CHALMERS
UNIVERSITY OF TECHNOLOGY

On using reservoir computing for sensing applications: exploring environment-sensitive memristor networks

The International Journal of Parallel, Emergent and Distributed Systems
Vol. 33, No. 4, February 2017, 1–27

GUIDE

On using reservoir computing for sensing applications: exploring environment-sensitive memristor networks

Vasileios Athanasiou[a] and Zoran Konkoli[a*]

[a]*Department of Microtechnology and Nanoscience, Chalmers University of Technology, Gothenburg, Sweden*

(*v3.6 released March 2011*)

Recently, the SWEET sensing setup has been proposed as a way of exploiting reservoir computing for sensing. The setup features three components: an input signal (the drive), the environment and a reservoir, where the reservoir and the environment are treated as one dynamical system, a super-reservoir. Due to the reservoir-environment interaction, the information about the environment is encoded in the state of the reservoir. This information can be inferred (decoded) by analyzing the reservoir state. The decoding is done by using an external drive signal. This signal is optimized to achieve a separation in the space of the reservoir states: Under different environmental conditions, the reservoir should visit distinct regions of the configuration space. We examined this approach theoretically by using an environment-sensitive memristor as a reservoir, where the memristance is the state variable. The goal has been to identify a suitable drive that can achieve the phase space separation, which was formulated as an optimization problem, and solved by a genetic optimization algorithm developed in this study. For simplicity reasons, only two environmental conditions were considered (describing a static and a varying environment). A suitable drive signal has been identified based an intuitive analysis of the memristor dynamics, and by solving the optimization problem. Under both drives the memristance is driven to two different regions of the one-dimensional state space under the influence of the two environmental conditions, which can be used to infer about the environment. The separation occurs if there is a synchronization between the drive and the environmental signals. To quantify the magnitude of the separation, we introduced a quality of sensing index: The ability to sense depends critically on the synchronization between the drive and environmental conditions. If this synchronization is not maintained the quality of sensing deteriorates.

Keywords: reservoir computing; sensing; memristor networks; environment-sensitive memristor

1. Introduction

According to Moore's law the number of microprocessors at the chip doubles roughly every second year. [1] It has been predicted that this trend will slow down for various reasons, either due to practical engineering limitations (e.g. the wiring problem) or due the presence of effects that are specific to small scales (e.g. relativistic or quantum effects). The field of unconventional computation emerged as a response to this challenge. The goal is to develop alternative information processing devices by using non-CMOS technologies. There are many examples of such computing frameworks, such as neuromorphic computing, molecular computing, or reaction-diffusion computing, to name a few. [2–7] In particular, reservoir computing gained a considerable interest in the recent decade.

*Corresponding author. Email: zorank@chalmers.se

The field of reservoir computing started as an insight about behaviour of synaptic weights during the neural network training process. [8–11] The weights need to be adjusted to achieve a desired computation. It has been realized that only a limited set of weights is adjusted in the network training process, normally referred to as an "outer layer". This led to the further insight that instead of neural network one could use an arbitrary dynamical system for computation, and augment it with the outer layer. The dynamical system used this way is referred to as a reservoir. The modern understanding of reservoir computing emphasizes the fact that a Turing universal expressive power can be achieved if the reservoir exhibits separation property on the state of inputs. The separation property is often realized by complex systems at the edge of chaos. [12]

Recently, a novel approach for using reservoir computing for sensing has been suggested, [13, 14] to be referred to as the SWEET sensing setup. The framework features three components: An environment one wishes to sense (a sensing target), an environment sensitive element, and a user defined signal (the drive). A tight link between the sensing element and the environment is assumed: both are treated conceptually as one dynamical system, a super-reservoir. The sensing is done by using an indirect sensing approach: A behavior of the sensing unit is studied under different environmental conditions q and different drives u, and if different environmental conditions lead to radically different responses, this can be used to infer the state of the environment.

In brief, the goal is to find a single drive u that will cause the reservoir to occupy vastly different regions of the configuration space, $\Omega_1, \Omega_2, \cdots, \Omega_E$, under distinct environmental conditions, $q_1, q_2, \cdots, q_E$. To achieve unambiguous sensing, it is necessary to find a drive such that the distance between the regions $\Omega_1, \Omega_2, \cdots, \Omega_E$ is maximized (phase space separation).

Such sensors could be used in situations where the use of the standard CMOS-based technology is not possible or is limited in some way, e.g. due to the lack of biocompatibility, if there is a need for low-power sensing, or simply where the embedded sensing application needs to be realized. For example, a series of medical challenges require advanced sensing techniques, and many novel biotechnological sensing platforms have been developed aiming at fast response (on-line) medical care for minimally invasive health monitoring. These developments often involve interdisciplinary work at the interface between physics, chemistry, biology, and computer science.

We investigate the possibility of using memristor networks to realize the SWEET sensing setup. There are two reasons for choosing the memristor to demonstrate the sensing principle. First, if one can assume that the memristance depends on the environmental condition, the memristor is one of the simplest electronic components that can function as the state weaver, and is extremely suitable for testing the SWEET algorithm both in theoretical and practical terms. Second, we wish to examine the principal question whether an environment sensitive memristor can function as the SWEET sensing element. Should this be possible, the number of memristor applications can be increased, realizing novel functionalities beyond the intended scope that the technology has been developed for (e.g. memory or generic analog computation).

The work is organized as follows. In section 2, the problem of identifying the drive is formulated as an optimization problem. In section 3, an instance of the optimization problem has been solved for two environmental conditions. A quality of sensing index is introduced and evaluated under various conditions. Section 4 contains the summary of the main findings.

2. Methods

Memristor model: A memristor is a two port element with time-dependent resistance and memory. The resistance depends on the amount of electrical charge $Q(t)$ that has passed through the element. Thus the resistance R changes depending on the current I that passed through the memristor. The current is controlled by the applied voltage ΔV that the memristor has been exposed to. This behavior can be described as

$$\dot{R}(t) = f_\xi[\Delta V(t)]\ \theta[R(t) - R_{min}]\ \theta[R_{max} - R(t)] \tag{1}$$

where t denotes time, $\theta(x)$ is the step function $[\theta(x > 0) = 1,\ \theta(x \leq 0) = 0]$ and ξ is a list of parameters that describe the memristor, e.g. the resistance bounds, the threshold voltages, etc. When convenient the index ξ will be omitted, and f will be used instead of f_ξ.

In this work we use a simple theoretical model of the memristor suggested in [15]. The model describes most of the experimental findings:

$$f[\Delta V(t)] = \beta\ \Delta V(t) + \frac{1}{2}(\alpha - \beta)\ [|\Delta V(t) + V_{\rm thr}| - |\Delta V(t) - V_{\rm thr}|] \tag{2}$$

The parameter α describes how fast the resistance changes when the applied voltage $|\Delta V|$ is smaller than the threshold $V_{\rm thr}$. For larger voltages the resistance changes more rapidly described by the parameter β. Equation (2) should only be used when $R_{\rm min} \leq R \leq R_{\rm max}$ since the resistance is always bounded. Therefore, outside the range $R \in [R_{\rm min}, R_{\rm max}]$, $\dot{R} = 0$. Thus for this particular model the list of parameters is given by $\xi = \{\alpha, \beta, V_{thr}, R_{\rm min}, R_{\rm max}\}$ which describe the properties of the material that the memristor is made of. In experiments usually $\beta \gg \alpha$.

It is assumed that the influence of the environment can be described by a suitable modification of Eq. (1),

$$\dot{R}(t) = f_\xi[\Delta V(t), q(t)]\ \theta[R(t) - R_{min}]\ \theta[R_{max} - R(t)] \tag{3}$$

where $q(t)$ is a variable (observable) that describes the environment. The key challenge is to construct the function $f_\xi[\Delta V, q]$. For practical reasons, we model the influence of the environment by assuming the original form of f but with the parameters that are affected by the environment. In rigorous mathematical terms, this can be formulated as

$$f_\xi[\Delta V, q] = f_{\xi(q)}[\Delta V] \tag{4}$$

The question is which of the parameters in $\xi(q)$ are environment-sensitive.

The general consensus regarding the principles of the memristor operation, e.g. as discussed in references [16, 17], is that the common mechanism behind the resistance change is the alternation in the microscopic structure of the material caused by the transport of chargers and atoms. This transport is controlled by the applied voltage, which affects the injection and the removal of charges at the interfaces, and the transport through the memristive material. Our hypothesis is that the presence of ions in the solution changes the value of the electrical potential in the memristive material. While not directly affecting the transport within the material, this could affect the charge injection and removal processes at the interfaces. Since the memristor is behaving as a static material in the regime when

the applied voltage is low, this should not affect parameter α, however, parameters β and V_{thr} might be affected. The minimal and maximal resistance values are controlled by the manufacturing processes (e.g. the width of the titanium dioxide layer) and they should not depend strongly on the environment. For simplicity reasons we only investigate the model where parameter β is environment sensitive: $\xi(q) = \{\alpha, \beta(q), V_{thr}, R_{\min}, R_{\max}\}$.

To simplify the analysis, we used the simplest possible form of $\beta(q)$, the linear relationship

$$\beta(t) = c_1 \; q(t) + c_0 \tag{5}$$

where c_0 and c_1 are arbitrary constant which for simplicity reasons are taken to be $c_0 = 0$ and $c_1 = 1$, which trivially implies $\beta = q$. This linear approximation should be valid in the limit when the concentration of ions is small. Accordingly, the function f is given by

$$f[\Delta V(t), q(t)] = q(t) \; \Delta V(t) + \frac{1}{2} \; [\alpha - q(t)] \; [|\Delta V(t) + V_{\rm thr}| - |\Delta V(t) - V_{\rm thr}|] \tag{6}$$

Figure 1 shows how f changes depending on variations in the environmental conditions.

Sensing procedure: The state of the reservoir at every time instance t is defined by the memristance values of the network: $R_i(t)$ with $i = 1, 2, \cdots, N_R$. The sensing is implemented as a computation. The input to the computation consists of the drive signal $u(t)$, and the external field variable $q(t)$. The output of the computation is constructed using the standard reservoir computing technique. A simple linear readout mechanism is used to assess the state of the system, the resistances $R_{\rm i}[q, u](t), i = 1, 2, \cdots, N_R$, and calculate the output as

$$y_w[q, u](t) = w_0 + \sum_{i=1}^{N_{\rm R}} w_{\rm i} \; R_{\rm i}[q, u](t) \tag{7}$$

where $w = \{w_0, w_1, ..., w_{N_R}\}$ are adjustable parameters and $N_{\rm R}$ is the number of memristors. The notation $R_{\rm i}[q, u](t)$ is used to emphasize the fact that the memristance $R_{\rm i}$ at a time point t depends on earlier values of q and u. In rigorous mathematical terms the memristance is a functional of these two variables. The memristance depends on the history of how these two variables change in time, thus on an infinite (uncountable) sequence of values. Figure 2 illustrates how the output is constructed.

The goal is to find a drive signal $u(t)$ such that $y_w[q, u](t)$ can be used to characterize the state of environment $q(t)$ [13, 14]. Mathematically the characterization procedure can be described as the process of computing

$$\phi[q](t) \tag{8}$$

at every time instance t where ϕ is a known mapping defined by the user. It specifies which feature of the environment one wishes to characterize. For example, if the goal were to distinguish between two environmental conditions q_0 and q_1 one could use the mapping ϕ with the following properties: $\phi[q] \approx 1$ for $q \approx q_1$ and $\phi[q] \approx 0$

for $q \approx q_0$. The goal is to find $u(t)$ such that

$$y_w[q,u](t) \approx \phi[q](t) \tag{9}$$

holds for every time instance t and every state of the environment q.

The procedure of finding the most optimal drive is equivalent to solving the following optimization problem. The goal is to minimize the difference between the desired output $\phi[q](t)$ and the simulated output $y_w[q,u](t)$ for all environmental conditions of interest. Therefore, the problem is to identify a drive signal $u(t)$ and the coefficients w such that

$$\delta_w^2[u](t) = \sum_{\gamma=1}^{E} |y_w[q_\gamma,u](t) - \phi[q_\gamma](t)|^2 \tag{10}$$

is as small as possible for every time instance and environmental condition. This is achieved by minimizing

$$\delta_w^2[u] = \lim_{T\to\infty} \frac{1}{T} \int_0^T \delta_w[u](t)\, dt \tag{11}$$

over both w and u. By assumption the system is observed starting from $t = 0$. Note that the equations above should be augmented with the information about the initial conditions (e.g. the state of the memristors, the state of the environment, etc.). However, since the memristor model assumed in here exhibits the fading memory property [5], the influence of the initial conditions "fades" in time if the system is driven for a long time. Thus if T is large the effect of the initial condition can be neglected, which will indeed be assumed for simplicity reasons. The following expression is a compact representation of the optimization problem:

$$\delta_* = \min_{w,u} \delta_w[u] \tag{12}$$

Genetic algorithm optimization: A genetic algorithm package was developed to solve the above optimization problem using the Mathematica programming language. The algorithm is based on the concepts of mutation and crossover, and it works as follows.

Since $q(t)$ is assumed to be periodic it is reasonable to search for periodic drive signals too. The drive signal $u(t)$ is defined in terms of its Fourier coefficients and its frequency:

$$u(t) = a_0 + \sum_{i=1}^{N_c} a_i \sin(i\omega t) + \sum_{i=1}^{N_c} b_i \cos(i\omega t) \tag{13}$$

In this way the signal u is encoded as a set of parameters

$$P_u = \{a_0, a_1, ..., a_{N_c}, b_1, ..., b_{N_c}, \omega\} \tag{14}$$

Therefore, the optimization problem given in Eq. (12) can be rewritten as follows:

$$\delta[P_u, w](t) = \sum_{\gamma=1}^{E} |y_w[q_\gamma,u](t) - \phi[q_\gamma](t)|^2 \tag{15}$$

For a given u, the weights w are identified by linear regression,

$$\delta[P_u] = \min_w \lim_{T\to\infty} \frac{1}{T}\int_0^T \delta[P_u, w](t)\, dt \tag{16}$$

where several environmental conditions are considered simultaneously. This is important because one drive u has to work for all environmental conditions. In the final step one needs to find the optimal set of parameters P_u by minimizing

$$\delta_* = \min_{P_u} \delta[P_u] \tag{17}$$

which will be referred to as the cost value.

The most optimal set of parameters $P_{\rm u}$ is found as follows. The algorithm is used to identify a globally optimal solution within the given bounds for P_u. Each parameter in P_u is limited to a user-defined interval. Once chosen these intervals are kept fixed. The algorithm includes the following steps:

(1) Initialize a pool of random initial candidate solutions. Their number is defined by the user and is denoted by M. Each candidate solution is constructed by choosing the parameters in $P_{\rm u}$ randomly from their respective intervals.
(2) Initialize variables $P_{u1}^{OLD}, P_{u2}^{OLD}, \cdots P_{uM}^{OLD}$ by using the candidate solutions found in step 1.
(3) Evaluate the fitness of $P_{u1}^{OLD}, P_{u2}^{OLD}, \cdots P_{uM}^{OLD}$ using Eq. (17). This is computationally the most demanding step of the algorithm.

FOR LOOP 1: For every P_{ui}^{OLD} construct the drive $u(t)$ by using Eq. (13).

FOR LOOP 2: For every environmental condition $q_1, q_2, \cdots, q_E$, simulate the memristor network in the interval $[0, T]$[1] and store the values of the resistances $R_i[q_\gamma, u](t)$ for all memristors $i = 1, 2, \cdots, N_R$, environmental conditions $\gamma = 1, 2, \cdots, E$, and with the pre-defined set of time instances that are used for sampling, t_k with $k = 1, 2, \cdots, N_T$. The separation between two adjacent time instances is $t_{k+1} - t_k = \Delta t$.

End LOOP 2

Calculate the w and the cost $\delta[{P_u}^{OLD}{}_i]$ by using

$$\delta[P_u] = \min_w \left(\frac{1}{N_T}\sum_{k=1}^{N_T} \delta[P_u, w](t_k) \right) \tag{18}$$

which is the discretized version of Eq. (16). Similarly, the discrete version of Eq. (11) is given as:

$$\delta_w^2[u] = \frac{1}{N_T}\sum_{k=1}^{N_T} \delta_w[u](t_k) \tag{19}$$

[1]in this work the simulator was developed in [18]

End LOOP 1

(4) Let $P_{u1}^{NEW}, P_{u2}^{NEW}, \cdots P_{uN_{new}}^{NEW}$ be the solutions which are created by either mutation or crossover between two or one of the solutions $P_{u1}^{OLD}, P_{u2}^{OLD}, \cdots P_{uM}^{OLD}$. Form new candidate solutions by randomly choosing between either crossover or mutation. If a crossover is chosen, perform N_{new} crossovers and create N_{new} new solutions. Every crossover is used to form one new candidate solution between two randomly chosen solutions of the solutions $P_{u1}^{OLD}, P_{u2}^{OLD}, \cdots P_{uM}^{OLD}$. If a mutation is flagged, perform N_{new} mutations and create N_{new} solutions. Every mutation is used to form one new candidate solution by mutating one of $P_{u1}^{OLD}, P_{u2}^{OLD}, \cdots P_{uM}^{OLD}$ chosen at random.
(5) Repeat step (3) with $P_{ui}^{OLD} \leftarrow P_{ui}^{NEW}$.
(6) Sort the new and old solutions based on their costs. Keep the M solutions with the smallest costs δ_* (the remaining ones are ignored), which will become $P_{u1}^{OLD}, P_{u2}^{OLD}, \cdots P_{uM}^{OLD}$ in the next pass through step 4.
(7) Terminate if the smallest cost is below the target threshold or if the maximum number of iterations has been reached. Otherwise, continue with step (4). If terminated, the solution with the smallest cost is an approximation to the optimization problem.

The maximum frequency $\omega_{\max} \equiv N_c\, \omega$ used to represent the drive $u(t)$ should be carefully chosen depending on the value of Δt so that roughly $\omega_{\max} \lesssim 2\pi/\Delta t$. The highest frequency of a Fourier component was chosen ten times smaller than $1/\Delta t$ so as to assure that there is enough resolution to represent the highest frequency components. The increment Δt should be small enough to ensure there is no aliasing of the high frequency components of $u(t)$.

3. Results

For simplicity reasons and to illustrate the key ideas, we have investigated in detail the simplest form of a memristor network, the one-memristor network as illustrated in Fig. 3. Only two environmental conditions are assumed, $E = 2$, that are denoted by $q_1(t)$ and $q_2(t)$. The function $q_1(t)$ is constant in time, and $q_2(t)$ has a periodic square-wave form; q_1 mimics an inert environment and q_2 is a simple example of a varying environment.

To obtain the numerical results all variables have been re-scaled to obtain dimensionless quantities. The mathematical details are shown in the appendix. In brief, every variable $z \in \{t, f, R, V, \alpha, \beta, q\}$ is expressed as $z = \tilde{z} z_0$ where z_0 carries the dimension. Then, all equations are expressed in the variables $\tilde{z} \in \{\tilde{t}, \tilde{f}, \tilde{R}, \tilde{V}, \tilde{\alpha}, \tilde{\beta}, \tilde{q}\}$. We use: $t_0 = 1s$, $R_0 = 1\Omega$, $V_0 = 1V$, and $q_0 = \alpha_0 = \beta_0$ is defined in the appendix. In the following, the tilde symbol will be omitted from $\tilde{t}$, $\tilde{f}$, $\tilde{R}$, $\tilde{V}$, $\tilde{\alpha}$, $\tilde{\beta}$, and $\tilde{q}$.

The functions $q_1(t)$ and $q_2(t)$ are defined as

$$W_{f_c,t_d}(t) = W[f_c \cdot (t - t_d)] \tag{20}$$

$$q(t) = q_0 + A_0 \cdot W_{f_c,t_d}(t) \tag{21}$$

where function W describes a square-wave form with the amplitude and the period equal to 1 (the nodes are at $n/2$ with n an integer). Function W_{f_c,t_d} is obtained after shifting W in time by t_d and re-scaling the frequency by factor f_c. The time dependence of W_{f_c,t_d} has also the form of a square wave, with frequency f_c. Function $q_1(t)$ is defined by $A_0 = 0$ and $q_0 = 3$. Function $q_2(t)$ is given by $q_0 = 3$, $A_0 = 1$, $f_c = 2$ and $t_d = 0$. The graphs of q_1 and q_2 are shown in Fig. 4.

To realize the SWEET sensing setup one has to find a drive $u(t)$ that can distinguish between these two environments, i.e. the network should output 0 when exposed to the environment q_1 and 1 when exposed to the environment q_2. Mathematically, the network should mimic ϕ that behaves as $\phi[q_1] = 0$ and $\phi[q_2] = 1$.

3.1 The one-memristor network sensor with an intuitive drive

After an extensive analysis of the numerical results, we identified the mechanism that was responsible for phase space separation. Based on this insight, a representative drive signal was constructed, without using the algorithm, that can be used to illustrate the key idea behind the sensing mechanism. This signal will be referred to as "guessed" or "intuitive" drive and will be denoted by u_1. The results of the computer simulation of the memristor behavior under the influence of the two environments are shown in Figs. 5, 6, 7. The drive u_1 was chosen to achieve the following.

In Fig. 5 the result of simulating the one memristor network is shown for the constant environment. The drive $u_1(t)$ was constructed so that the resistance $R_{11}(t) \equiv R[q_1, u_1](t)$ increases when $u_1(t)$ is positive and it decreases when $u_1(t)$ is negative. Because the environment signal $q_1(t)$ does not change in time, the only parameter which affects the function f (cf. Eq. 2) is the drive signal $u_1(t)$. Since $u_1(t)$ has equal positive and negative amplitudes, the increase and decrease of the $R_{11}(t)$ happens with the same slope (of the opposite sign). As a result of that behavior, there is no overall change in $R_{11}(t)$ over one period.

In Fig. 6 the result of simulating the one memristor network is shown when the environment has a square wave behavior ($q = q_2(t)$). The drive was constructed so that $u_1(t)$ is positive (negative) when $q_2(t)$ is high (low). In this way, $R_{21}(t) \equiv R[q_2, u_1](t)$ has an increase with a high slope and a decrease with a smaller slope. Accordingly, there is an overall increase in $R_{21}(t)$ over one period. Clearly, this tendency gets amplified over subsequent periods and after a finite number of periods, $R_{21}(t)$ reaches the maximum value R_{max}.

Output $y[q,u_1](t)$ for both environmental conditions is shown in Fig. 7. The resistances $R_{11}(t)$ and $R_{21}(t)$ were used to calculate the output $y[q, u_1](t)$ based on Eq. (7).

Key result: The one memristor network with the drive $u_1(t)$ functions as a SWEET sensor that can distinguish between the two environments, the inert one q_1 and the varying one q_2. When exposed to q_1 the sensor outputs a value close to 0, and when exposed to q_2 a value close to 1. However, if there was a slight loss of synchronization between $u_1(t)$ and $q_2(t)$, the sensor would be less efficient.

A difficulty with constructing a drive by hand is that the way of thinking presented in this section cannot be generalized for larger networks. In the next section, an identical analysis will be done but with a drive found by using the automatic procedure detailed earlier (genetic algorithm optimization, section 2).

3.2 Solving the optimization problem of a one-memristor network

The algorithm discussed in the methods section has been used to find the optimal drive for the sensing problem depicted in Fig. 3. The algorithm was run with $N_c = 4$ which resulted in the drive $u = u_2$. The operation of the sensor (driven by u_2) has been simulated under the influence of the two environmental conditions. The results of the simulations are shown in Figs. 8, 9, 10.

Figure 8 shows the result of the simulation of the SWEET sensor while exposed to q_1. One can see that the resistance $R_{12}(t) \equiv R[q_1, u_2](t)$ increases gradually, for the same reasons that were discussed earlier regarding the choice of the intuitive drive $u_1(t)$: When the drive $u_2(t)$ is positive, the $R_{12}(t)$ increases and when $u_2(t)$ is negative the $R_{12}(t)$ decreases. Since q_1 is constant, the overall rate of change over one period of $u_2(t)$ depends on the mean value of $u_2(t)$. The algorithm produced a signal with the mean value that is positive and, accordingly, the resistance $R_{12}(t)$ has a tendency to increase (over one period of u_2).

Figure 9 shows the same as Fig. 8 but for the varying environment q_2. Under this environmental condition the resistance $R_{22}(t) \equiv R[q_2, u_2](t)$ gradually decreases in time. It is not easy to interpret this behavior as for u_1. The graphs show a complex interplay between the timings of q_2 and u_2. Very likely a human could never guess such a signal. The resistances $R_{12}(t)$ and $R_{22}(t)$ are further processed by the readout layer to produce the output which is shown in Fig. 10.

Key result: The drive found by the algorithm leads to the phase separation too. However, the signal found by the algorithm leads to a different separation of the configuration space of the sensor. This is discussed in the next section.

3.3 Comparison between the outputs

Intuitively one would expect that the sensor performs better if operated with the drive found by the algorithm. A quick inspection of the graphs in Figs. 7 and 10 shows that to some extent this might be true. However, the output with u_2 varies more than the one with u_1, and it is rather challenging to claim a clear "winner". In the following several intriguing differences of the sensor behavior when operated with u_1 and u_2 are discussed. These differences are summarized in tab. 1.

Table 1. The overview of the sensor behavior under the two drives with the main conclusions (top row). The second and the third rows provide the supporting information. Variable R_{init} is used to denote the initial condition of the memristance.

	Signal u_2 achieves a better phase space separation	Signal u_1 achieves a better accuracy (smaller variation)	Signal u_2 achieves a faster phase separation
u_1	the key parameter to observe: the separation $R_{max} - R_{init}$ in Figs. 5 and 6	the key quantity to observe: the variation of $y[q, u](t)$ around $\phi[q_1]$ and $\phi[q_2]$ in Fig. 7	phase space separation is achieved after $\Delta t \approx 5$, see Fig. 7
u_2	the key parameter to observe: $R_{max} - R_{min}$ in Figs. 8 and 9	the key quantity to observe: the variation of $y[q, u](t)$ around $\phi[q_1]$ and $\phi[q_2]$ in Fig. 10	phase space separation is achieved after $\Delta t \approx 2$, see Fig. 10

Signal u_2 achieves a better phase separation: For example, on one hand, in the case of the signal found by the algorithm, u_2, the resistance is made to increase

to $R_{\max}$ and decrease to $R_{\min}$ for the distinct environmental conditions. On the other hand, for the sensor operated with u_1 (the guessed drive), only the increase to $R_{\max}$ is observed. This implies that, in principle, the optimized drive is more efficient in achieving the phase space separation.

Signal u_1 achieves a better accuracy: The output with u_1 varies less than the output with u_2 Signal u_2 is an order of magnitude larger than u_1 and, as a result, the resistance $R[q, u_2]$ changes faster than $R[q, u_1]$. Therefore, for a better accuracy, smaller drives might be better, leading to more stable signals.

Signal u_2 achieves a faster phase separation: This behavior is potentially important for on-line sensing applications, when the rapid changes in the state of the environment should be detected. To do this, the sensor needs to be sufficiently responsive. One can see that the phase separation is achieved for u_1 (u_2) within the interval $\Delta t \approx 5$ (2). This happens since $u_2 \gg u_1$.

This indicates that, in principle, larger drives lead to a fast response, but at the cost of decrease in accuracy, and there is a trade-off between these choices. When the drive u_1 was constructed, this trade-off was not a determining factor. On the other hand, the optimization problem weights them both, and the signal u_2 incorporates this trade-off. By design, the algorithm "demands" both a fast response and a high accuracy (non-volatile output). The high accuracy is rewarded by minimizing δ_* (to achieve the clustering of the outputs around the respective sensing goals). The fast response is favoured implicitly. Equation (17) involves all the time-points from the initial state, and for the finite simulation time T the initial output values could dominate. For $T \to \infty$ the accuracy is favoured over the speed of the response.

3.4 Measuring quality of sensing

The question is whether it is possible to compare the performance of different sensors in a simple way, e.g. to study how the quality of sensing depends on changes in different sensing setups. This is important from a practical point of view. The pertinent question is how an applied drive signal affects the performance of the sensor, or one might be interested to understand how a replacement of a sensor component affects the device.

In principle, one could always use δ_* to estimate the quality of sensing, but this quantity depends on how the readout layer has been engineered. It would be better to quantify the performance of the sensor in a more direct way. Since the phase space separation realized by the sensor is determining factor for good sensing performance, we exploit it instead to define a quantitative measure of the quality of sensing.

In the case of the one-memristor sensor it is possible to measure the phase space separation in a very precise way. In rigorous mathematical terms, we introduce a *Quality of Sensing index* ν when $E = 2$,

$$\nu = \frac{|\mu_1 - \mu_2|}{\sigma_1 + \sigma_2} \tag{22}$$

where μ_1, μ_2 and σ_1, σ_2 are the means and the standard deviations of $R[q_1, u](t)$ and $R[q_2, u](t)$ respectively. A relatively high index ν means that a high degree of configuration space separation has been achieved. This in turn implies that the sensing element can easily distinguish between two environments.

For example, by calculating the index ν for the guessed drive and the drive found by the algorithm, we found that the index ν is larger for the drive found by the

algorithm ($\nu = 2.07$) than for the guessed drive ($\nu = 1.92$), but the values are rather close. This would suggest that the drive found by the algorithm works a bit better. This is not an unrealistic prediction, given the discussion in the previous section.

3.5 Robustness of the sensing procedure

The drives u_1 and u_2 have been optimized for the specific sensing setup with two *a priori* known environmental conditions. However, in real applications, there are unpredictable factors which might deteriorate the performance of the sensor. The question is whether the sensor operated with an optimized drive could function under realistic experimental conditions. To analyze this, the robustness of the configuration space separation has been examined in two ways. First, this has been done by analyzing how ν varies when $q_2(t)$ is altered and $q_1(t)$ and $u_2(t)$ are kept unchanged: $\nu(q_1, q_2, u_2) \to \nu(q_1, q_2 + \delta q, u_2)$. This has been achieved by separately altering each of f_c, t_d and A_0. Second, variations in ν have been analyzed when noise $\eta(t)$ is added to the drive $u_2(t)$ and $q_1(t)$ and $q_2(t)$ are kept unchanged: $\nu(q_1, q_2, u_2) \to \nu(q_1, q_2, u_2 + \eta)$.

The dependency of ν on f_c (the frequency of q_2) is shown in Fig. 11. The index ν is largest when $f_c = 2$, since the system was optimized at that point. When the frequency changes, ν becomes smaller. For example, from $f_c = 2$ to $f_c = 1.98$, there is a change from $\nu = 2.07$ to $\nu = 0.31$. In Fig. 12, the outputs $y[q_1, u_2](t)$ and $y[q_2, u_2](t)$ are shown for $f_c = 1.98$ and $u = u_2$. One can see there is no clear separation between the outputs. The ν was also calculated for broader range of f_c values (Fig. 13). The figure shows that there are f_c values (except for $f_c = 2$) with a relatively high ν, e.g. when $f_c = 1.43$. The outputs for the two environmental conditions for $f_c = 1.43$ are shown in Fig. 14. The outputs are sufficiently distinct to distinguish between the two environmental conditions.

The dependence of ν on t_d (the time-shit parameter of q_2) is shown in Fig. 15. A non zero t_d means that there is a phase difference between u_2 and q_2. As the parameter t_d is altered, the quality of sensing oscillates. These oscillations occur since the synchronization between the drive u_2 and the environment q_2 is lost and regained periodically as t_d is altered. Note that the largest values for ν occur at the multiples of $\Delta t_d = 0.1$ (including $t_d = 0$ since the system was optimized at that point). Interestingly, the period of these peaks differs from the period of q_2.

The dependence of ν on the A_0 parameter is shown in Fig. 16. As A_0 increases in the range [0,1] the ν index increases too, and attains a maximum value for $A_0 = 1.3$. Note that the sensor has been optimized for $A_0 = 1.0$. It is somewhat surprising that ν increases in such a way. The quality of sensing has deteriorated when $A_0 < 1.0$ but has not deteriorated when $A_0 > 1.0$. This finding shows that, in a situation when the lower bound on the environmental signal is known, it might be advantageous to optimize the one-memristor sensor for the lowest expected amplitude.

The dependency of the index ν was also calculated with regard to two types of variations in the input signal $u_2 \to u_2 + \eta$ with $\eta = u_0 + \eta'$. The variable u_0 describes uniform shifts. The variable η' represents a genuine noise, a stochastic variable with zero mean and a finite standard deviation σ. Since the uniform shifts play a crucial role we consider the cases where $u_0 \neq 0$: For example, the sign of the drive is important because it affects whether the memristance will be driven to the region of the maximum or the minimum value. In fact, as the examples with u_1 and u_2 show, this balance is actively exploited for sensing. The presence of $u_0 \neq 0$ can destroy the balance.

This behavior is illustrated in Fig. 17 which depicts the assumed probability distribution function of η. The mean of the distribution is u_0. The standard deviation is given by $\sigma = \chi/\sqrt{12}$ where the parameter χ defines the width of the uniform distribution. The variation of the index ν with regard to u_0 and χ is analyzed in Figs. 18 and 19. As anticipated sufficiently large values of $|u_0|$ and χ deteriorate the quality of sensing.

Key result: The quality of sensing deteriorates regardless whether the environmental conditions are changed or the drive is altered. It seems that the quality of sensing depends critically on the:

- the choices of the drive,
- the environment frequency
- and the shifts in time

This indicates that the synchronization between the environment signal and the drive signal is important for the device operation and should be carefully maintained.

4. Conclusions

The possibility of using environment sensitive memristor networks for advanced sensing applications has been investigated theoretically, in the context of the SWEET sensing setup. The key challenge was to demonstrate the feasibility of realizing the SWEET sensing setup where the reservoir (the state waver) is a memristor network. For simplicity reasons, the simplest possible memristor network has been considered (a single environment sensitive memristor unit), and only two distinct environmental conditions have been assumed q_1 and q_2. The key challenge was to find a drive u that can be used to operate the sensor under these environmental conditions, in a way that the device outputs 0 (1) when exposed to environment q_1 (q_2). This is possible if the drive can achieve the phase space separation (in the space of weaver states). A relatively simple model of the interaction between the environment and the memristor has been assumed. It has been shown that indeed the desired drive u can be found. Accordingly, memristor networks have the potential to be used for realizing SWEET sensors.

Drive 1: In Section 3.1, a drive $u = u_1$ was guessed on purely intuitive basis such that the memristance $R[q, u]$ is driven to two different regions for the two environmental conditions q_1, q_2. When the environmental condition is constant q_1, the memristance $R[q_1, u]$ stays in the region around the initial value. On the other hand, when the environmental condition has a square wave form q_2 with the same phase and frequency as the drive u, then, the memristance value is driven to the region of the maximum possible value. However, if there is a slight loss of synchronization between u and q_2, then, the memristance $R[q_2, u]$ would not be driven to the region of the maximum value and would stay in the region around the initial value. Therefore, in this example an exact synchronization between q_2 and u is needed. Such an exact synchronization would be unlikely to be achieved under real experimental conditions. Another limitation of such a method is that it is unlikely that a useful drive could be guessed for large memristor networks. This example was discussed only to illustrate the basic mechanisms of using one memristor for sensing.

Drive 2: In section 3.2 the genetic algorithm optimization was used to identify a solution of the SWEET sensing problem being illustrated in Fig. 3. The procedure is generic and could be used for more complex reservoirs, e.g. memristor networks with a large number of memristors. The algorithm identified a drive $u = u_2$ such

that $R[q_1, u]$ was driven to the region of the maximum possible value and $R[q_2, u]$ was driven to the region of the minimum possible value. The fact that $R[q, u]$ is driven to two different regions for q_1 and q_2 respectively is the reason why such a sensor would distinguish between the environmental conditions.

The strength of the drive signal controls the trade-off between two important behaviors. On one hand, the size of the drive signal affects the quality of sensing. Larger drives lead to more volatile outputs, which imply worse phase separation as the system crosses easier into different regions of the configuration space. On the other hand, stronger drive signals improve (shorten) the response time of the sensor. The speed of response might be important in on-line sensing applications where the goal is to detect fast environmental changes. It is an interesting observation that the trade-off between the accuracy and the speed of response can be regulated to some extent by choosing the strength of the drive signal.

In section 3.5 the quality of sensing index ν has been suggested as a measure of the phase space separation and the related sensing performance of the device. The deterioration of the quality of sensing was analyzed with regard to the changes in environmental signals, and the variations of the drive signal due to the presence of noise. Since the sensor has been optimized for a specific set of conditions, as expected, for most of the induced changes the quality of sensing deteriorates. (For example, index ν has increased when the environmental signal q_2 got stronger.) The key observation is that the quality of sensing worsens if the synchronization is lost between the drive and the environmental signal. An interesting question is whether it is possible to identify a drive u that can result in a more robust sensing, at least within certain range of changes, so that the synchronization is maintained in some way.

The question is whether it may be possible to sense more than two environmental conditions ($E > 2$) with the one-memristor network, i.e. whether it is possible to achieve the desired phase space separation. This is still under investigation. For example, Fig. 20 (drawn by hand) is an illustration of the desired phase space structure. Regardless of whether a drive can be found that would result in such a structure it should be noted that the use of the linear readout layer has its limitations. The linear readout cannot separate one-dimensional configuration space, since the equation system

$$\begin{aligned} w_1 \cdot R_\mathrm{a} + w_0 &= \phi[q_\mathrm{a}] \\ w_1 \cdot R_\mathrm{b} + w_0 &= \phi[q_\mathrm{b}] \\ w_1 \cdot R_\mathrm{c} + w_0 &= \phi[q_\mathrm{c}] \end{aligned} \tag{23}$$

does not have a solution in the coefficients w_0 and w_1. Only if there is a suitable alignment of memristance values then the linear readout could work (provided the values $\phi[q_a]$, $\phi[q_b]$, and $\phi[q_c]$ could be freely chosen to match the resistance ordering).

The work can be extended in several ways. It might be possible to generalize this construction to assess the sensing performance for SWEET sensors realized with more complicated networks. Potentially, larger and more complex memristor networks can be used for sensing a larger number of environmental conditions or for investigating the sensing of more complicated environments with more accuracy. In the situations where there is an upper limit for the number of memristor components, the important question is whether the performance of the sensor can be optimized by changing the connectivity pattern of the network.

Further, the question is whether the quality of sensing depends strongly on the environment-memristor interaction model. Additionally, the uncertain variability

of parameters such as V_{thr} and β among different memristor devices could impose limitations regarding the optimum drive and deteriorate the quality of sensing. However, they could also increase the sensing power of the device (by enlarging the complexity of the configuration space). The effects of such variations could be considered in future work.

The recent development of the memristor simulators [19] could be leveraged to test experimentally the ideas presented in this study. In particular, the features that could be exploited is the greater flexibility and possible better control of the manufacturing process of the emulator devices. However, this is only possible if the the emulator contains at least one environment-sensitive component.

The optimization algorithm could be improved too (e.g. by allowing for explicit optimization of the time shift, and overall signal amplitude, or by trying to find signals that are resilient to the loss of the synchronization). Despite the fading memory property, for shorter time intervals T, the choice of the initial condition could play a role, since the response time of the sensor might change. This might be important in on-line sensing scenarios and affect how the sensor operates when there are rapid environmental changes.

Acknowledgements

The work was supported by an FET Open HORIZON 2020 grant RECORD-IT (Reservoir Computing with Real-time Data for future IT) #664786. The authors wish to acknowledge a series of discussions with all partners in the RECORD-IT consortium. These discussions contributed greatly in formulating the ideas presented in this work.

References

[1] G.E. Moore, *Cramming more components onto integrated circuits (reprinted from electronics, pg 114-117, april 19, 1965)*, Proceedings of the Ieee 86 (1998), pp. 82–85.

[2] T. Sienko, *Molecular Computing*, MIT Press, 2003, URL `https://books.google.se/books?id=RleBQgAACAAJ`.

[3] M. Hiratsuka, T. Aoki, and T. Higuchi, *Enzyme transistor circuits for reaction-diffusion computing*, Ieee Transactions on Circuits and Systems I-Fundamental Theory and Applications 46 (1999), pp. 294–303.

[4] A. Hjelmfelt, E.D. Weinberger, and J. Ross, *Chemical implementation of finite-state machines*, Proceedings of the National Academy of Sciences of the United States of America 89 (1992), pp. 383–387.

[5] C. Bennett, A. Jesorka, G. Wendin, and Z. Konkoli, *ADVANCES IN UNCONVENTIONAL COMPUTATION*, in *Advances in Unconventional Computation*, A. Adamatzky, ed., Vol. Vol 2. Prototypes and algorithms, Springer, 2016.

[6] E. Bergh and Z. Konkoli, *ADVANCES IN UNCONVENTIONAL COMPUTATION*, in *Advances in Unconventional Computation*, A. Adamatzky, ed., Vol. Vol 2. Prototypes and algorithms, Springer, 2016.

[7] Z. Konkoli, *ADVANCES IN UNCONVENTIONAL COMPUTATION*, in *Advances in Unconventional Computation*, A. Adamatzky, ed., Vol. Vol 1. Theory, Springer, 2016.

[8] H. Jaeger, *The "echo state" approach to analysing and training recurrent neu-*

ral networks, Report GDM Report 148 (contains errors), German national research center for information technology, 2001.

[9] H. Jaeger and H. Haas, *Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication*, Science 304 (2004), pp. 78–80.

[10] W. Maass, T. Natschlager, and H. Markram, *Real-time computing without stable states: A new framework for neural computation based on perturbations*, Neural Computation 14 (2002), pp. 2531–2560.

[11] H.M. T. Natschl?ger W. Maass, *The "liquid computer": A novel strategy for real-time computing on time series (special issue on foundations of information processing)*, TELEMATIK 8 (2002), pp. 39–43.

[12] M. Lukosevicius, H. Jaeger, and B. Schrauwen, *Reservoir computing trends*, KI - Könstliche Intelligenz 26 (2012), pp. 365–371.

[13] Z. Konkoli, *The sweet algorithm: generic theory of using reservoir computing for advanced sensing applications*, Journal of Parallel Emergent and Distributed Systems (2016).

[14] Z. Konkoli, *The state weaving environment-echo tracker (sweet/record-it) sensing setup and algorithm (patent pending)* (2016).

[15] Y.V. Pershin and M. Di Ventra, *Experimental demonstration of associative memory with memristive neural networks*, Neural Networks 23 (2010), pp. 881–886.

[16] M.D. Ventra, Y.V. Pershin, L.O. Chua, and F. IEEE, *Circuit elements with memory: memristors, memcapacitors, and meminductors*, Proceedings of the IEEE 97 (2009), pp. 1717 – 1724.

[17] E. Zhitlukhina and M. Belogolovskii, *Memristor, a nano-scaled element for the computer memory: A mini-review with some new results for an ac-driven memristor*, Journal of Photonic Materials and Technology 1 (2015), pp. 27 – 32.

[18] Z. Konkoli and G. Wendin, *A generic simulator for large networks of memristive elements*, NANOTECHNOLOGY 24 (2013).

[19] I. Vourkas, A. Abusleme, V. Ntinas, G.C. Sirakoulis, and A. Rubio, *A digital memristor emulator for fpga-based artificial neural networks*, 2016 IEEE Int Verification and Security Workshop (IVSW), Sant Feliu de Guixols, Catalunya, Spain, July 4-6 , pp. 65 – 68.

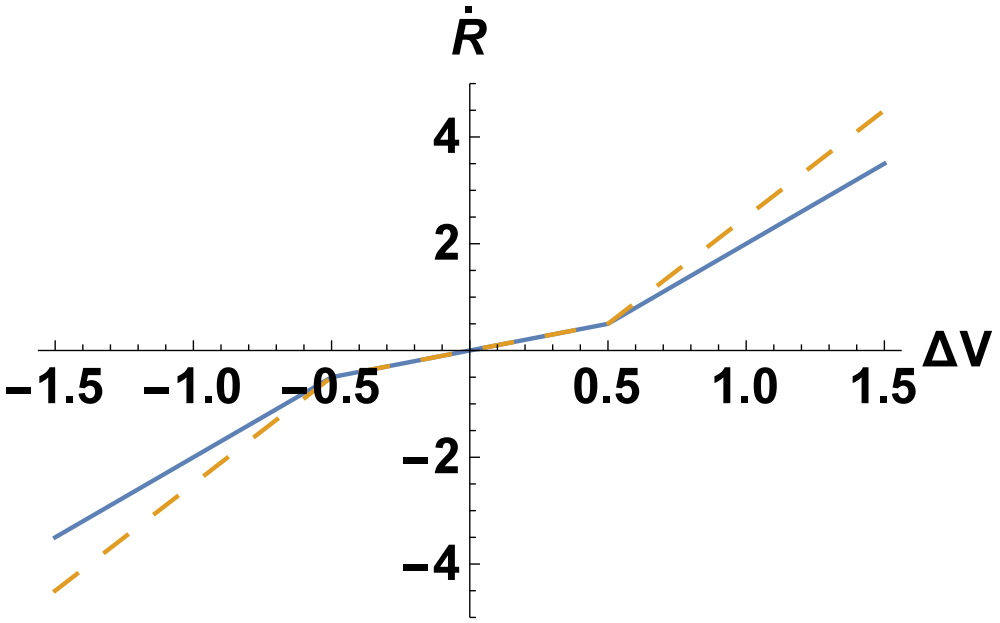

Figure 1. The form of function f given in Eq. (6) plotted with $V_{\rm thr} = 0.5$, $\alpha = 1$ for two distinct environmental conditions $q = 3$ (full line) and $q = 4$ (dashed line).

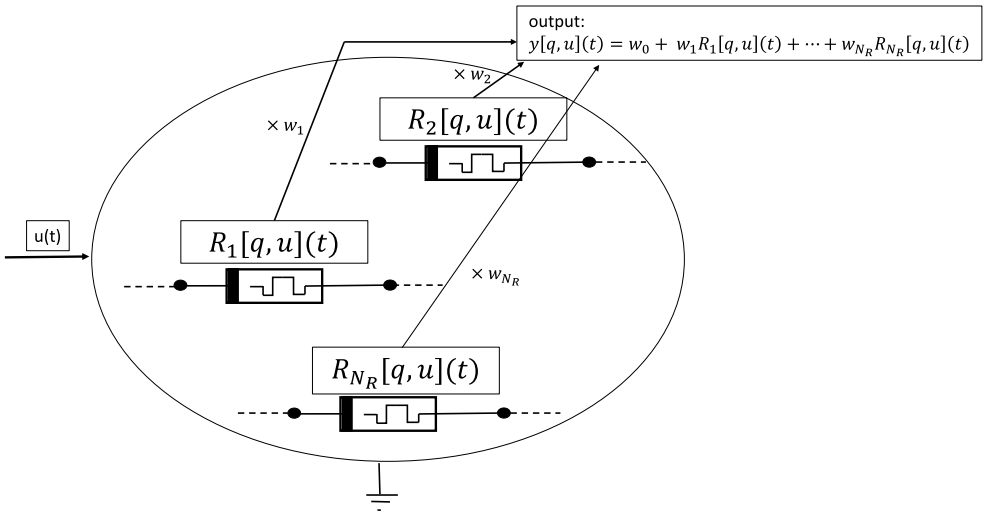

Figure 2. A memristor network consists of $N_{\rm R}$ memristors. The network is driven by a signal u under an environmental condition q. The output is a weighted sum of the memristances.

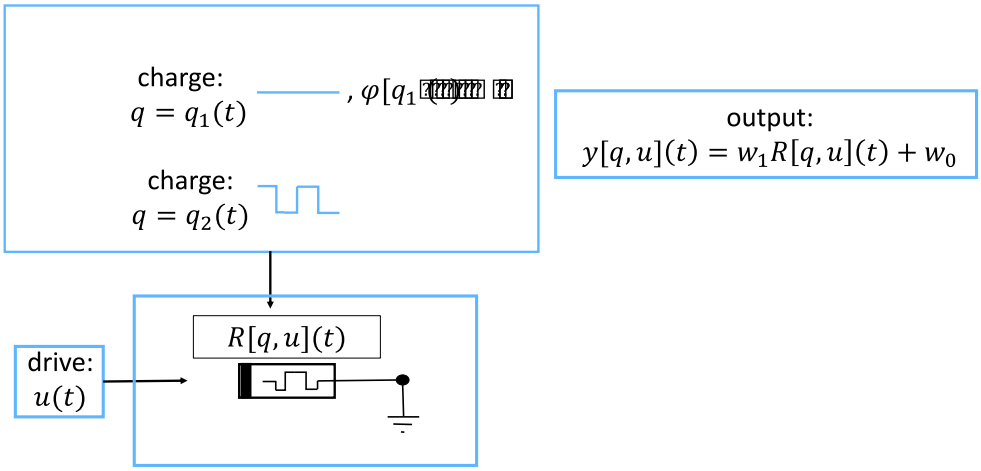

Figure 3. a) The sensor consists of a single memristor driven by a signal $u(t)$. Two environmental conditions are considered; case 1: a constant environmental signal $q = q_1(t)$; case 2: a varying environmental signal with a square-wave form $q = q_2(t)$. The function $\phi[q](t)$ is the sensing goal. The sensor should output 0 (1) when exposed to q_1 (q_2). b) The output $y[q,u](t)$ is a re-scaled memristance with a shift. The optimization problem: find a drive $u(t)$ such that $y[q_1,u](t) \approx \phi[q_1](t)$ and $y[q_2,u](t) \approx \phi[q_2](t)$ for every time t.

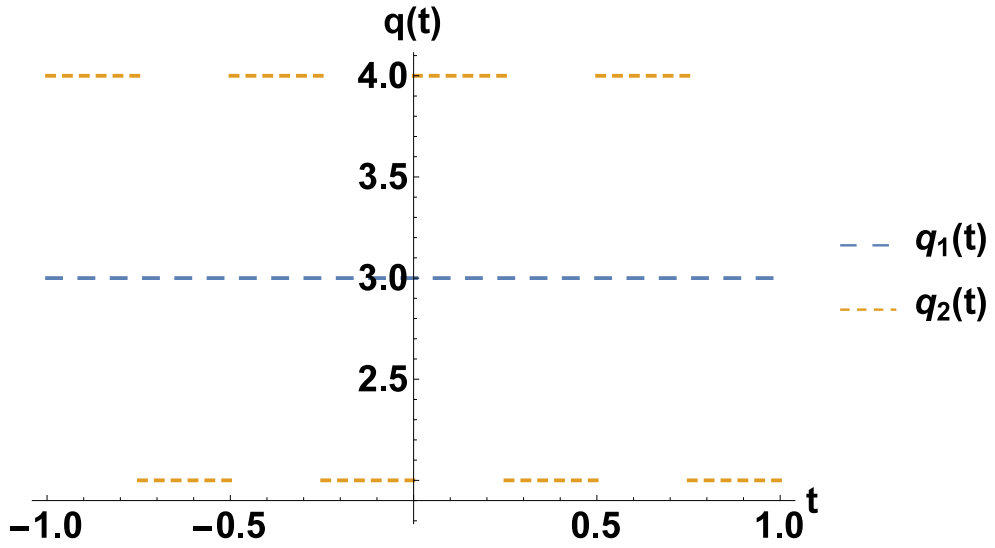

Figure 4. A graphical representation of the environmental signals. The environment $q(t) = q_1(t)$ is assumed to be inert. The environment $q(t) = q_2(t)$ is assumed to be changing in time as a square wave.

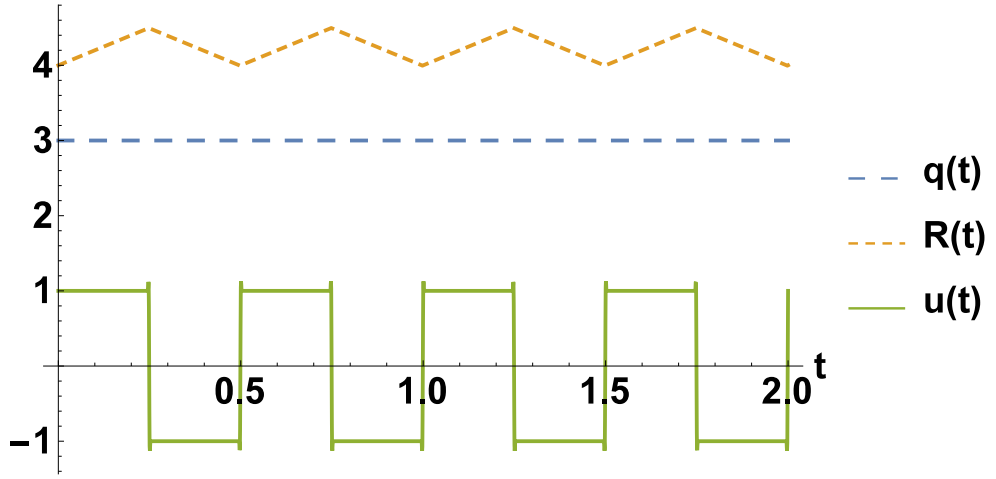

Figure 5. The time-dependence of the resistance $R(t) = R[q_1, u_1](t) \equiv R_{11}(t)$ when the sensor is driven by the signal $u(t) = u_1(t)$ and exposed to environment $q = q_1(t)$. There is no overall change in $R(t)$ over one period of $u(t)$: When $u(t)$ is positive $R(t)$ increases and when $u(t)$ is negative $R(t)$ decreases, always by the same amount (since u has equal positive and negative amplitudes). Note that if the drive $u(t)$ had a positive/negative mean value, $R(t)$ would be gradually increasing/decreasing.

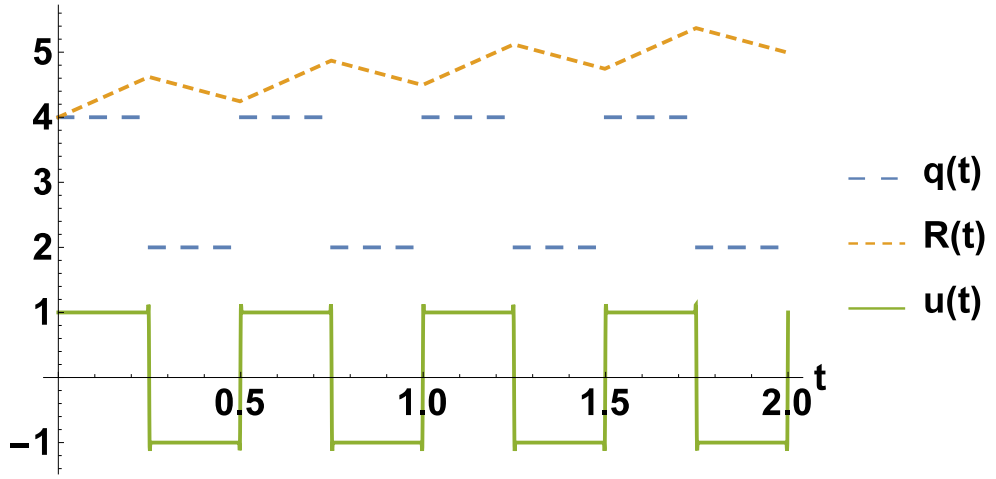

Figure 6. The time-dependence of the resistance $R(t) = R[q_2, u_1](t) \equiv R_{21}(t)$ when the sensor is driven by the signal $u(t) = u_1(t)$ and exposed to $q = q_2(t)$. There is an overall increase in $R(t)$ over one period of $u(t)$. When $u(t)$ is positive $R(t)$ increases and when $u(t)$ is negative $R(t)$ decreases. $R(t)$ increases with a larger rate since the increase happens with a larger slope ($q(t) = 4$) than the decrease ($q(t) = 2$). In this way R_{max} is gradually reached.

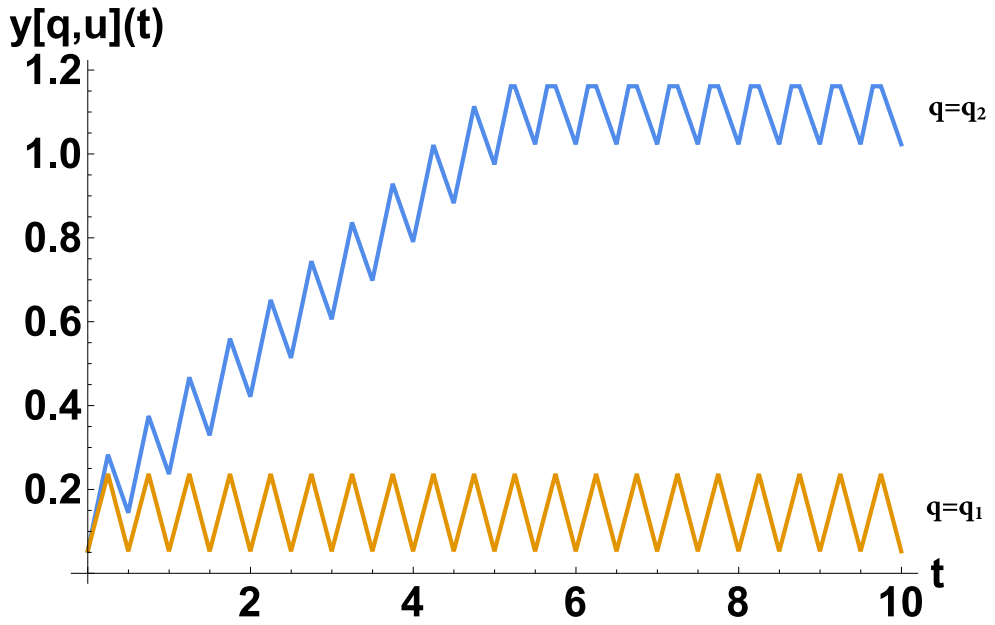

Figure 7. The output $y[q,u](t)$ of the one-memristor network for the two environmental conditions $q(t) = q_1, q_2$ and the drive $u(t) = u_1(t)$. The output was driven to the two different sensing goals: around $\phi[q] = 0.0$ for $q = q_1$ and $\phi[q_2] = 1.0$ for $q = q_2$.

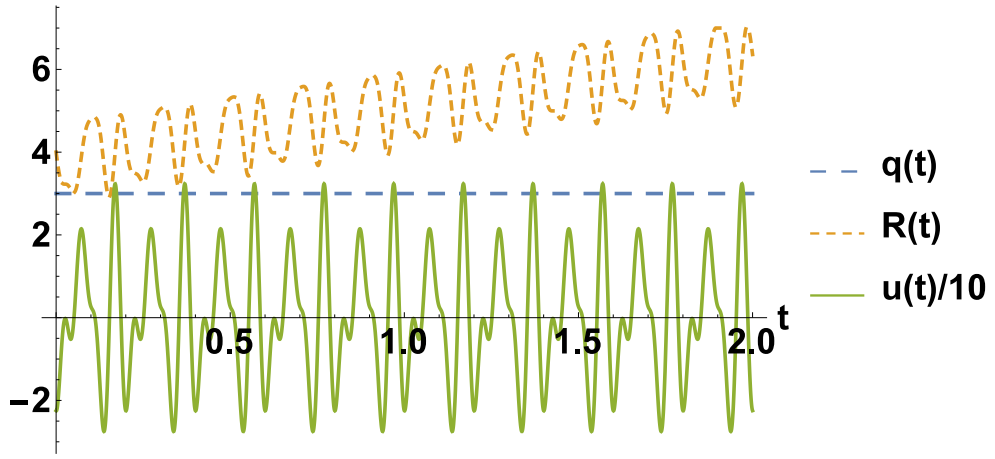

Figure 8. Resistance $R(t) = R[q_1, u_2](t) \equiv R_{12}$ as a function of time under the drive $u(t) = u_2(t)$ and the environmental condition $q = q_1(t)$. When $u(t)$ is positive $R(t)$ increases and when $u(t)$ is negative $R(t)$ decreases. Therefore, since $u(t)$ has a positive mean value, $R(t)$ gradually increases over periods of $u(t)$, and R_{max} is reached.

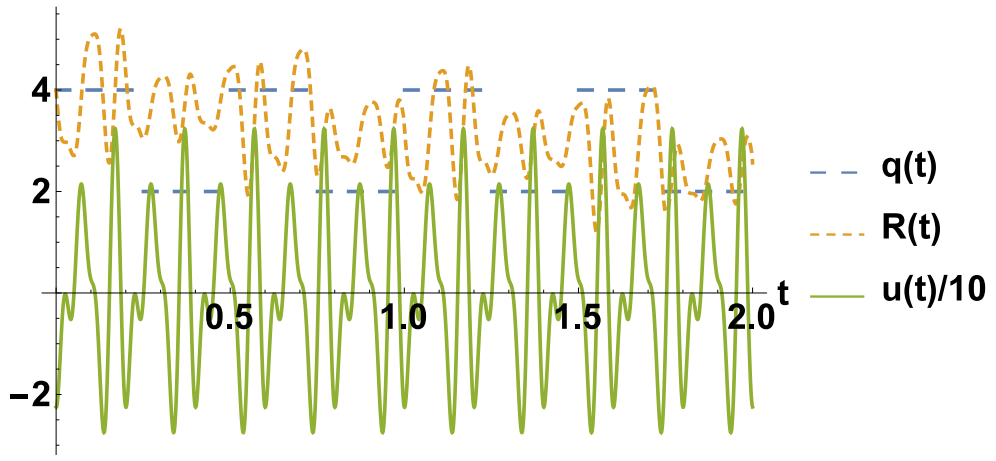

Figure 9. Resistance $R(t) = R[q_2, u_2] \equiv R_{22}$ under $u(t) = u_2(t)$ and $q = q_2(t)$. When $u(t)$ is positive (negative) $R(t)$ increases (decreases). The rate of change of $R(t)$ depends on whether the environmental signal is large ($q = 4$) or small ($q = 2$). When $q = 4$, $R(t)$ has larger rates of change than when $q = 2$. The synchronization of the drive $u(t)$ and the environmental condition $q(t)$ is such that there is an overall decrease in $R(t)$ and it is driven to the region around R_{min}.

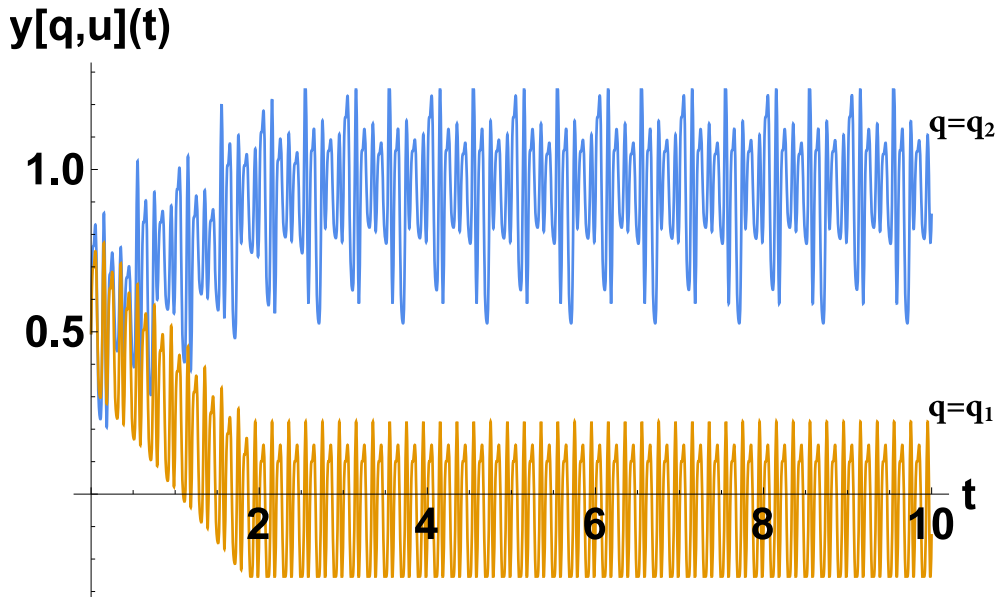

Figure 10. The output $y[q, u](t)$ of the one-memristor sensor for the two environmental conditions $q(t) = q_1, q_2$ and the drive $u(t) = u_2(t)$. The output mimics the sensing goal: $\phi[q] \approx 0.0$ for $q = q_1$ and $\phi[q] \approx 1.0$ for $q = q_2$.

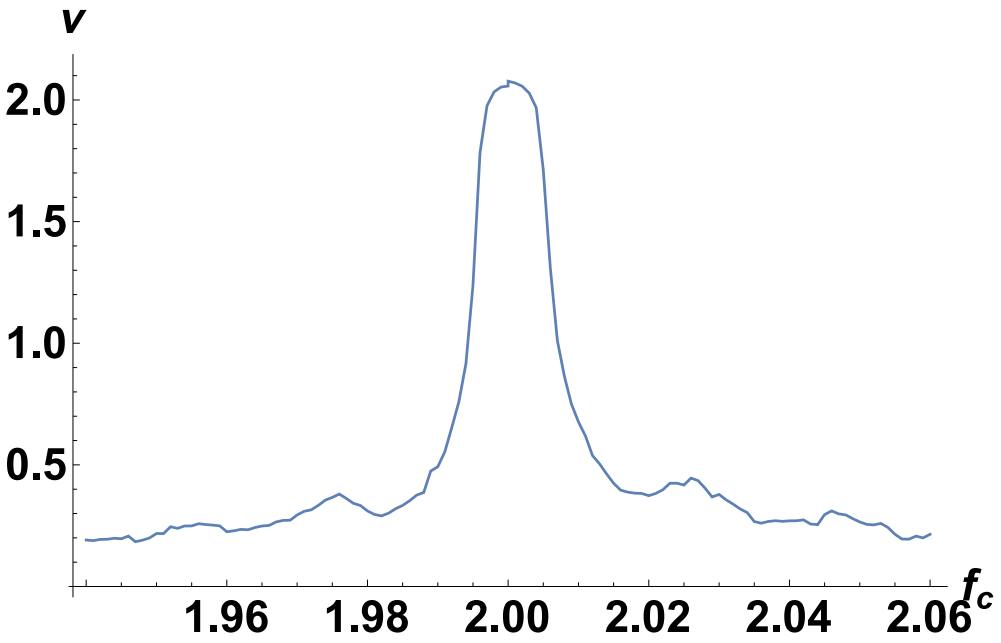

Figure 11. The quality of sensing index ν with respect to the parameter f_c of q_2 and $u = u_2$.

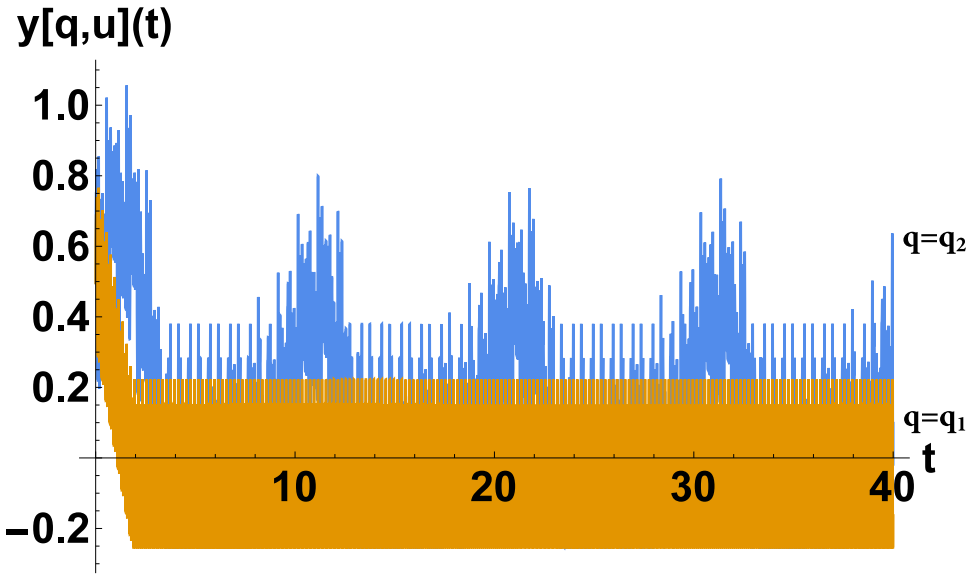

Figure 12. The output $y[q, u](t)$ of the one-memristor sensor for the two environmental conditions when $u = u_2$ and the parameter f_c of q_2 has been changed to $f_c = 1.98$ instead of $f_c = 2.0$. The output $y[q, u](t)$ mimics the sensing goal for $q = q_1$ but not for q_2. The phase separation is weaker: $\nu \approx 0.3$.

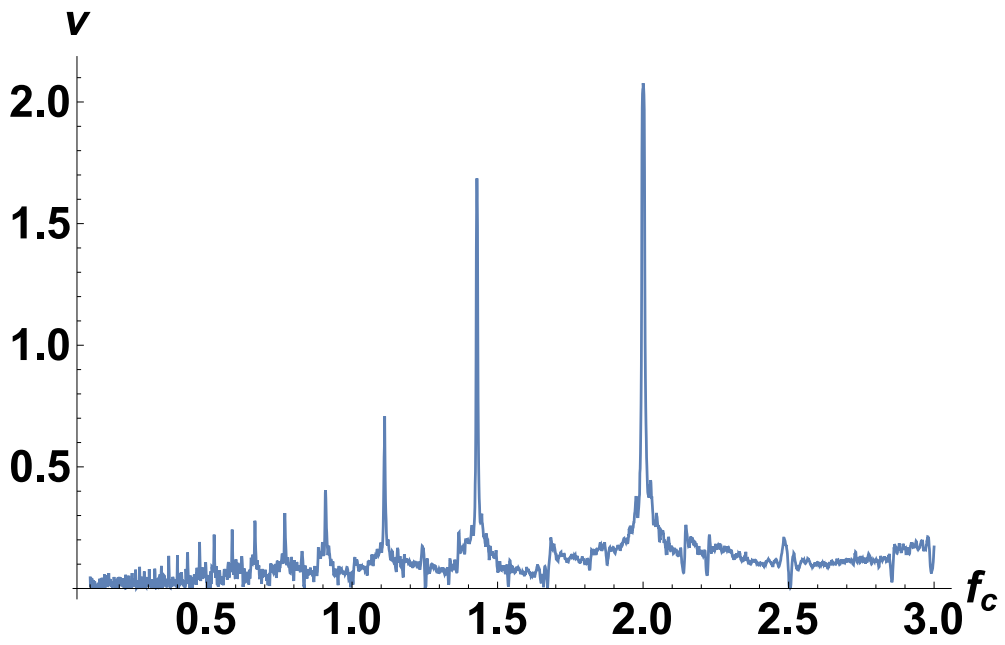

Figure 13. The quality of sensing index ν with respect to a broad range of the parameter f_c of q_2 and $u = u_2$. ν is largest for $f_c = 2.0$. However, there were other values of f_c with a high index ν (e.g. $f_c = 1.43$).

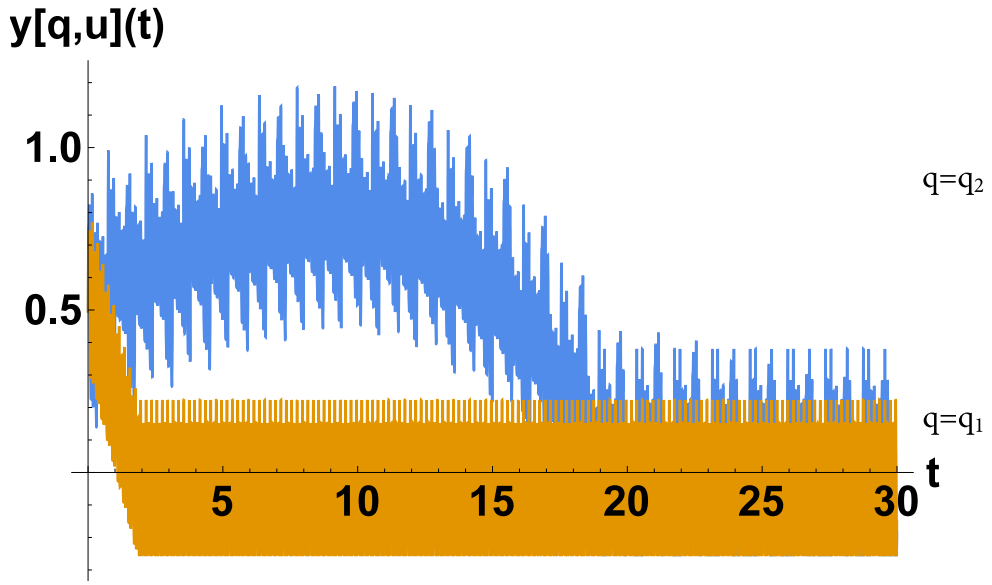

Figure 14. The output $y[q, u](t)$ of the one-memristor sensor for the two environmental conditions with f_c being altered from $f_c = 2.0$ to $f_c = 1.43$. The sensing is possible (distinction between the two outputs exists) within $\Delta t \approx 12$ after the beginning of the simulation. This behavior results in a large index $\nu \approx 1.7$.

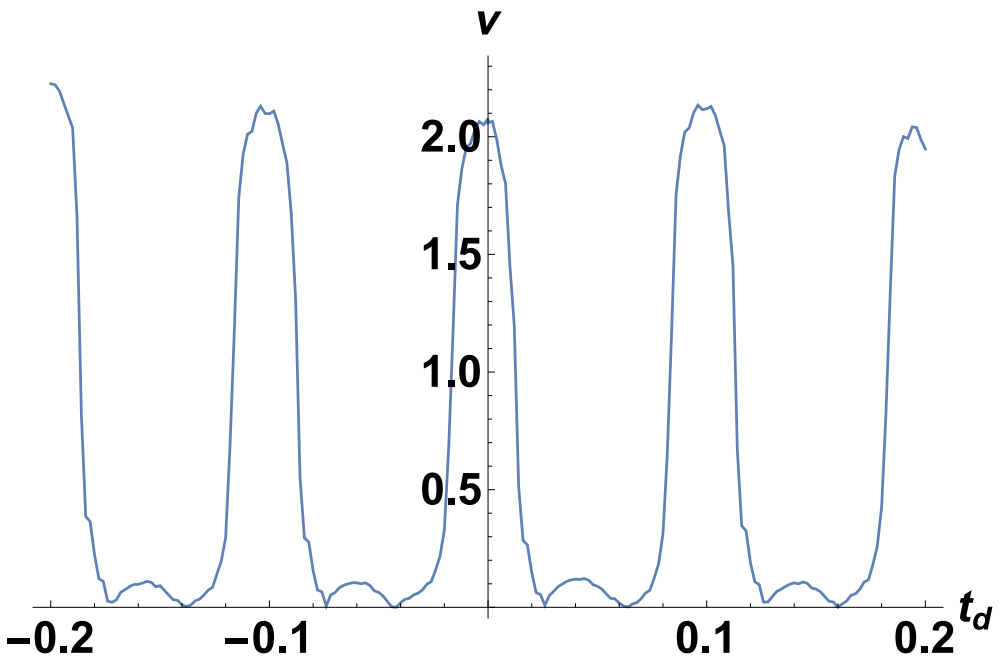

Figure 15. The quality of sensing index ν with respect to the parameter t_{d} of q_2 and $u = u_2$.

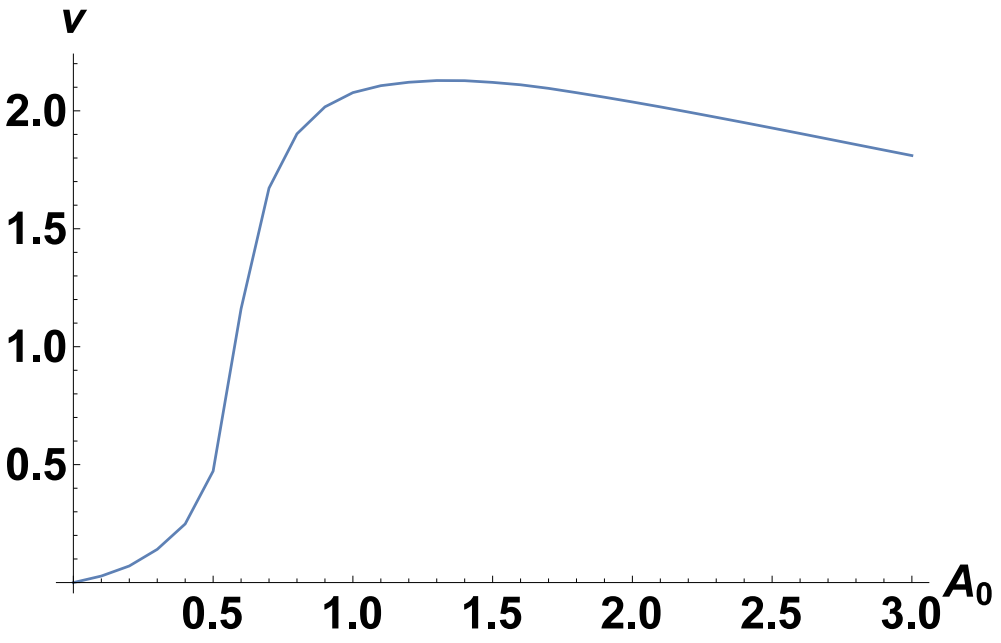

Figure 16. The quality of sensing index ν with respect to the parameter A_0 of q_2 for $u = u_2$.

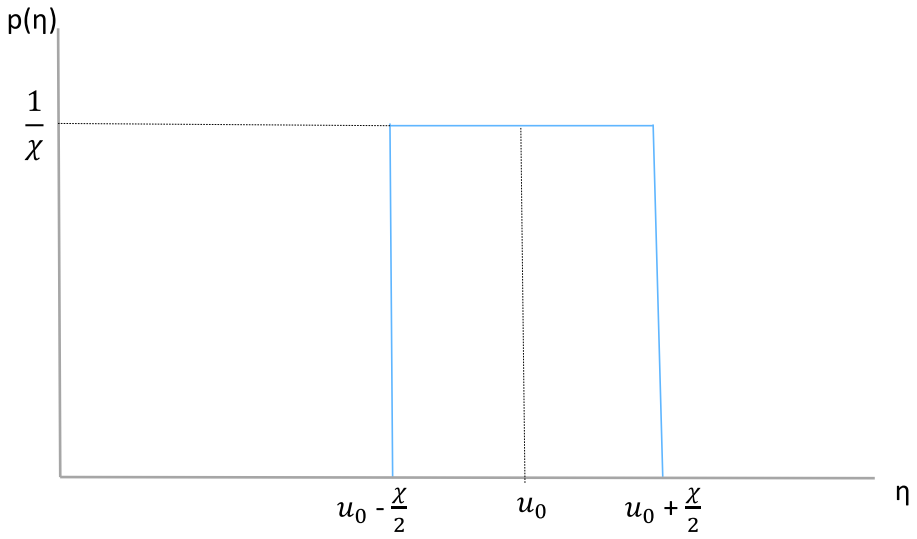

Figure 17. The probability density function of the noise η. η is uniformly distributed with a mean value u_0 in an interval equal to χ.

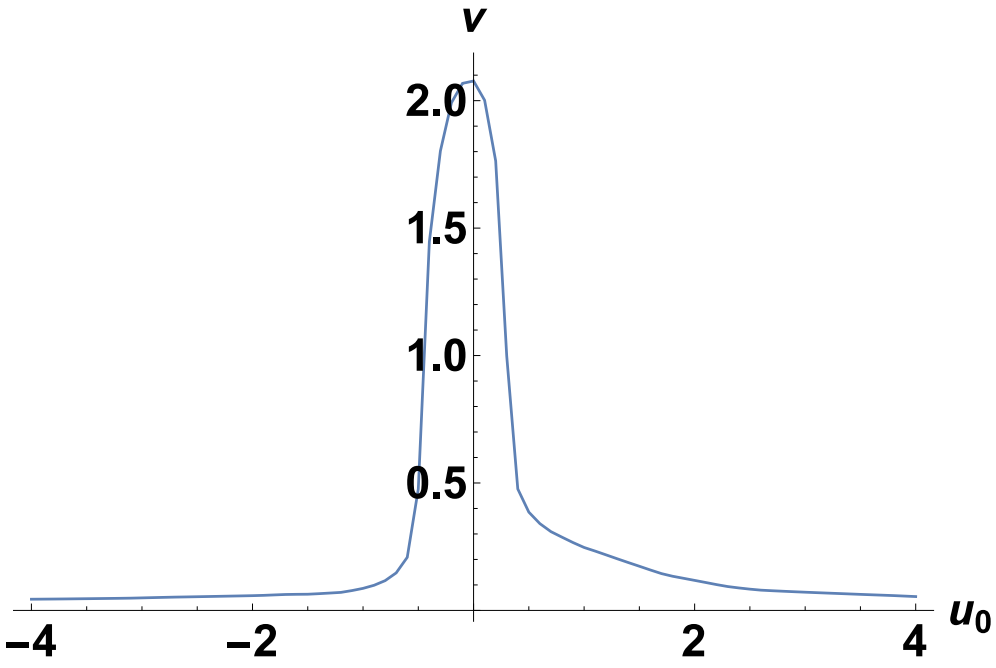

Figure 18. The Quality of sensing index ν for $u(t) = u_2(t) + u_0$.

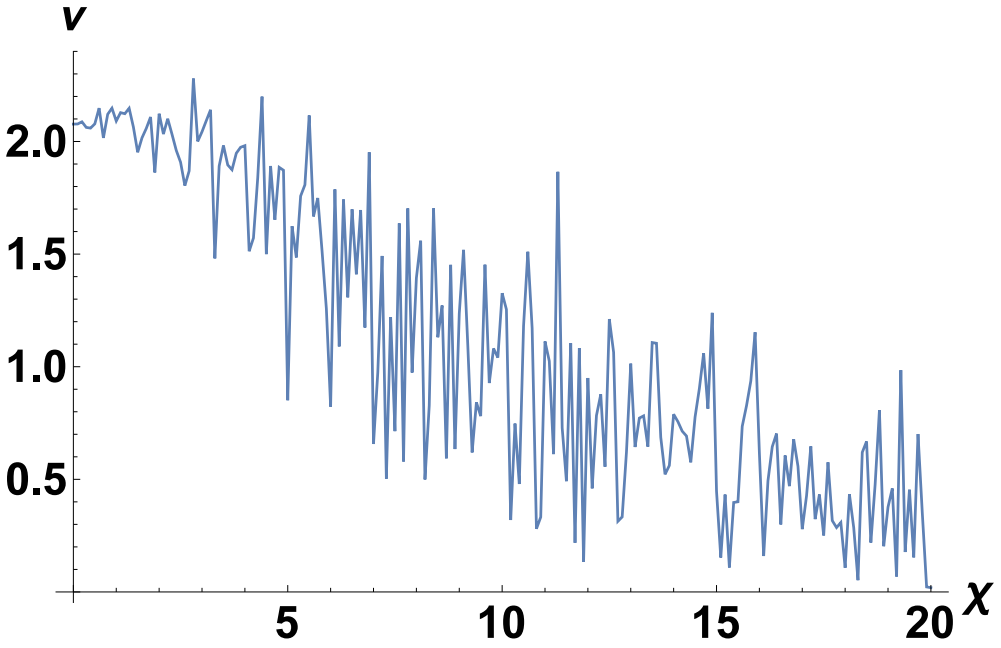

Figure 19. The Quality of sensing index ν as a function of χ. Parameter χ controls the standard deviation of the noise. The index ν gradually decreases as the drive signal becomes more random.

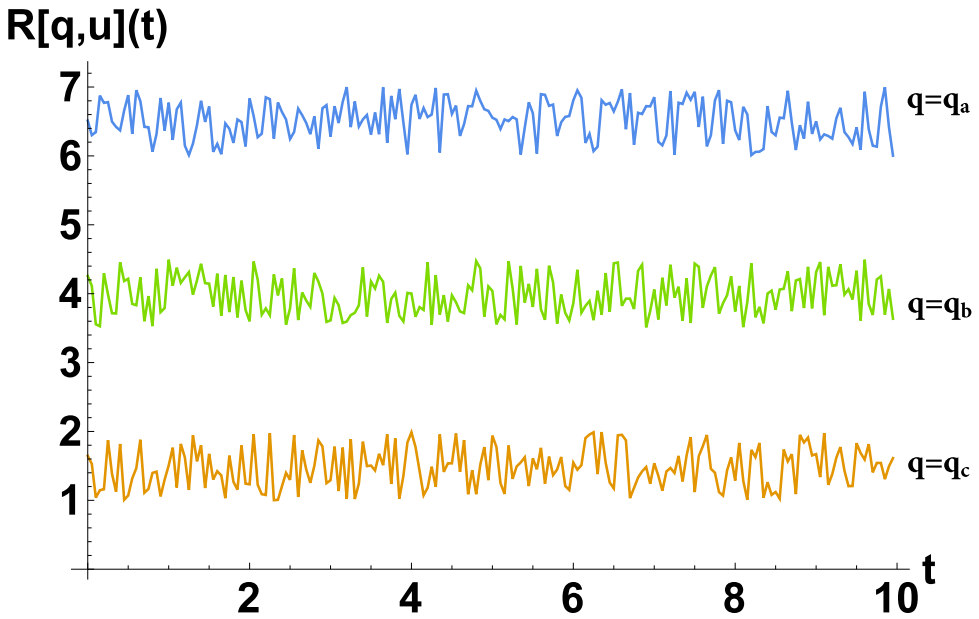

Figure 20. A sketch of the phase space separation (drawn by hand) that is necessary for using the one-memristor sensor under three environmental conditions. A drive u should be found such that the resistance $R(t)$ is driven to three separate regions of the configuration space $R(t) \approx R_a$, $R(t) \approx R_b$ and $R(t) \approx R_c$ under the environmental condition $q = q_a$, $q = q_b$ to $q = q_c$ respectively.

Appendix A. Dimensional analysis

Every measurable quantity has to be expressed in specific units. To make the numerical analysis easier, all variables are resealed to obtain dimensionless quantities. The symbols with subscript 0 define the units of the variables, and the ones with the tilde represent dimensionless variables used in the numerical simulations. The following scaling is used for the numerical work:

$$\begin{aligned} t &= t_0 \cdot \tilde{t} \\ R(t) &= R_0 \cdot \tilde{R}(\tilde{t}) \\ \Delta V(t) &= V_0 \cdot \tilde{\Delta V}(\tilde{t}) \\ V_{\text{threshold}} &= V_0 \cdot \tilde{V}_{\text{threshold}} \\ \beta &= \beta_0 \cdot \tilde{\beta} \\ \alpha &= \beta_0 \cdot \tilde{\alpha} \end{aligned} \tag{A1}$$

The physical memristor model is defined by

$$\begin{aligned} \frac{\partial R(t)}{\partial t} = \beta \cdot \Delta V(t) + \frac{1}{2} \cdot (\alpha - \beta) \cdot ([\Delta V(t) + V_{\text{threshold}}] \\ -[\Delta V(t) - V_{\text{threshold}}]) \end{aligned} \tag{A2}$$

By using Eq. (A1) in (A2) gives

$$\begin{aligned} \frac{\partial (R_0 \cdot \tilde{R}(\tilde{t})}{\partial t} = \beta_0 \cdot \tilde{\beta} \cdot V_0 \cdot \tilde{\Delta V}(\tilde{t}) + \\ \frac{1}{2} \cdot (\beta_0 \cdot \tilde{\alpha} - \beta_0 \cdot \tilde{\beta}) \cdot ([V_0 \cdot \tilde{\Delta V}(\tilde{t}) + V_0 \cdot \tilde{V}_{\text{threshold}}] \\ -[V_0 \cdot \tilde{\Delta V}(\tilde{t}) - V_0 \cdot \tilde{V}_{\text{threshold}}) \end{aligned} \tag{A3}$$

Using the standard calculus, shown here for pedagogical reasons, one obtains the following expressions:

$$\begin{aligned} \frac{\partial (R_0 \cdot \tilde{R}(\tilde{t}))}{\partial t} = R_0 \cdot \frac{\partial (\tilde{R}(\tilde{t}))}{\partial t} = R_0 \cdot \frac{\partial (\tilde{R}(\tilde{t}))}{\partial \tilde{t}} \cdot \frac{\partial \tilde{t}}{\partial t} = \\ R_0 \cdot \frac{\partial (\tilde{R}(\tilde{t}))}{\partial \tilde{t}} \cdot \frac{1}{t_0} \end{aligned} \tag{A4}$$

$$\begin{aligned} R_0 \cdot \frac{\partial (\tilde{R}(\tilde{t}))}{\partial \tilde{t}} \cdot \frac{1}{t_0} = \beta \cdot V_0 \cdot (\tilde{\beta} \cdot \tilde{\Delta V}(\tilde{t}) + \\ \frac{1}{2} \cdot (\tilde{\alpha} - \tilde{\beta}) \cdot ([\tilde{\Delta V}(\tilde{t}) + \tilde{V}_{\text{threshold}}] \\ -[\tilde{\Delta V}(\tilde{t}) - \tilde{V}_{\text{threshold}}]) \end{aligned} \tag{A5}$$

$$\frac{\partial(\tilde{R}(\tilde{t}))}{\partial\tilde{t}} = \frac{\beta_0 \cdot V_0 \cdot t_0}{R_0} \cdot (\tilde{\beta} \cdot \tilde{\Delta V}(\tilde{t}) + \frac{1}{2} \cdot (\tilde{\alpha} - \tilde{\beta}) \cdot ([\tilde{\Delta V}(\tilde{t}) + \tilde{V_{\text{threshold}}}] - [\tilde{\Delta V}(\tilde{t}) - \tilde{V_{\text{threshold}}}]) \tag{A6}$$

We choose β_0 as

$$\beta_0 = \frac{R_0}{t_0 \cdot V_0} \tag{A7}$$

since this eliminates the factor that contains the variables with a dimension (V_0, R_0, and t_0). This defines the units of the variables α and β. Finally, by combining Eqs. (A3-A7) results in

$$\frac{\partial(\tilde{R}(\tilde{t}))}{\partial\tilde{t}} = \tilde{\beta} \cdot \tilde{\Delta V}(\tilde{t}) + \frac{1}{2} \cdot (\tilde{\alpha} - \tilde{\beta}) \cdot ([\tilde{\Delta V}(\tilde{t}) + \tilde{V_{\text{threshold}}}] - [\tilde{\Delta V}(\tilde{t}) - \tilde{V_{\text{threshold}}}]) \tag{A8}$$

which is the desired equation that contains dimensionless quantities only. This equation was used to implement the algorithm on the computer.