regions is given as the procedure `vtRHSR` (see the Appendix).

In order to apply the Poincaré map approach in the MATLAB environment, we also need to provide a function defining conditions to stop integration. This can be achieved using the 'Events' option. The corresponding function is given as the procedure `vtEvents` (see the Appendix). Integration using the Poincaré map approach is implemented as the procedure `vtIntegrate` (see the Appendix).

The results of applying the Poincaré map approach are shown in Fig. 2(c). Changes of smooth regions are denoted as red vertical lines in the time plot. Note that the trajectory is not stuck in the region $w > w_{\rm off}$ and high frequency oscillations are not observed. This example shows that one can accurately simulate the behavior of a memristor with bounded internal states without resorting to artificial modifications of a window function.

As a second example let us consider a memristor based oscillator [4] shown in Fig. 3. This oscillator is derived from the Chua's circuit by replacing the Chua's diode with the parallel connection of a negative conductance and two memristors connected in anti-parallel. The dynamics of the circuit is given by

$$\begin{aligned}
\dot{x}_1 &= -\alpha G^{-1}(W(x_4) + W(x_5) + G_{\rm N2})x_1 - \alpha x_3,\\
\dot{x}_2 &= \gamma x_2 + x_3,\\
\dot{x}_3 &= \beta(x_1 - x_2 - x_3),\\
\dot{x}_4 &= i_0^{-1} W(x_4) x_1 f(x_4, x_1),\\
\dot{x}_5 &= -i_0^{-1} W(x_5) x_1 f(x_5, -x_1),
\end{aligned} \tag{16}$$

where x_1 and x_2 are the voltages across the capacitors C_1 and C_2, i_L is the inductor current, $x_3 = -i_L/G$, $x_4, x_5 \in [0, 1]$ are dimensionless internal variables of memristors m_1 and m_2, $\alpha = C_2/C_1$, $\beta = C_2/(LG^2)$, $\gamma = -G_{N1}/G$, $t_0 = C_2/G$ is the time scale and i_0 is the constant defining the rate of ionic motion. $W(x) = G_{\rm on}G_{\rm off}/(G_{\rm on} - (G_{\rm on} - G_{\rm off})x)$ is the memductance and $f(x, v)$ is a window function. For more details see [4].

We consider the system (16) with the following parameter values: $G = 3.3$ mS, $G_{N1} = -0.4$ mS, $G_{N2} = -1.2$ mS, $\alpha = 0.74$, $\beta = 0.0333$, $\gamma = 0.12$, $G_{\rm off} = 0.06$ mS, $G_{\rm on} = 1.9$ mS, and $i_0 = 8.9189$. We consider the initial conditions $x_1 = 0.006$, $x_2 = 0.02$, $x_3 = -0.3$, $x_4 = x_5 = 0$.

Let us first consider the boundary condition-based model

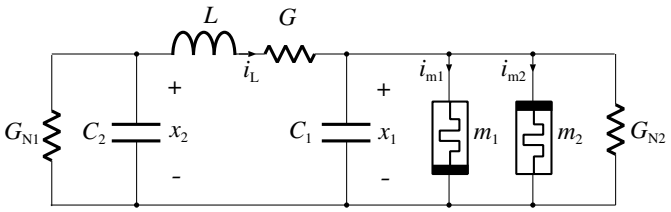

Fig. 3. Memristor based oscillator.

with the window function defined as

$$f_{\rm BC}(x, v) = \begin{cases} 0, & x = 0 \text{ and } v \le 0,\\ 0, & x = 1 \text{ and } v \ge 0,\\ 1, & \text{otherwise.} \end{cases} \tag{17}$$

The window function is zero only if $x = 0$ and $\dot{x} < 0$ or $x = 1$ and $\dot{x} > 0$. It follows that the window function (17) is mathematically equivalent to the ideal rectangular window function $f_{\rm R}(x) = \mathcal{H}(x)\mathcal{H}(1 - x)$, and in case of error-free computations the results are the same.

The RHS of (16) with the BC model is given as the procedure `moRHS` (see the Appendix).

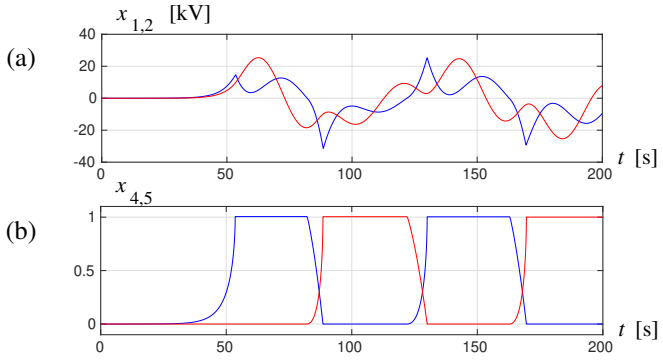

Fig. 4. Solutions for the memristor-based oscillator found using the `ode45` procedure with the maximum time step $\tau = 0.1$; (a) voltages $x_1(t)$, $x_2(t)$, (b) internal states $x_4(t)$ and $x_5(t)$ of memristors m_1 and m_2.

Fig. 4 shows the solution of the system (16) obtained using the procedure `ode45(@moRHS,[0 200],xs,op,p)` with the initial conditions `xs=[0.006,0.02,-0.3,0,0]'` and options `op=odeset('MaxStep',1e-1)` in which the maximum integration time step is set to $\tau = 0.1$.

Let us now assess the accuracy of the results obtained. The accuracy in the `ode45` procedure can be controlled using various options, for example `AbsTol`, `RelTol`, or `MaxStep`. To control the accuracy, we will use the parameter `MaxStep` defining the maximum allowed time step because it has a more clear interpretation. Fig. 5 shows results obtained using different values of the maximum time step $\tau \in \{0.01, 0.001, 10^{-4}, 10^{-5}, 5 \times 10^{-6}\}$. One can see that the solutions are close to each other for $t < 80$ but differ considerably for larger t. The last two plots cannot be distinguished visually. Further decrease of the maximum time step does not change the results obtained for the integration time $\tau = 200$. From Fig. 5 one can also conclude that the accuracy of the solutions obtained for $\tau \le 10^{-4}$ is low.

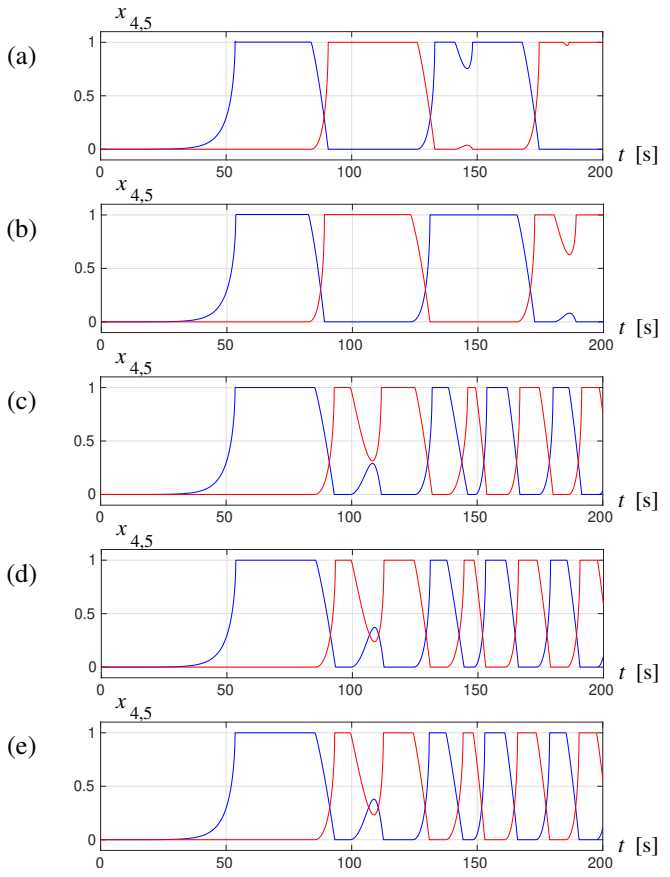

Fig. 5. Internal states $x_4(t)$ and $x_5(t)$ of memristors m_1 and m_2 for solutions found using the `ode45` procedure with various maximum time steps; (a) $\tau = 10^{-2}$, (b) $\tau = 10^{-3}$, (c) $\tau = 10^{-4}$, (d) $\tau = 10^{-5}$, (e) $\tau = 5\times10^{-6}$.

Let us now consider the Poincaré map approach. There are seven smooth regions for the system (16). Their description is presented in Table III. For each region $\mathcal{R}_k$, we report the conditions defining the region and the formulas for $\dot{x}_4$ and $\dot{x}_5$ valid in this region (formulas for other variables are the same in each region). The fourth column of Table III contains lists of exit conditions. For each exit condition the corresponding next region $\mathcal{R}_l$ is listed in the last column. Note that only in the first region the ODE has order five. The order is decreased to four for regions $\mathcal{R}_2$, $\mathcal{R}_3$, $\mathcal{R}_4$, $\mathcal{R}_5$ and to three for the last two regions.

The right hand side of (16) for the Poincaré based approach is given as the function `moRHSR` (see the Appendix). The function defining exit conditions and the integration procedure are given in the appendix as the procedures `moEvents` and `moIntegrate`, respectively.

The results of integrating the system (16) using the Poincaré map approach with the maximum time step $\tau = 0.1$ are shown in Fig. 6. Moments at which there is a change of a smooth region are denoted as gray vertical lines. The plot in Fig. 6(b) is very similar to plots shown in Fig. 5(d,e). The results obtained for $\tau = 0.001$ are visually indistinguishable from the results obtained for $\tau = 0.1$ and are not shown for the sake of brevity. In the following, the results obtained for $\tau = 0.001$ are regarded as the reference values.

TABLE III
SMOOTH REGIONS FOR THE MEMRISTOR BASED OSCILLATOR.

k	$\mathcal{R}_k$	$\dot{x}_4$, $\dot{x}_5$	$\mathcal{B}_k$	$\mathcal{R}_l$
1	$x_4 \in (0,1)$ $x_5 \in (0,1)$	$\dot{x}_4 = i_0^{-1} W(x_4) x_1 F(x_4, x_1)$ $\dot{x}_5 = -i_0^{-1} W(x_5) x_1 F(x_5, -x_1)$	$x_4 = 0$ $x_4 = 1$ $x_5 = 0$ $x_5 = 1$	$\mathcal{R}_2$ $\mathcal{R}_3$ $\mathcal{R}_4$ $\mathcal{R}_5$
2	$x_4 = 0$ $x_5 \in (0,1)$ $x_1 < 0$	$\dot{x}_4 = 0$ $\dot{x}_5 = -i_0^{-1} W(x_5) x_1 F(x_5, -x_1)$	$x_1 = 0$ $x_5 = 1$	$\mathcal{R}_1$ $\mathcal{R}_6$
3	$x_4 = 1$ $x_5 \in (0,1)$ $x_1 > 0$	$\dot{x}_4 = 0$ $\dot{x}_5 = -i_0^{-1} W(x_5) x_1 F(x_5, -x_1)$	$x_1 = 0$ $x_5 = 0$	$\mathcal{R}_1$ $\mathcal{R}_7$
4	$x_4 \in (0,1)$ $x_5 = 0$ $x_1 > 0$	$\dot{x}_4 = i_0^{-1} W(x_4) x_1 F(x_4, x_1)$ $\dot{x}_5 = 0$	$x_1 = 0$ $x_4 = 1$	$\mathcal{R}_1$ $\mathcal{R}_7$
5	$x_4 \in (0,1)$ $x_5 = 1$ $x_1 < 0$	$\dot{x}_4 = i_0^{-1} W(x_4) x_1 F(x_4, x_1)$ $\dot{x}_5 = 0$	$x_1 = 0$ $x_4 = 0$	$\mathcal{R}_1$ $\mathcal{R}_6$
6	$x_4 = 0$ $x_5 = 1$ $x_1 < 0$	$\dot{x}_4 = 0$ $\dot{x}_5 = 0$	$x_1 = 0$	$\mathcal{R}_1$
7	$x_4 = 1$ $x_5 = 0$ $x_1 > 0$	$\dot{x}_4 = 0$ $\dot{x}_5 = 0$	$x_1 = 0$	$\mathcal{R}_1$

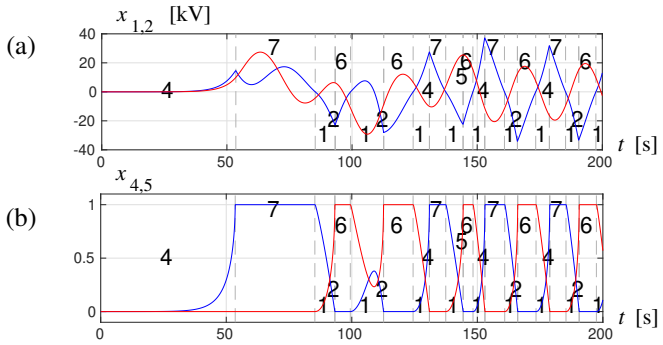

Fig. 6. Solutions for the memristor-based oscillator found using the Poincaré map approach with the maximum time step $\tau = 0.1$; (a) voltages $x_1(t)$, $x_2(t)$, (b) internal states $x_4(t)$ and $x_5(t)$ of memristors m_1 and m_2.

The comparison of the results obtained using the two methods considered is presented in Table IV. For different values of the maximum time step τ, we report the integration error ε and the computation time t_c. The integration error ε is defined as the relative error at the integration time $T = 200$

$$\varepsilon = \frac{\|x(T) - x_{\mathrm{ref}}(T)\|}{\|x_{\mathrm{ref}}(T)\|}, \tag{18}$$

TABLE IV
THE RELATIVE INTEGRATION ERROR ε AND THE COMPUTATION TIME t_c FOR THE STANDARD INTEGRATION METHOD AND THE POINCARÉ MAP APPROACH VERSUS THE MAXIMUM TIME STEP.

τ	standard method		Poincaré map approach	
	ε	t_c[s]	ε	t_c[s]
1.000000	1.13361	0.08	0.10857	0.20
0.100000	1.79309	0.17	0.00033	0.33
0.010000	2.22435	0.88	0.00011	1.39
0.001000	2.06627	7.83	—	12.00
0.000100	0.59704	78.52		
0.000050	0.28111	155.34		
0.000020	0.11115	384.16		
0.000010	0.03811	771.64		
0.000005	0.00170	1538.64		

where $x(T)$ is the result of integration obtained using the method considered at the time T and $x_{\rm ref}(T)$ is the result of integration obtained using the reference method (the Poincaré map approach with $\tau = 0.001$).

For both methods decreasing τ leads to better accuracy (ε decreases) at the expense of the computation time ($t_{\rm c}$ grows). For given τ the Poincaré map approach is approximately two times slower than the standard approach. This is related to the necessity of verifying exit conditions in each integration step. When an exit condition is detected additional computations are needed to find the return time and the exit point. Let us note that the Poincaré map approach is much more accurate. For $\tau = 1$ the relative error is $\varepsilon = 0.10857$ and the computation time is $t_{\rm c} = 0.20$ s. With the standard method, such a value of the relative integration error is achieved for $\tau = 0.00002$ with the computation time $t_{\rm c} = 384.16$ s. The Poincaré map approach with $\tau = 0.1$ produces the relative error $\varepsilon = 0.00033$. In this case the computation time is $t_{\rm c} = 0.33$ s. The accuracy obtained is better than the accuracy obtained using the standard approach for all values of τ considered with the computation time exceeding 1500 s. These results show that the Poincaré map approach permits using much larger time steps than the standard numerical integration procedure considered. In this way the computation time may be significantly reduced. Using standard numerical integration methods allows us to obtain accurate results at the expense of very long computation times.

Let us explain in more detail the reasons for inaccuracies observed when using the standard numerical integration procedure and larger time steps. Clearly, the `ode45` procedure is not responsible for these errors. Otherwise, similar errors would be observed when using the Poincaré map approach. Inaccuracies are caused by errors introduced when crossing smooth regions. When leaving a given smooth region the next point of the trajectory is found using the right hand side which becomes invalid after leaving this region. As a result overshooting happens and the internal memristor's state is assigned a value outside the allowed bounds. This value remains constant until the right hand side of the equation defining the derivative of the internal variable changes its sign. Until this moment the memristor's resistance has an erroneous value, which introduces errors in other ODE equations. These errors accumulate and may result in the totally wrong final results. The overshooting error depends on the time step used when crossing the smooth region boundary. Decreasing the maximum time step reduces the overshooting and the related errors. However, the computation time required to obtain a required accuracy may be very large. The Poincaré map approach is a preferred method to reduce these effects.

V. Conclusions

The Poincaré map approach for simulations of circuits containing memristors has been proposed. Its usefulness has been shown using a sinusoidally driven memristor described by the VTEAM model and a memristor based oscillator. It has been shown that the proposed approach is a preferred method for simulations of circuits with memristors having bounded internal states. The proposed approach outperforms standard numerical integration methods in terms of accuracy and the computation time.

Appendix

Series connection of a linear resistor and a VTEAM memristor

The function defining the RHS of (15) for the standard approach (we assume that $w_{\rm on} = 0$):

```
function rhs=vtRHS(t,w,p)
Rm=(p.Roff*w+p.Ron*(p.woff-w))/p.woff;
v=p.u(t)*Rm/(Rm+p.R); % memristor voltage
rhs=0;
if v>p.voff;
  rhs=p.koff*(v/p.voff-1)^p.aoff;
elseif v<p.von;
  rhs=p.kon*(v/p.von-1)^p.aon;
end
if w>=p.woff || w<=0; rhs=0; end
```

In the procedure `vtRHS` the data structure defining parameters of the system is called `p`.

The function defining the RHS of (15) for the Poincaré map approach:

```
function rhs=vtRHSR(t,w,p)
if p.Reg==1 || p.Reg==2
  Rm=(p.Roff*w+p.Ron*(p.woff-w))/p.woff;
  v=p.u(t)*Rm/(Rm+p.R);
  if p.Reg==1;
    rhs=p.koff*(v/p.voff-1)^p.aoff;
  else rhs=p.kon*(v/p.von-1)^p.aon; end
else rhs=0; end
```

The function defining exit conditions for (15) for the Poincaré map approach:

```
function [val,isterm,dir]=vtEvents(t,w,p)
Rm=(p.Roff*w+p.Ron*(p.woff-w))/p.woff;
v=p.u(t)*Rm/(Rm+p.R);
if p.Reg==1                % events for Region 1
  val=[v-p.voff;w-p.woff]; isterm=[1;1]; dir=[-1;1];
elseif p.Reg==2            % events for Region 2
  val=[v-vonp.;w-p.won]; isterm=[1;1]; dir=[1;-1];
elseif p.Reg==3            % events for Region 3
  val=[v-p.voff;v-p.von]; isterm=[1;1]; dir=[1;-1];
elseif p.Reg==4            % events for Region 4
  val=[v-p.von]; isterm=[1]; dir=[-1];
elseif p.Reg==5            % events for Region 5
  val=[v-p.voff]; isterm=[1]; dir=[1];
end
```

In the procedure `vtEvents` the vector of variables defining events is called `val` (an exit event happens when a variable crosses zero). Directions of crossing are defined by the `dir` vector (the value $-1/+1$ selects crossings when the variable defining the event is decreasing/increasing). All events are terminal, i.e., after detection of an event the integration procedure is stopped. This is defined by setting all elements of the `isterm` vectors equal to 1.

The procedure for the integration of (15) using the Poincaré map approach.

```
function vtIntegrate
p.voff=0.15; p.von=-0.20;     % set parameters
p.Ron=100; p.Roff=1000; p.R=200; p.woff=10e-9; p.won=0;
p.aoff=3; p.aon=3; p.kon=-8e-6;  p.koff=4e-6;
p.u=@(t)(0.35*sin(2*pi*t/0.01));
ws=6e-9; p.Reg=3;             % initial point and region
ts=0; te=0.03;                % integration interval
tall=[]; wall=[];             % variables to store results
op=odeset('Events',@vtEvents);
while 1
  [t,w,te,ye,ie]=ode45(@vtRHSR,[ts,te],ws,op,p);
  tall=[tall; t]; wall=[wall; w];
  if (isempty(ie)); break; end % computations finished
```

```
  ts=t(end);                % update initial time
  ws=w(end);                % update initial condition
  if p.Reg==1 && ie==1; tmp=3; end
  if p.Reg==1 && ie==2; tmp=4; end
  if p.Reg==2 && ie==1; tmp=3; end
  if p.Reg==2 && ie==2; tmp=5; end
  if p.Reg==3 && ie==1; tmp=1; end
  if p.Reg==3 && ie==2; tmp=2; end
  if p.Reg==4; tmp=2; end
  if p.Reg==5; tmp=1; end
  p.Reg=tmp;                % update region
end
```

The `ie` variable returned by the `ode45` procedure is the index of an event detected. If `ie` is empty then the computations are stopped due to reaching the integration time `te` and the computations are finished.

Memristor-based oscillator

The function defining the RHS of (16) with the BC memristors' model for the standard approach:

```
function dxdt=moRHS(t,x,p)
Wx4=p.Gon*p.Goff/(p.Gon-(p.Gon-p.Goff)*x(4));
Wx5=p.Gon*p.Goff/(p.Gon-(p.Gon-p.Goff)*x(5));
dxdt1=-p.alpha*((Wx4+Wx5+p.GN2)*x(1)/p.G+x(3));
dxdt2=p.gamma*x(2)+x(3);
dxdt3=p.beta*(x(1)-x(2)-x(3));
dxdt4=(1/p.i0)*Wx4*x(1);
dxdt5=(-1/p.i0)*Wx5*x(1);
if x(4)>1 && dxdt4>0; dxdt4=0; end
if x(4)<0 && dxdt4<0; dxdt4=0; end
if x(5)>1 && dxdt5>0; dxdt5=0; end
if x(5)<0 && dxdt5<0; dxdt5=0; end
dxdt=[dxdt1 dxdt2 dxdt3 dxdt4 dxdt5]';
```

The function defining the RHS of (16) for the Poincaré map approach:

```
function dxdt=moRHSR(t,x,p)
Wx4=p.Gon*p.Goff/(p.Gon-(p.Gon-p.Goff)*x(4));
Wx5=p.Gon*p.Goff/(p.Gon-(p.Gon-p.Goff)*x(5));
dxdt1=-p.alpha*((Wx4+Wx5+p.GN2)*x(1)/p.G+x(3));
dxdt2=p.gamma*x(2)+x(3);
dxdt3=p.beta*(x(1)-x(2)-x(3));
dxdt4=0;
if p.Region==1 || p.Region==4 || p.Region==5
  dxdt4=(1/p.i0)*Wx4*x(1);
end
dxdt5=0;
if p.Region==1 || p.Region==2 || p.Region==3
  dxdt5=(-1/p.i0)*Wx5*x(1);
end
dxdt=[dxdt1 dxdt2 dxdt3 dxdt4 dxdt5]';
```

The function defining exit conditions for the system (16) for the Poincaré map approach:

```
function [val,isterm,dir]=moEvents(t,x,p)
if p.Reg==1               % events for Region 1
  val=[x(4);x(4)-1;x(5);x(5)-1];
  isterm=[1;1;1;1]; dir=[-1;1;-1;1];
elseif p.Reg==2           % events for Region 2
  val=[x(1);x(5)-1]; isterm=[1;1]; dir=[1;1];
elseif p.Reg==3           % events for Region 3
  val=[x(1);x(5)]; isterm=[1;1]; dir=[-1;-1];
elseif p.Reg==4           % events for Region 4
  val=[x(1);x(4)-1]; isterm=[1;1]; dir=[-1;1];
elseif p.Reg==5           % events for Region 5
  val=[x(1);x(4)]; isterm=[1;1]; dir=[1;-1];
elseif p.Reg==6           % events for Region 6
  val=[x(1)]; isterm=[1]; dir=[1];
elseif p.Reg==7           % events for Region 7
  val=[x(1)]; isterm=[1]; dir=[-1];
end
```

The procedure for the integration of (16) using the Poincaré map approach:

```
function moIntegrate
p.Goff=0.06e-3; p.Gon=1.9e-3; % set parameters
p.G=3.3e-3; p.GN1=-0.4e-3; p.GN2=-1.2e-3;
p.alpha=0.74; p.beta=0.0333; p.gamma=0.12; p.i0=8.9189;
ts=0; te=200;             % integration interval
xs=[0.006,0.02,-0.3,0,0]'; % initial point
p.Reg=4;                  % initial region
op=odeset('MaxStep',1e-3,'Events',@moEvents)
tall=[]; xall=[];         % variables to store results
while 1
  [t,x,te,ye,ie]=ode45(@moRHSR,[ts,te],xs,op,p);
  tall=[tall; t]; xall=[xall' x']';
  ts=t(end);              % update initial time
  xs=x(end,:);            % update initial condition
  if (isempty(ie)); break; end % computations finished
  if p.Reg==1 && ie==1; tmp=2; end
  if p.Reg==1 && ie==2; tmp=3; end
  if p.Reg==1 && ie==3; tmp=4; end
  if p.Reg==1 && ie==4; tmp=5; end
  if p.Reg==2 && ie==1; tmp=1; end
  if p.Reg==2 && ie==2; tmp=6; end
  if p.Reg==3 && ie==1; tmp=1; end
  if p.Reg==3 && ie==2; tmp=7; end
  if p.Reg==4 && ie==1; tmp=1; end
  if p.Reg==4 && ie==2; tmp=7; end
  if p.Reg==5 && ie==1; tmp=1; end
  if p.Reg==5 && ie==2; tmp=6; end
  if p.Reg==6 || p.Reg==7; tmp=1; end
  p.Reg=tmp;              % update region
end
```

REFERENCES

[1] D. B. Strukov, G. S. Snider, D. R. Stewart, and R. S. Williams, "The missing memristor found," *Nature*, vol. 453, pp. 80–83, May 2008.

[2] Z. Biolek, D. Biolek, and B. Biolkova, "Spice model of memristor with nonlinear dopant drift," *Radio Eng.*, vol. 18, no. 2, pp. 210–214, June 2009.

[3] Y. N. Joglekar and S. T. Wolf, "The elusive memristor: Properties of basic electrical circuits," *Eur. J. Phys.*, vol. 30, no. 4, pp. 661–675, May 2009.

[4] F. Corinto and A. Ascoli, "A boundary condition-based approach to the modeling of memristor nanostructures," *IEEE Trans. Circuits Syst. I*, vol. 59, no. 11, pp. 2713–2726, Nov. 2012.

[5] T. Prodromakis, B. P. Peh, C. Papavassiliou, and C. Toumazou, "A versatile memristor model with non-linear dopant kinetics," *IEEE Trans. Electron Devices*, vol. 58, no. 9, pp. 3099–3105, Sept. 2011.

[6] A. F. Filippov, *Differential Equations with Discontinuous Righthand Sides: Control Systems*. Norwell, MA, USA: Kluwier, 1988.

[7] Z. Galias, "Return map approach for simulations of electronic circuits with memristors," in *Proc. Int. Conf. Signals Electronic Syst. (ICSES)*, Kraków, Poland, Sept. 2018, pp. 147–150.

[8] A. Ascoli, R. Tetzlaff, Z. Biolek, Z. Kolka, V. Biolková, and D. Biolek, "The art of finding accurate memristor model solutions," *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 5, no. 2, pp. 133–142, June 2015.

[9] F. Corinto and M. Forti, "Memristor circuits: Flux—Charge analysis method," *IEEE Trans. Circ. Syst. I, Reg. Papers*, vol. 63, no. 11, pp. 1997–2009, Nov. 2016.

[10] ——, "Complex dynamics in arrays of memristor oscillators via the flux-charge method," *IEEE Trans. Circ. Syst. I, Reg. Papers*, vol. 65, no. 3, pp. 1040–1050, Mar. 2018.

[11] L. O. Chua, "Memristor—The missing circuit element," *IEEE Trans. Circ. Theory*, vol. 18, no. 5, pp. 507–519, Sept. 1971.

[12] L. O. Chua and S. M. Kang, "Memristive devices and systems," *Proc. IEEE*, vol. 64, no. 2, pp. 209–223, Feb. 1976.

[13] L. O. Chua, "Everything you wish to know about memristors but are afraid to ask," *Radioengineering*, vol. 24, no. 2, pp. 319–368, June 2015.

[14] S. Kvatinsky, M. Ramadan, E. G. Friedman, and A. Kolodny, "VTEAM: A general model for voltage-controlled memristors," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 62, no. 8, pp. 786–790, Aug. 2015.

[15] M. Hénon, "On the numerical computation of Poincaré maps," *Physica D*, vol. 5, no. 2–3, pp. 412–414, 1982.

[16] W. Govaerts, Y. A. Kuznetsov, and A. Dhooge, "Numerical continuation of bifurcations of limit cycles in MATLAB," *SIAM J. Sci. Comput.*, vol. 27, no. 1, pp. 231–252, 2005.

[17] L. F. Shampine and S. Thompson, "Event location for ordinary differential equations," *Computers and Mathematics with Applications*, vol. 39, no. 5–6, pp. 43–54, 2000.

[18] CAPD library. [Online]. Available: http://capd.ii.uj.edu.pl/.

Zbigniew Galias received the M.Sc. degree in electronics from the AGH University of Science and Technology, Kraków, Poland in 1990, the M.Sc. degree in mathematics from Jagiellonian University, Kraków in 1992, and the Ph.D. and Senior Doctorate degrees in electrical engineering from the AGH University of Science and Technology, Kraków, in 1996 and 2004, respectively.

In 2015, he was conferred the title of Professor by the President of Poland. Since 2005, he has been a Professor of electrical engineering at the AGH University of Science and Technology. He has held visiting research positions at the Technical University Munich, the University of California at Berkeley, the University of California at San Diego, and RMIT University, Melbourne. He has published more than 100 research articles. His research interests include simulation, analysis, and design of nonlinear systems, chaos, interval arithmetic, computer assisted proofs and sliding mode control.

Dr. Galias has served/is serving as an Associate Editor for the *IEEE Transactions on Circuits and Systems—I* and *II*, the *IEEE Circuits and Systems Magazine*, and the *International Journal of Bifurcation and Chaos*. He was awarded the Polish Research Foundation scholarship in 1994, the Polish Prime Minister Award for the Ph.D. thesis in 1997, and the Fulbright Fellowship in 1999.