PHYSICAL REVIEW E **84**, 060102(R) (2011)

Computation as an emergent feature of adaptive synchronization

M. Zanin,[1,2,*] D. Papo,[1] I. Sendiña-Nadal,[1,3] and S. Boccaletti[1]

[1]*Center for Biomedical Technology, Technical University of Madrid, Campus Montegancedo, 28223 Pozuelo de Alarcón, Madrid, Spain*

[2]*Innaxis Foundation & Research Institute, José Ortega y Gasset 20, E-28006 Madrid, Spain*

[3]*Complex Systems Group, Universidad Rey Juan Carlos, 28943 Fuenlabrada, Madrid, Spain*

(Received 8 August 2011; revised manuscript received 20 October 2011; published 20 December 2011)

We report on the spontaneous emergence of computation from adaptive synchronization of networked dynamical systems. The fundamentals are nonlinear elements, interacting in a directed graph via a coupling that adapts itself to the synchronization level between two input signals. These units can emulate different Boolean logics, and perform any computational task in a Turing sense, each specific operation being associated with a given network's motif. The resilience of the computation against noise is proven, and the general applicability is demonstrated with regard to periodic and chaotic oscillators, and excitable systems mimicking neural dynamics.

DOI: 10.1103/PhysRevE.84.060102 PACS number(s): 64.60.aq, 05.45.Xt, 89.20.Ff

Synchronization of networked units [1] is ubiquitously observed in natural systems [2]. Most of the studies on the subject have revealed the mechanisms for the self-organization of the interacting nodes in an either synchronous or asynchronous state [3]. However, neither state in isolation is sufficient to carry out a computational task, as a certain degree of heterogeneity is required on information processing networks. On the other hand, as far as bridging dynamics and computation are concerned, examples of information processing were provided with cellular automata [4], lattices of coupled chaotic maps [5], or by creating chemical and neuronal diodes with excitable fields [6] and patterned neuronal cultures [7]. Undoubtedly *chaos computing*, i.e., designing Boolean gates based on chaotic systems, has been the most successful approach so far. However, all related studies [8] rely on the common requirement of an external nonfeedback control, as (e.g.) the use of a threshold on a state variable, or the selection of specific values for the system's parameters, in order to confine the dynamics into a desired state.

In this Rapid Communication, we introduce a paradigm where, instead, computation emerges spontaneously from the balanced alternation of synchronization and desynchronization in an adaptive, directed, network; and we show how different network motifs [9] can be associated to specific computational tasks.

A basic requirement is to have a reliable representation for storing, processing, and retrieving information from memory. Here, we use the well-known Boolean logic, and we binary code the information on the level of synchronization of each network's unit with two signals, $S(t)$ and $R(t)$. The fundamental element of computation is sketched in Fig. 1, and consists of a dynamical system [in the following, the cases of a Kuramoto phase oscillator, a chaotic Rössler oscillator, and a Hodgkin-Huxley (H-H) neuron in an excitable regime will be separately considered], and two input ports (A and B). The assumption is that almost all networked units are subjected to *the same* external source (a synchronizing signal in the Kuramoto and Rössler cases, and a Gaussian white noise in the case of H-H), in a way that their dynamics (in the absence of any further interaction) would result in a time series synchronous with $S(t)$ (which, from here on, will be taken as the 0-state of the computation). We also assume that a second reference signal $R(t)$, constituting the 1-state, is present in the network, as produced by the evolution, for instance, of *at least one unit* that is not suffering the effect of the common forcing.

The computational capability is warranted by a second coupling mechanism, represented by an input signal entering port A, and associated with an adaptively regulated strength that tends to zero when the input V_A of port A is synchronized with that of port B (V_B), and to a positive value otherwise. Specifically, the general form of the equations describing the computational unit is

$$\dot{\theta}_i = f(\theta_i, \mathbf{p}) + C_{\mathrm{com}}(t) + W_i(t) h(\theta_i, V_A(t)), \qquad (1)$$

where θ_i is the vector state of the ith unit of the graph, f is a local evolution function, $\mathbf{p}$ is a set of control parameters, $C_{\mathrm{com}}(t)$ represents the common forcing, $W_i(t)$ is the strength of the coupling with the signal $V_A(t)$, and $h(\ldots,\ldots)$ is a coupling function, to be later specified. As for the evolution of $W_i(t)$, we consider the following adaptive dynamics:

$$\dot{W}_i = -W_i(W_i - w_1)(W_i - w_2) + k(\chi(V_{A_i}, V_{B_i}) - \tau), \qquad (2)$$

with w_1 and w_2 being suitable constants. Notice that, when $w_1 = 0.5$ and $w_2 = 1.0$, the first right-hand-side (rhs) term of Eq. (2) creates three equilibrium points, two of them being stable ($W = 0$ and $W = 1$). The second rhs term is instead responsible for driving W_i to one of the two above values. Moreover, $\chi(V_A, V_B)$ is a function that quantifies the synchronization error between the signals entering ports A and B, $k > 0$ is a parameter defining the time scale of the adaptation, and τ is a threshold needed to filter out small synchronization errors that may be due to noise. All parameters and functions are here set in a way that, as $\chi(V_A, V_B) > \tau$, W_i is forced to the value $W_i = 1$, so that the unit synchronizes with $V_A(t)$. When, instead, $\chi(V_A, V_B) < \tau$, W_i vanishes (thus decoupling the unit from the input of port A), and the system follows the guidance of $C_{\mathrm{com}}(t)$, generating the pattern $S(t)$ (see the truth table in Fig. 1).

We now show how the applicability of the proposed paradigm is general. To this end, we start by implementing the simplest logical gate, e.g., the unary NOT gate, whose output

*massimiliano.zanin@ctb.upm.es

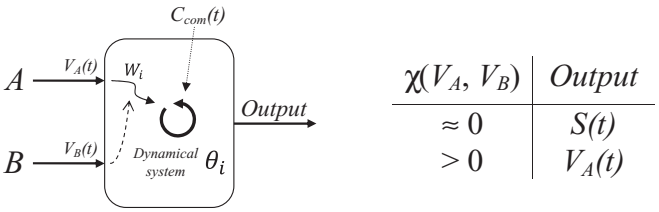


FIG. 1. Schematic representation of the basic computational unit. Left: The core is a dynamical system (depicted as a loop) whose state θ_i is forced by two signals, $C_{\rm com}(t)$, and $V_A(t)$. The latter is multiplied by a coupling strength W_i that adaptively depends on the synchronization error $\chi(V_A, V_B)$ between the signals entering ports A and B. Right: Output of the unit as a function of $\chi(V_A, V_B)$.

is 0 (1) when its single input is 1 (0). Figure 2(a) shows the needed configuration to recover the gate: The input signal $I(t)$ is plugged as $V_B(t)$, while $V_A(t) = R(t)$. Following Eqs. (1) and (2), when $I(t)$ is synchronized with $R(t)$ (i.e., the entrance is a 1 bit), $\chi(V_A, V_B)$ is close to zero, and the output $O(t)$ of the unit will produce the dynamics $S(t)$ (the 0 bit); on the contrary, as the 0 bit is presented to port B, $\chi(V_A, V_B)$ will be positive, and $O(t)$ will synchronize with $R(t)$, thus returning a 1 output. Such an emergent computation can be verified in the three following cases:

(i) When the dynamical systems are Kuramoto phase oscillators [10] [Fig. 2(b)], i.e., when θ is a phase variable evolving as

$$\dot{\theta}_i = \omega + 0.1\sin(\theta_p - \theta_i) + W_i(t)\sin[\theta(V_A(t)) - \theta_i],$$

where $\theta_p = \omega_p t, \theta(V_A(t))$ is the phase of V_A, and $W_i(t)$ evolves according to Eq. (2) [with $\chi(\theta_A, \theta_B) = (\frac{\theta_A - \theta_B}{2\pi})\text{mod}_1$]. In this case $S(t)$ [$R(t)$] is $\sin(\theta_p)$ [$\sin(\omega_R t)$], with $\omega = 1150$ Hz, $\omega_p = 500$ Hz, and $\omega_R = 1600$ Hz.

(ii) When the systems are chaotic Rössler oscillators [11] [Fig. 2(c)], i.e., for $\theta_i = (\theta_i^X, \theta_i^Y, \theta_i^Z)$,

$$\dot{\theta}_i^X = -\theta_i^Y - \theta_i^Z + 0.25\big(S(t) - \theta_i^X\big) + W_i(t)\big(V_A(t) - \theta_i^X\big),$$
$$\dot{\theta}_i^Y = \theta_i^X + 0.165\theta_i^Y,$$
$$\dot{\theta}_i^Z = 0.2 + \theta_i^Z\left(\theta_i^X - 10\right).$$

W_i evolves following Eq. (2). Here, $\chi(V_A, V_B) = 1$ for $r < 0$, and $\chi(V_A, V_B) = 1 - r$ for $r \geqslant 0$ (with r being the Pearson's correlation coefficient [12]), and $R(t)$ is the x component of a network's unit (uncoupled with the rest of the graph) for which $C_{\rm com}(t) = 0$.

(iii) When the units are under the form of an excitable Hodgkin-Huxley neuron [13] [Fig. 2(d)], defined by

$$C_m \frac{dV}{dt} = I_l - I_{Na} - I_K - C_{\rm com}(t) - W(V - V_A),$$

where θ is identified with the membrane voltage V, C_m is the membrane capacitance, $I_l = g_l(V - V_l)$ is a passive leak current, and I_{Na} and I_K represent simplified depolarizing and repolarizing currents, respectively (see Ref. [13] for all other relevant definitions). Here, a well-known phenomenon is that of *noise-induced synchronization*, which drives the unit's output dynamics into a synchronous spike pattern in the presence of a same noise source [14]. Therefore, the 0-state dynamics $S(t)$ is here created by a Gaussian noise term $C_{\rm com}(t) = \xi_n$, and $\chi(V_A, V_B)$ is defined as the proportion of spikes in the signals from both ports (A or B) that do not coincide with a spike in the signal of the other.

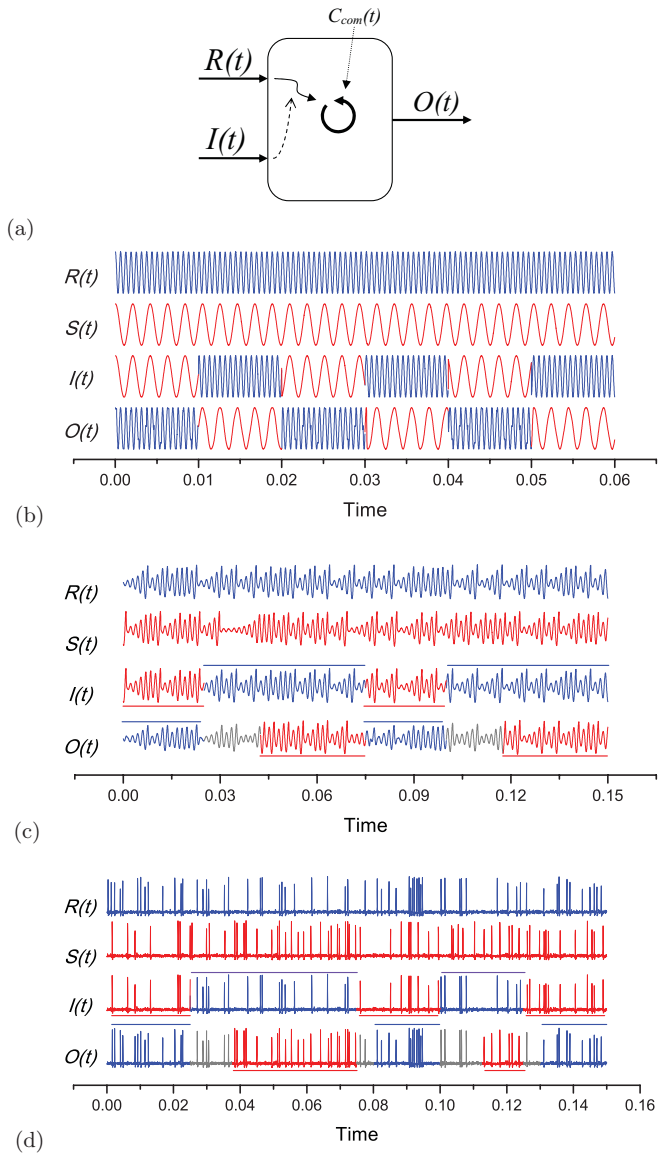
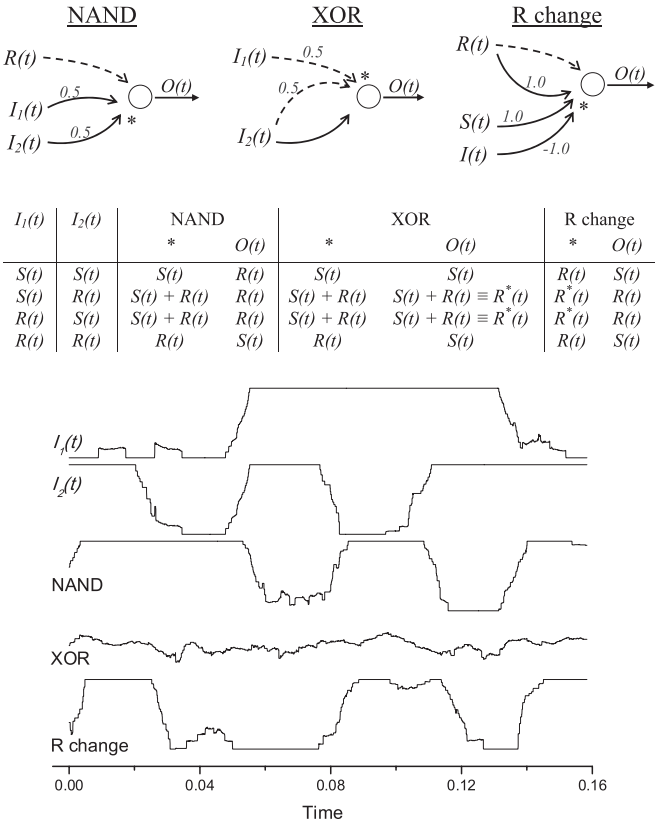


FIG. 2. (Color online) Diagram (a) and numerical simulations of the unary NOT gate, obtained by feeding the input signal inside port B, and $R(t)$ into port A. (b), (c), and (d) correspond, respectively, to the cases of Kuramoto oscillators, Rössler chaotic oscillator, and Hodgkin-Huxley neuron. See text for all stipulations and parameters used in each case. In all cases, the signal $R(t)$ (in blue/dark gray), the signal $S(t)$ (in red/gray), the input signal, and the processed output are reported. In the bottom panels, lines mark the changes in the input and output from the bit 0 to 1 and vice versa (for the sake of a better visualization of the grayscale version of the figure), and the light gray code of $O(t)$ is used for the transient periods needed to efficiently respond to such input changes.

Figure 2 shows that the NOT operation is correctly performed in all three cases, and the unit's output $O(t)$ converges to the prescribed dynamics, after a small transient (colored in light gray) is elapsed, whose average duration depends on the specific case.



$I_1(t)$	$I_2(t)$	NAND *	NAND $O(t)$	XOR *	XOR $O(t)$	R change *	R change $O(t)$
$S(t)$	$S(t)$	$S(t)$	$R(t)$	$S(t)$	$S(t)$	$R(t)$	$S(t)$
$S(t)$	$R(t)$	$S(t)+R(t)$	$R(t)$	$S(t)+R(t)$	$S(t)+R(t)\equiv R^*(t)$	$R^*(t)$	$R(t)$
$R(t)$	$S(t)$	$S(t)+R(t)$	$R(t)$	$S(t)+R(t)$	$S(t)+R(t)\equiv R^*(t)$	$R^*(t)$	$R(t)$
$R(t)$	$R(t)$	$R(t)$	$S(t)$	$R(t)$	$S(t)$	$R(t)$	$S(t)$

FIG. 3. Top: Network motifs associated to the NAND, XOR, and reference change gates. Continue (dashed) lines indicates the signals entering port B (A). Center: The associated *truth table* for the three computational tasks. Bottom: Numerical representations of the two inputs and of the output of the H-H unit, reported in terms of the corresponding synchronization levels with the reference signal $R(t)$.

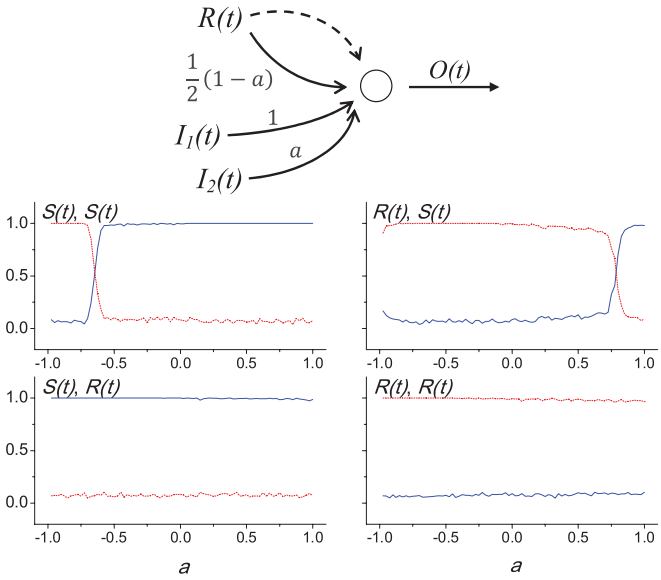


FIG. 4. (Color online) Top: Motif associated to the morphing gate, where the weights of the links depend on the parameter a, defined in $[-1,1]$. Bottom: Outputs of numerical experiments with the H-H system [represented in terms of their synchronization levels with $S(t)$ (red/gray lines) and $R(t)$ (blue/dark gray lines)] vs a, for the four possible combinations of the two inputs (reported on top of the corresponding panels).

By further embedding the computational unit into a directed network, more complex logical gates can be constructed. The output of a unit can feed the input(s) of one (or multiple) other unit(s), and each unit may receive information from several other sources within a motif architecture. While the same generalities apply for all logical gates that will be described, from here on we will focus on the H-H case, as the results there would be of relevance in neuroscience, because of the parallelism with the experimentally observed transient pairwise synchronized neuron's dynamics [15].

By summing two input signals, and feeding the result into port B [while port A still receives the reference signal $R(t)$], a NAND gate is implemented (Fig. 3, top), which returns 0 only when its two inputs have a value of 1. NAND gates are of particular importance, in that they are *universal Boolean gates*, i.e., any other Boolean logic can be constructed by properly combining different NAND gates, and a *universal Turing machine* can be constructed [16]. In particular, by assembling four NAND gates, it is possible to build a XOR gate, which returns 1 only when its two inputs have opposite values. Notice that the proposed paradigm offers an even more efficient solution, in which the same gate is obtained with only a computational unit, by feeding the two inputs inside port A, and the second input inside port B. The signal $O(t)$ outcoming from this motif, yet, encodes the 1 bit with a signal that is the sum of $R(t)$ and $S(t)$ (Fig. 3, center and bottom). Though such an output can be used as a different reference $R^*(t)$ by other computation units, it can also be converted back to $R(t)$ by plugging it in a *reference change* gate (Fig. 3, top right-hand side). These last two examples prove the flexibility of the proposed paradigm, as it is possible to design far more efficient circuits by means of specific motifs of the network. Moreover, the computation is not limited to the two standard Boolean states (0 and 1), as more than one reference signal can be used to encode the bit 1, thus expanding the graph's computational options.

Furthermore, when different operations are to be executed, the approach is to create circuits that can switch between different single-purpose gates. One of the most promising features of computation outside the digital realm is, indeed, the possibility of creating *morphing* gates that can change the operation executed by means of tuning a parameter, this way reducing the number of needed computing elements. The solution here involves the option of tuning the weights at the entrance of the ports A and B by a parameter a. Figure 4 shows the motif for a such element that is able to execute three different tasks. When a is set to 1, the circuit acts as a NAND gate (see Fig. 3). When $a = 0$, the second input $I_2(t)$ is disregarded, and the signal $O(t)$ is the output of a unary NOT gate. Finally, when $a = -1$, the resulting operation is known as the "I_1 is implied by I_2" Boolean algebraic function, which returns 0 only if $I_1 = 0$ and $I_2 = 1$. The outcomes of these three operations are reported in the bottomg of Fig. 4, where each plot represents the synchronization level of $O(t)$ with $R(t)$ and $S(t)$, for the four possible inputs combinations ([0,0], [1,0], [0,1], and [1,1]).

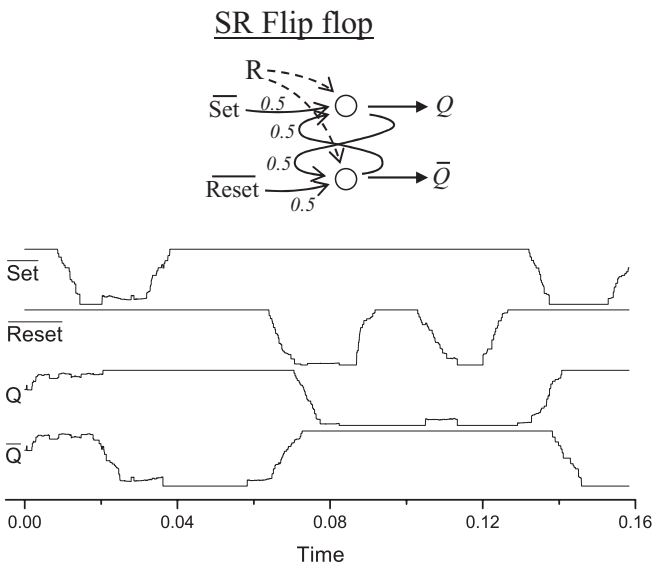


FIG. 5. Top: Motif associated to the set reset flip-flop memory circuit. Bottom: Numerical simulations of the flip-flop circuit with the H-H model. The signals are represented with the same stipulations as in the caption of Fig. 3.

So far we have focused on *static* Boolean gates, where outputs only depend on the inputs at the same time. Figure 5 shows instead a *dynamic* gate, the flip-flop gate, whose output Q changes to 0 (1) when the *Reset* (*Set*) input is set to 0, and is maintained until a different input is activated. This circuit is therefore a memory that stores a bit of information until another input is presented to its ports. The dynamics of Q is shown in the bottom of Fig. 5, together with that of $\bar{Q}$, the other output of the circuit that is always the opposite of Q (except for an initial period in which the circuit is not fed by inputs). The associated motif (Fig. 5, top) includes two computational units and a double feedback: The output of each unit is sent to the port B of the other unit. Notably, such feedback loops represent a striking challenge for any circuit, as small errors in the output of one unit may disturb the computation of the second unit, whose output will contain higher levels of noise, and so forth; therefore, a small noise might be amplified in the loop, and eventually lead to meaningless computations. For the loop to be stable, each computational unit should be able to perform the requested computation even in the presence of noise, and therefore provide a noise-free output signal. In order to assess the robustness of the single unit's computation, we have considered a Rössler NOT gate (the one shown in the bottom left-hand side of Fig. 2) subjected to different levels of Gaussian noise added to both signals V_A and V_B, and the percentage of wrong outputs has been monitored. For noise amplitudes of 20%, 15%, and 10% of that of θ_i^X, the measured success rates are, respectively, of 75%, 93.6%, and more than 99.5%, calling for a highly resilient computation even for large external perturbations.

In conclusion, we introduced a computational paradigm, where the coding and processing of information emerge from adaptive synchronization processes. We demonstrated the efficacy, scalability, robustness, and resilience of such a paradigm in performing single-input (NOT) gates, multiple-input (NAND, XOR, and morphing) gates, as well as dynamical (flip-flop memory) gates, with three totally different nonlinear systems: a phase oscillator, a chaotic oscillator, and a spiking neuron. Our results are of relevance in enlightening possible biological mechanisms at the basis of the processing and integration of information across distributed neural systems, where neural assemblies are known to organize their dynamics in a balance between synchronization and desynchronization [15], with modifications associated with a number of neurological illnesses, including schizophrenia and Alzheimer's disease [17].

[1] R. Albert and A. L. Barabási, Rev. Mod. Phys. **74**, 47 (2002); S. Boccaletti *et al.*, Phys. Rep. **424**, 175 (2006).
[2] S. Boccaletti *et al.*, Phys. Rep. **366**, 1 (2002); A. Arenas *et al.*, *ibid.* **469**, 93 (2008).
[3] E. Rodriguez *et al.*, Nature (London) **397**, 430 (1999); P. Fries *et al.*, J. Neurosci. **22**, 3739 (2002).
[4] T. Toffoli and N. Margolus, *Cellular Automata Machines: A New Environment for Modeling* (MIT Press, Cambridge, MA, 1987).
[5] S. Sinha and W. L. Ditto, Phys. Rev. Lett. **81**, 2156 (1998).
[6] O. Steinbock, P. Kettunen, and K. Showalter, J. Phys. Chem. **100**, 18970 (1996); I. Motoike and K. Yoshikawa, Phys. Rev. E **59**, 5354 (1999).
[7] O. Feinerman, A. Rotem, and E. Moses, Nat. Phys. **4**, 967 (2008).
[8] K. Murali and S. Sinha, Phys. Rev. E **75**, 025201 (2007); K. Murali *et al.*, Phys. Lett. A **373**, 1346 (2009); W. L. Ditto *et al.*, Chaos **20**, 037107 (2010).
[9] R. Milo *et al.*, Science **298**, 824 (2002).
[10] Y. Kuramoto, *Chemical Oscillations, Waves and Turbulence* (Springer, New York, 1984).
[11] O. E. Rössler, Phys. Lett. A **57**, 397 (1976).
[12] H. E. Soper *et al.*, Biometrika **11**, 328 (1917).
[13] A. L. Hodgkin and A. F. Huxley, J. Physiol. **117**, 500 (1952).
[14] A. S. Pikovsky, Radiophys. Quantum Electron. **27** 390 (1984); F. Moss, L. M. Ward, and W. G. Sannita, Clin. Neurophysiol. **115**, 267 (2004).
[15] K. J. Friston, Philos. Trans. R. Soc. London B **355**, 237 (2000); M. Breakspear, L. M. Williams, and C. J. Stam, J. Comput. Neurosci. **16**, 49 (2004); J. Benda, A. Longtin, and L. Maler, Neuron **52**, 347 (2006).
[16] T. C. Bartee, *Computer Architecture and Logic Design* (McGraw-Hill, New York, 1991).
[17] C. J. Stam *et al.*, Neuroimage **32**, 1335 (2006); R. Y. Cho, R. O. Konecky, and C. S. Carter, Proc. Natl. Acad. Sci. USA **103**, 19878 (2006); P. J. Uhlhaas and W. Singer, Nat. Rev. Neurosci. **11**, 100 (2010).