Journal of Physics: Complexity

OPEN ACCESS

RECEIVED
17 August 2021

REVISED
1 November 2021

ACCEPTED FOR PUBLICATION
17 November 2021

PUBLISHED
14 December 2021

PAPER

Bio-inspired computing by nonlinear network dynamics—a brief introduction

Fabio S Neves[1,*] and **Marc Timme**[1,2,3]

[1] Chair for Network Dynamics, Center for Advancing Electronics Dresden (cfaed) and Institute for Theoretical Physics, TU Dresden, 01062 Dresden, Germany
[2] Cluster of Excellence Physics of Life, TU Dresden, 01062 Dresden, Germany
[3] Lakeside Labs, 9020 Klagenfurt am Wörthersee, Austria
[*] Author to whom any correspondence should be addressed.

E-mail: fabio.neves@tu-dresden.de and marc.timme@tu-dresden.de

Keywords: bio-inspired computation, spiking neural networks, dynamical systems, nonlinear dynamics, *k*-winners-take-all

Abstract

The field of bio-inspired computing has established a new Frontier for conceptualizing information processing, aggregating knowledge from disciplines as different as neuroscience, physics, computer science and dynamical systems theory. The study of the animal brain has shown that no single neuron or neural circuit motif is responsible for intelligence or other higher-order capabilities. Instead, complex functions are created through a broad variety of circuits, each exhibiting an equally varied repertoire of emergent dynamics. How collective dynamics may contribute to computations still is not fully understood to date, even on the most elementary level. Here we provide a concise introduction to bio-inspired computing via nonlinear dynamical systems. We first provide a coarse overview of how the study of biological systems has catalyzed the development of artificial systems in several broad directions. Second, we discuss how understanding the collective dynamics of spiking neural circuits and model classes thereof, may contribute to and inspire new forms of 'bio-inspired' computational paradigms. Finally, as a specific set of examples, we analyze in more detail bio-inspired approaches to computing discrete decisions based on multi-dimensional analogue input signals, via *k*-winners-take-all functions. This article may thus serve as a brief introduction to the qualitative variety and richness of dynamical bio-inspired computing models, starting broadly and focusing on a general example of computation from current research. We believe that understanding basic aspects of the variety of bio-inspired approaches to computation on the coarse level of first principles (instead of details about specific simulation models) and how they relate to each other, may provide an important step toward catalyzing novel approaches to autonomous and computing machines in general.

1. Introduction

Biological information processing systems are multi-scale, ranging from systems of a few molecules and single cells to large neural networks (NNs). Complex NNs, such as in the animal brain, can perform a broad range of information processing tasks [1]. By integrating multi-dimensional sensory signals with memory, they generate complex self-organized behaviors, often also integrating past experiences in seemingly unrelated fields. Even though these natural dynamical systems operate in continuous-time, often in continuous-state (analog) and in intrinsically noisy environments, they are capable of performing robust parallel computations. Understanding neuronal networks and basic mechanisms underlying their dynamics thus provide a formidable opportunity to unveil new architectures for artificial computing machines.

Over the past decades, an increasing number of studies on artificial systems attempting to remodel or outperform related biological systems has been developed across subjects, from mathematics and physics [1–4] to computer science and engineering [5–7]. Some already drive the latest information revolution via a paradigm of machine learning (ML) [8]. ML approaches provide ways to explore large data sets for classification or regression, and, more interestingly, to find solutions to computational tasks in an unsupervised manner (without human intervention) by exploiting various correlations of provided data, even if they are not apparent to a trained human. Another example is dedicated artificial intelligence, where machines extract probabilities of future events based on innumerable examples, culminating on computers beating the best humans in games with incredible numbers of configurations as chess [9] and, later, the GO games [10].

Even though powerful, broadly used ML approaches only exploit a few features of biological systems, such as parallel and distributed computation, mostly on feed-forward architectures. In contrast, biological NNs exhibit a wide variety of neurons, neural and communication features and network architectures [11–13], exploiting an equally large variety of emergent dynamics to compute. Their study has already unveiled a variety of new bio-inspired approaches to computation, such as long-term memory models [14], computation via precise spike timings [15] or complex trajectories in state space [16–20], signal processing via nonlinear recurrent interactions [21] and neural plasticity driven approaches to robotics [22] and computer vision [23]. Even though these are not broadly developed or adopted, core aspects of them may in time prove essential, as for ML after the advent of large unlabeled data sets. Some of the most intriguing approaches exploit dynamical systems to compute and take advantage of dynamics of pulse-coupled systems that communicate only at discrete instances in time, to efficiently exploit the time domain.

In this article, we focus on the value of studying the dynamics of biological neural systems and basic models of them to characterize fundamental computing features with potential applications in artificial systems. We first provide a short overview on novel and prominent bio-inspired approaches to computation such as layered NNs [8], pattern generators [24] and reservoir computing (RC) [21] to emphasize their complementarity and show that basic theoretical work on fields like biology and cognition may yield applications and implementations with high impact; second, we discuss the role of spiking neurons [25, 26] and their potential for unveiling and implementing novel computational paradigms; finally, we present in more detail the main dynamical systems' approaches to discrete decision making for the specific instance of winner-takes-all (WTA) computations [27]. In particular, we consider in depth two collective dynamics underlying computing, one based on heteroclinic dynamics [20, 28] and one on state-dependent inhibitory coupling [29], opening up a pathway toward arbitrarily complex information processing via WTA computation.

2. Bio-inspired perspectives on computation

Bio-inspired computation is a much older field than the latest wave of ML may suggest. Differently from the problem-solving direction of recent research, its first steps were more concerned with understanding the underlying mechanisms of neural systems [30, 31], such as the fundamentals of neuronal activity [32] and memory formation [33], and attempted, with different levels of success, to provide an understanding of cognition involving language and memory [34]. The field now known as connectionism [35] provided an alternative to more traditional symbolic approaches to model and study cognition [31], offering a functional approach instead. The basic idea is: if we can create artificial NNs with a certain level of detail, modeling a given neural circuit, we could in principle observe the emergence of discrete patterns of activity which match the behavior or measured neuronal activity in biological systems, thus providing insights on their underlying mechanisms. A seminal result related to computation has been achieved early from analyzing simple threshold neurons and networks of such units, now known as perceptron networks [36]. Such units exhibit binary outputs defined via a simple threshold. If the input signal is strong enough, the output is one; if not, it is zero. Due to their mathematical tractability, perceptrons and systems of threshold units received broad interdisciplinary interest, from biology and cognitive science to exact sciences.

The original perceptron consists of a two-layer network, one input layer and one output layer. Information is encoded in the strength (or weights) of the connections between both layers. The generalization of the perceptron architecture, known as multi-layered perceptrons includes hidden layers between the input and output layers and allow, in principle, for the encoding of any function [30]. These layered networks are the precursors of modern ML approaches, as they differ from modern systems mostly regarding the larger number of layers and the learning rules employed rather than the fundamental architecture: simple nodes distributed in layers with multiple interconnections in a forward direction, see figure 1(b). Later, the introduction of the back-propagation algorithms [37] provided the mathematical foundations for teaching desired associations in deeper networks (many layers), what was not possible, in general, before. Among the many variations and improvements on this combination of architecture an learning rules, a foundational example is the back-propagation through time algorithm [38], which provides a way to codify time varying information in certain recurrent networks, with a similar approach as to layered networks.

In complementary architectures with recurrent connectivity, dynamical systems theory combined with insights from neuroscience has been used for classification or function approximation. For example, two equivalent dynamical systems approaches were proposed independently by Maass and Jaeger, termed liquid

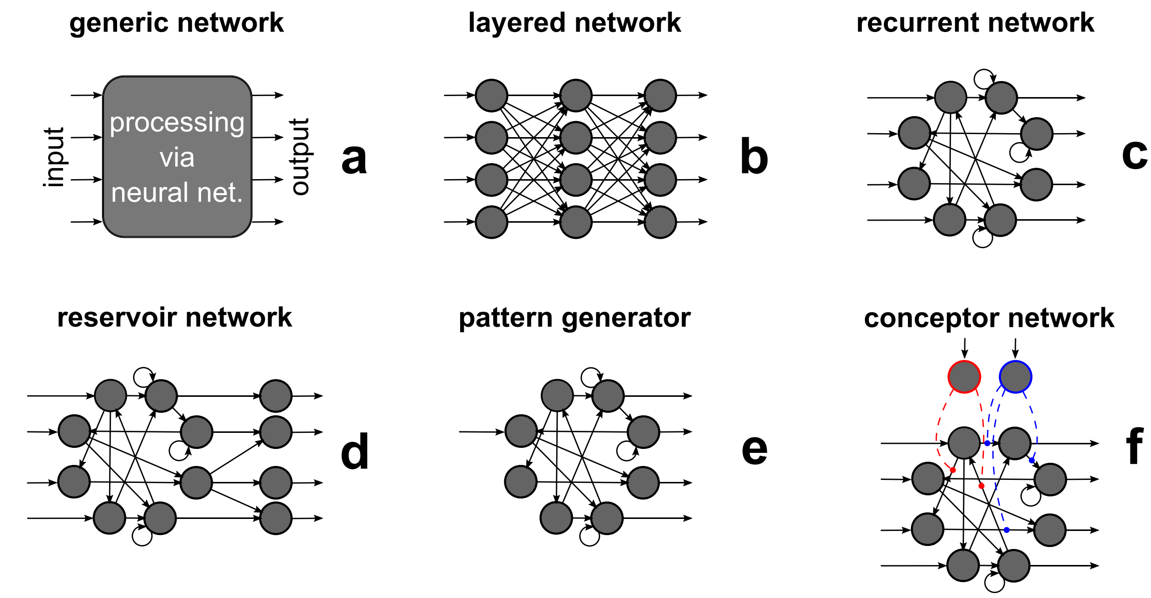

Figure 1. Examples of network architectures underlying various computing paradigms. (a) Schematic generic architecture underlying a computation. Inputs are processed by a network that generates or updates an output state. (b) In layered networks, forming the basis of standard perceptrons as well as deep learning paradigms. There are only forward connections from any layer to the next. (c) In a recurrent network, signals are sent also backwards via feedback connections, such that various collective dynamics may emerge. (d) RC (random) networks are coupled to an output layer, the readout weights, yet not the internal weights, are adapted during a training phase. (e) A pattern generator transforms a low-dimensional input to a higher-dimensional one (possibly also exploiting the time domain). (f) In a conceptor network not only a standard input is used to activate a pattern, but inputs from (lateral) conceptor nodes that adapt recurrent connection strengths (dashed lines).

state computing [39] and echo state machines [40, 41], respectively. Both paradigms combined the intrinsic memory of recurrent networks with the well established learning algorithms for layered networks. Today, the unified theory is called RC [21]. In this computing paradigm, a recurrent network (figure 1(c)) with random connectivity is inserted between the inputs and a (typically) shallow output network (few layers), see figure 1(d). The reasoning is that the recurrent network provides memory to the system and due to the many, typically nonlinear internal interactions, the system calculates random features of the original signal, therefore, enriching it (similarly to kernel machines). The classification or regression process is then performed over this enriched signal with learning occurring only at the shallow, often single-layer, output network. RC, thus, not only provides a natural way of applying standard ML algorithms to time-varying inputs, but also requires learning only relatively few parameters of the shallow network, thus reducing learning resources, in particular learning times. Overall today, RC is a maturing field that has already been implemented in customized hardware, including electronics [42] and the combinations of electronics and optical system [42–44], potentially on pair with state-of-the-art digital realizations, e.g. in speech recognition [45].

Early research on recurrent networks also laid the foundations to long term memory models [14]. Such early research demonstrated on the network level (nodes and connections) how reinforcement learning [33], as envisioned in animal models, can create memories represented as (stable) attractors of a dynamical system. The seminal works of Hopfield [14] and Hebb [33] provided the fundamental ideas on how a neural system can learn via examples by reinforcing connections between neurons that are often concurrently active. These works had a large impact on a variety of field, from biology to applied computer science, with many spinoffs and variations on the topic. Attractor networks have a downside, though. As the relevant states are stable, switching between two states becomes non-trivial from a dynamical systems perspective. In short, one must either strongly move the state of the system in state space or directly reset it, both radical interventions likely far from a biological systems approach.

Conceptors [46] constitute a recently developed dynamical approach to control the activity of NNs originally exhibiting stable attractors. Consider a NN encoding a given stable attractor. To encode the attractor, a learning algorithm must have made some connections in the network stronger and some weaker. A conceptor neuron associated to this attractor is simply an external neuron that, when active, would amplify such effect, making the same connections either stronger or weaker, thus highlighting the specific attractor, see figure 1(f). If a set of attractors is encoded in the same network with an associated set of conceptors, it may be possible to seemingly turn on and off any given attractor, as desired and controlled by the conceptor neuron(s). Moreover, by controlling the intensity with which a set of conceptor neurons influences the dynamics concurrently, it is possible to ‘interpolate’ attractors, to generate smooth transitions between network states, as one conceptor is slowly turned off while another is slowly activated. Conceptors, thus, serve as a bio-inspired control systems for dynamical NNs with potential applications, for instance in controlling locomotion modes for robotics, or more generally switching between states in abstract NNs.

Another interesting subject in bio-inspired computing is periodic activity created by pattern generator systems, with applications to locomotion systems. In biology, central pattern generators play a fundamental role executing complex high-dimensional activity patterns in response to more abstract lower dimensional signals from the motor cortex [47]. They encode high-level decisions, for example 'move forward'. The moving order may be represented in neural activity as simple as the activation of a single neuron, while the movement itself may require a complex pattern that centrally, i.e. from one network, coordinates the movement of multiple muscles in multiple limbs. Works on dynamical artificial pattern generators have shown how small recurrent NN (figure 1(e)), exhibiting either emergent synchronization patterns [24, 48] or controllable chaos [49], can be exploited to generate complex motions adequate under a variety of conditions, e.g. slow or fast gates, walking or swimming patterns or even self-untrapping. Artificial central pattern generators, thus, may become, in time, essential contributors in robotic applications as autonomous agents, because demand for automation and the complexity of automated tasks are ever increasing.

The small sample of computing paradigms above already shows how bio-inspired paradigms can be remarkably complementary. While ML networks are trained to identify features in their raw input signals, resembling (in broad strokes) the cortex, recurrent networks model memory storage and recall via stable attractors and pattern generators can provide a rich variety of locomotion modes with simple dedicated NNs.

In the next sections, we will first discuss how spiking neurons may help to unveil new computing paradigms by exploiting the time domain and later, in more depth, the specific topic of discrete decision-making modeled as a WTA computation performed by spiking NNs.

3. Spiking neurons and spiking NNs

A prominent mark of bio-inspired computing is the use of NNs, with a strong emphasis on the coupling network. The neuronal models broadly used are simple functions, similar to the original McCulloch and Pitts model [32]. The resemblance to a neuron is relegated to the summation of inputs from (potentially) many nodes, in contrast to its single-variable output state. Spiking neurons [25], even though less prominent, may have a much larger potential for computation, because they can encode information concurrently in space and on the time domain, yielding a variety of possible encoding schemes, including precise spike-timings [50–52] and synchronization [20, 28, 53, 54].

The biological neuron is the fundamental building block in the animal nervous system. Its function has been shown to be wildly varied, from simple information transmission, e.g. for muscle contraction, to the direct computation of chemical concentration by receptor neurons. More complex computing processes in the brain arise from emergent dynamics in networks of neurons rather than from single neurons alone. NNs are complex systems in the sense that their global dynamical and thus computational characteristics cannot be explained exclusively from the dynamics of their composing neurons. Nevertheless, different individual neuronal properties play an essential role in the resulting network dynamics.

Neurons are varied in size, their functionality and even their chemistry. Broadly speaking, neurons are subdivided into two main parts, the dendrites and body acting as the input sites and the axon transmitting the output, see figure 1(a). The former is tasked, from a functional perspective, with receiving inputs from other neurons and for initiating short-lasting electrical pulses called action potentials or spikes (figure 1(c)), the axon delivers such pulses to other neurons. The physical connection between neurons called synapses, where axon terminals contact dendrite or body of other neurons, come in two main variations: electrical and (electro-) chemical. Electric connections act like a resistor and enable electrical currents to flow between two neurons continuously in time, while electro-chemical ones, for example in the human cortex, transmit electricity via discrete discharges (spikes) of neural transmitter that change the conformation of proteins in the cell wall and enable (or disable) the throughflux of ions triggered by action potentials. In these synapses, transmitter vesicles in the presynaptic terminal face neural transmitter receptors in the post-synaptic neuron, maximizing their effectiveness. The flow of ionic charges either further polarize the cell membrane potential (a characteristic of an inhibitory connection) or depolarize it and the connection is called excitatory.

Mathematical neural models at different levels of detail play a fundamental role not only in applied fields but also for our understating of the basic functioning of biological systems. Even though highly detailed neural models, e.g. compartment models, may provide insights on the functionality of small networks or of the neuron itself, their effectiveness on understanding the broader functionality of large or intricately connected networks is limited, mainly due to the large number of parameters and variables to consider simultaneously. Simplified models, such as integrate-and-fire (IF) neuronal models drastically reduce the number of parameters

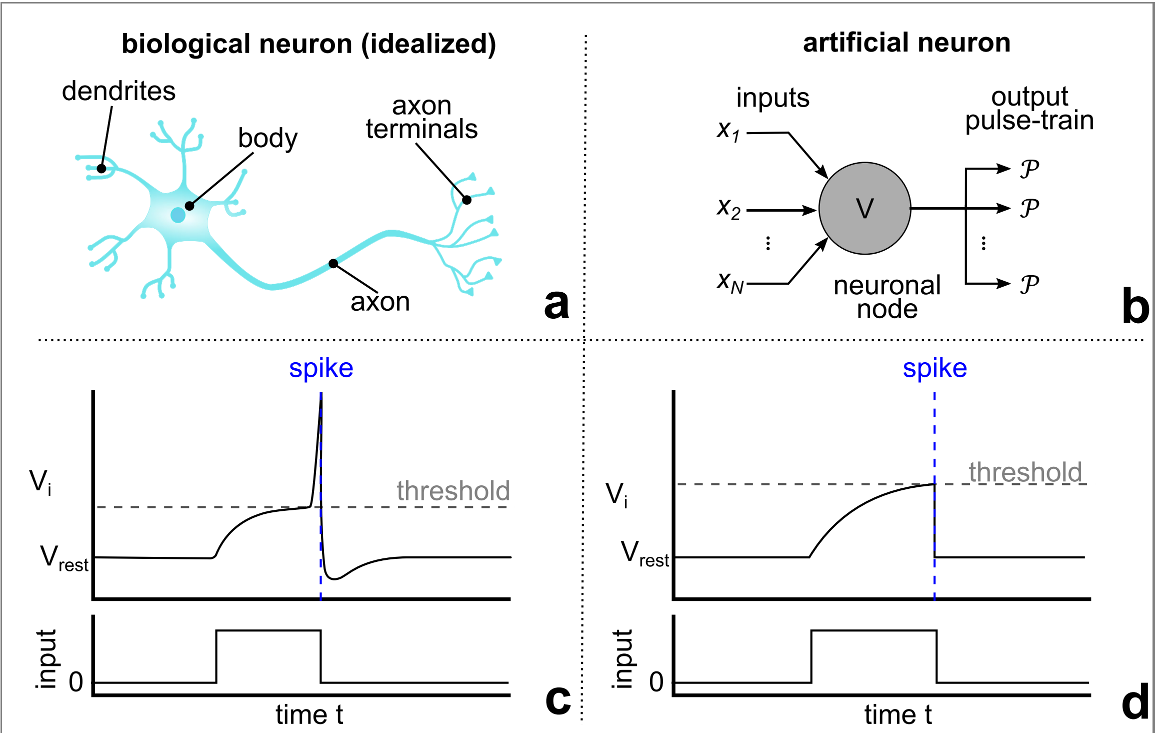

Figure 2. Idealized and artificial neuron and their dynamics. (a) Main subdivisions of an idealized neuron: dendrites, body, axon and terminals ('synapses'). (b) Schematic of an artificial neuron resembling a biological neuron. Multiple inputs and a single output targeting several other neurons. (c) Sketch of the dynamics of a biological neuron. Inputs drive the voltage from the resting potential V_{rest} until a threshold. At that point, an action potential is elicited, largely independent of future input. (d) Sketch of the dynamics of an artificial neuron, for which the action potential dynamics is replaced by a simple rule of spike emission and reset upon threshold crossing.

describing a network. A generic IF network is defined as

$$\frac{\mathrm{d}V_i(t)}{\mathrm{d}t} = f(V_i) + I_{\mathrm{ext}}(t) + I_{\mathrm{net}}(t), \qquad (1)$$

where V_i is a voltage-like variable, $f(\cdot)$ defines the internal dynamics of the neuron (typically a concave down function), $I_{\mathrm{ext}}(t)$ is an external driving current and $I_{\mathrm{net}}(t)$ is the pulse interaction at time t. Moreover, a threshold condition at V_{tr} is set, such that if $V_i(t)$ crosses the threshold from below, it is reset according to $V_i(t) := 0$, see figures 2(b)–(d). This reduced number of parameters may help to unveil fundamental features underlying a given collective phenomenon, for example synchronization, balanced dynamics or more complex emergent dynamics such as chaos and chimera states. The level of detail needed to study a given observed phenomenon, thus, is a non-trivial matter and gradually removing details is an established approach to define the sufficient conditions for the model. In turn, understanding computation on biological systems may lead to new paradigms and models for artificial computing. For example, IF neural models are often sufficient to understand many aspects of synchronization modes in artificial and biological systems. In the example above, one main simplification concerns the spike generation. Instead of modeling the fast rise in voltage follow by a fast drop (the action potential), the spike generation is substituted by a simple rule: if a threshold is reached, a pulse is sent and the voltage is reset to its base value. This simplification drastically reduces computing time in software and, due to the reduced model complexity, also simplifies potential hardware implementations.

As paradigms of artificial computation (the focus of this work), spiking NNs follow the same Occam's razor principle 'as simple as possible, but not simpler', used to understand emergent dynamics, in order to reduce architectural complexity while increasing functionality transparency. To date, a variety of computing paradigms have been unveiled which rely on the pulsatile (spiking) nature of the dynamics of coupling yet not on any precise form or model of spiking dynamics. From a single neuron perspective, spiking neurons may either encode information in their precise spike times [15, 50], such that the information is only bounded by the noise level and the precision of parameters, or in their average spike rate (rate models), expressing, for example, chemical concentrations. Moreover, network-level phenomena support a variety of computing paradigms involving emergent dynamics, including synchronicity modes, chaos and local spike rate variability, some already mentioned in the previous section.

Artificial spiking neurons, thus, may provide an interesting substrate for new bio-inspired computing paradigms, potentially combining simplicity with fundamental computing properties, as the signal integration

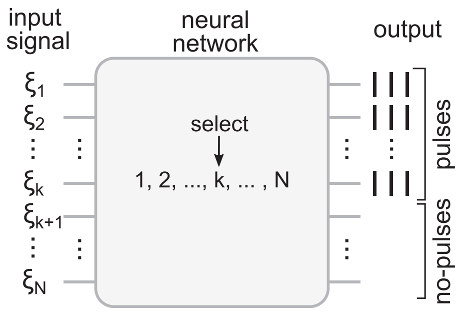

Figure 3. A schematic representation of a k-WTA computing system. Analogue input signals ξ_i are sorted into two groups, the k strongest, represented by spikes (discrete pulses), and another of $N - k$ weakest, represented by no spikes. k is selected or designed via global parameters of the underlying spiking NN.

and discrete interactions. In the subsequent parts of this article, we review in some depth two implementations of decision making via k-WTA emerging from two different collective dynamics emerging in spiking NNs.

4. Decision making via dynamical WTA computation

Biological systems are capable of performing high level discrete decisions, e.g. 'turn right', 'go fly', 'eat now', or 'rest', based on high-dimensional analogue input signals that are integrated in real time. This high-dimensionality is due, in great part, to the rich, multi-modal nature of the biological sensory systems. How can (artificial) NNs or in general, coupled dynamical systems encode such functionality?

Promising options are systems implementing a mathematical operation called WTA, because they not only exhibit discrete outputs given analogue inputs but also naturally emerge in a variety of dynamical systems. Furthermore, independently of the system type, the number of discrete outputs increases combinatorially with the number of units composing the underlying physical system, thus providing a large number of output options (decisions) in rather small systems. WTA or, more generally, k-WTA over N real valued inputs is an operation that calculates the partial rank order of inputs, sorting them into two sets, the k strongest, and $N - k$ weakest inputs, where $k \in \mathbb{N}$ is a system property (at given parameters), see figure 3. The k-WTA function is interesting, because it may serve as the basic building block for realizing or approximating any function [27, 55].

In this section, we discuss and compare two different dynamical systems paradigms to implement this function, namely heteroclinic computing [20] and dynamical inhibition [29], stressing their singular features where appropriate. In both approaches, we will consider networks composed of IF neurons, because, even though simple, they capture the two essential (and potentially advantageous) features of biological neurons, namely, interaction via exchange of spikes (or pulses) and a nonlinear summation over incoming spike trains. We remark that the essential mechanisms worked out below do not depend on these simplifications as we will see.

Specifically, in both paradigms, we analyze networks of oscillatory leaky-IF neurons with all-to-all connections (no self-coupling and thus loops). Each neural unit $i \in \{1, \ldots, N\}$ exhibits a voltage-like state variable x_i satisfying the differential equations

$$\frac{\mathrm{d}x_i}{\mathrm{d}t} = I - \gamma x_i + \xi_i(t) + \sum_{\substack{j=1\\ j\neq i}}^{N} \sum_{t_{j,\ell}\in P_j} \varepsilon(x_i)\delta(t - t_{j,\ell} - \tau), \tag{2}$$

for $x_i \in [0, x^\theta)$ where x^θ is a spiking threshold. The parameter I represents a constant current, γ is a dissipation parameter, and ξ_i an external signal serving as input, possibly including noise. Whenever a threshold is reached, $x_i(t^-) \geqslant x^\theta$, the state variable is reset to $x_i(t) = 0$ and a pulse (spike) is sent to all other neurons, mathematically reflected in the time $t = t_{j,\ell}$ of the ℓth threshold crossing, $\ell \in \mathbb{Z}$, by the neuron j. Without loss of generality we fixed $x^\theta = 1$. The sum in (2) is the contribution of all spikes arriving from the other $N - 1$ neurons j to i at time t, where P_j is the set of all times $t_{j,\ell}$ of spikes sent by neuron j. Moreover, $\tau \geqslant 0$ is the fixed coupling delay between neurons and $\varepsilon(x_i)$ quantifies the coupling strength as a function of the neuron i's state variable x_i.

4.1. Heteroclinic computing

Heteroclinic computing is a framework that exploits heteroclinic networks [56] or, more generally, networks of unstable states to compute [57, 58]. In such *networks in state space*, each node is a saddle periodic orbit,

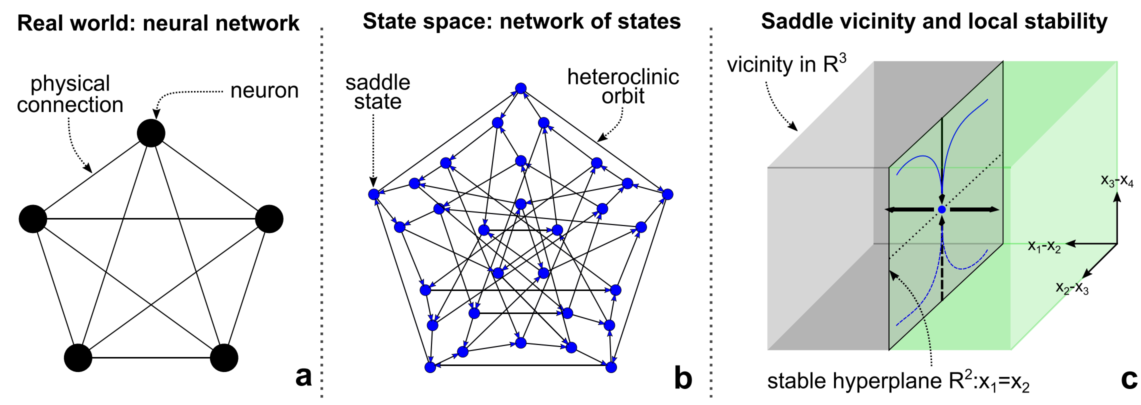
Real world: neural network
physical connection
neuron
a
State space: network of states
saddle state
heteroclinic orbit
b
Saddle vicinity and local stability
vicinity in R^3
x_3-x_4
x_1-x_2
x_2-x_3
stable hyperplane R^2:x_1=x_2
c

Figure 4. A network of neuron-like units and the resulting network of unstable states in state space. (a) A graph representing a NN in real space, composed of $N = 5$ identical neurons connected all-to-all ($I = 1.04$, $\tau = 1.59$, $\varepsilon = 0.025$). Nodes represent neurons, links represent physical connections. (b) A graph representation of the network of states emerging from the system in (a). Nodes represent saddle periodic orbits and directed links represent heteroclinic orbits. (c) A three-dimensional representation of the vicinity of a saddle orbit in state space, sampled at $x_5 = 0$, showing one saddle point (sampled form the orbit) surrounded by the basins of attraction of two other saddles, $x_1 > x_2$ and $x_1 < x_2$, where 1 and 2 stands for the indices of the neurons in the unstable cluster. The set of state space points where $x_1 = x_2$ defines the stable manifold.

i.e. a periodic orbit with both stable and unstable directions. Each connection between two such nodes is a heteroclinic orbit, that is, a directed connection between the stable manifold of a saddle with the unstable manifold of a second saddle, see figure 4. Networks of unstable states naturally emerge in symmetrical systems of identical phase- or pulse-coupled oscillators [59, 60], reducing the number of parameters of the model to the number of parameters defining one (oscillator) neuron and one single connection, independently of the system size $N > 2$. These properties, thus, facilitate remarkably the search and design of complex or large symmetrical networks of states exhibiting heteroclinic-like dynamics.

Computations are performed if external input signals $\xi_i(t)$ break the system symmetry and thereby break the heteroclinic connection. Sufficiently small signals typically induce long 'complex' periodic orbits that pass near a sequence of several 'simple' saddle periodic orbits, that we term *periodic sequences of unstable states*. It was shown in [20, 28] that such orbits can robustly encode k-WTA functions in phase-coupled as well as in pulse-coupled oscillator networks. In this section we first explain in detail how this encoding takes place and later show how such states can be decoded for further processing. We specify our model for heteroclinic dynamics by defining

$$\varepsilon(x_i) = \varepsilon, \tag{3}$$

in equation (2), i.e., as a simple positive constant.

4.1.1. Saddle periodic orbits

It has been shown that networks of neural oscillators with excitatory couplings ($\varepsilon > 0$) and connection delays ($\tau > 0$) evolving according to (1) exhibit a form of synchronization with fixed inter-spike times [61] and two or more groups of identically synchronized neurons. The mechanism behind such dynamics is twofold. First, spikes from a given neuron may trigger simultaneous threshold crossings and thereby resets in more than one (other) neuron, synchronizing their states exactly and simultaneously. Second, due to the delay, the neuron eliciting a spike will always reset a time τ before the neurons that are reset by the spike it sent, leading to the formation of subgroups of synchronized neurons or simply 'clusters'. If the frequencies of the neural oscillators composing the network are similar enough, this sequence of resets via supra-threshold incoming pulses may become cyclic, leading to a periodic orbit (in the absence of noise). A third option is that a pulse sent by a given neuron is received by a group of non-synchronous neurons at small x_i values, implying that these will not cross threshold, and thereby, further desynchronize causing an instability.

If all resets during one orbit's period are triggered by incoming pulses (supra-threshold events), the orbit is stable, because any small variation in the voltages of synchronized neurons is reduced to zero within one cycle (see figure 5(a)). The orbit is indeed superstable because the distance of the perturbed orbit to the unperturbed one does not decay exponentially but instantaneously. If one or more resets are not triggered by incoming spikes, small asynchronicities among neurons originally in one synchronized group will grow larger during each cycle due to the desynchronization mechanism described above. Incoming pulses will have different contribution to each neuron due to nonlinear (concave down) voltage curves [57], and the orbit (linearly) unstable (see figure 5(b)). If a periodic orbit exhibits stable and unstable manifolds, it is a saddle periodic orbit. During this work we call any 'short' saddle periodic orbit (one pulse per neuron until the sequence repeats) simply as

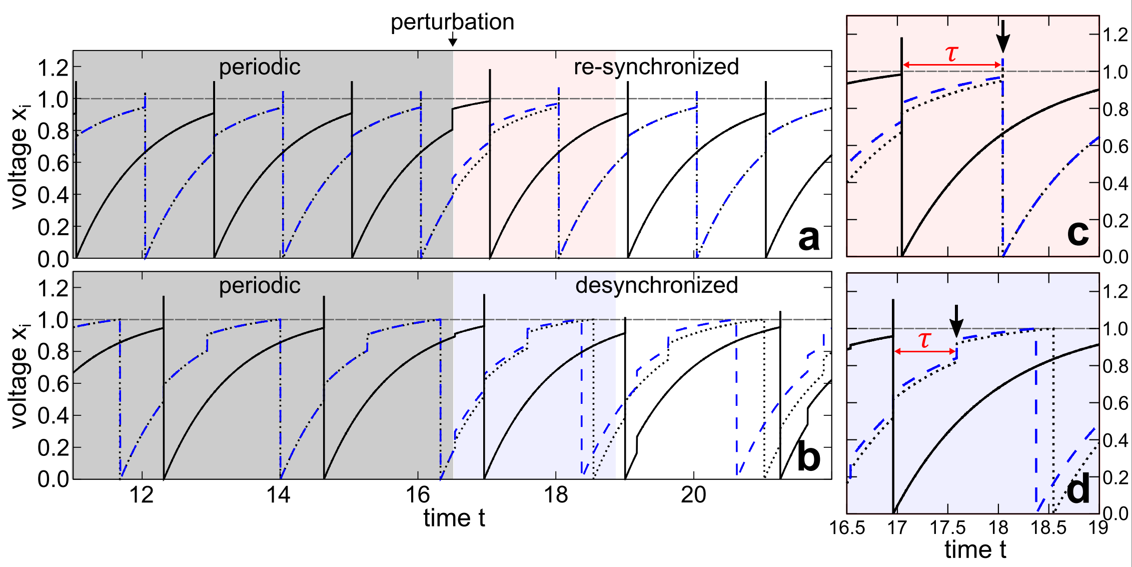

Figure 5. Dynamical response to an instantaneous perturbation. All panels: small system of $N = 3$ neurons, each line (black solid, blue dashed, black dotted) represents the voltage of one neuron. After the system converged to a periodic orbit (gray background), the system is perturbed by adding a small random vector. (a) A stable periodic orbit in which there is one cluster with two synchronized neurons and one singleton. The state variables x_i of all neurons cross threshold and are reset by incoming spikes, thus the perturbation is erased after one cycle (one spike emission per neuron); parameters are $I = 1.05$, $\tau = 1$, $\varepsilon = 0.1$. (b) A saddle periodic orbit in which there is one cluster and one singleton. After the perturbation, the cluster desynchronizes and the system leaves the original orbit; parameters as above with $\tau = 0.63$. (c) The arrow indicates the moment two neurons in (a) resynchronize after a spike arrive. (d) The arrow indicates the moment two neurons in (b) fail to resynchronize after a spike arrive. The spike promotes further desynchronization.

a 'saddle state' and the 'long' orbits resembling sequences of such saddle states as a 'complex (periodic) orbit' [20, 59, 62, 63]. We discuss features of the latter in the following subsection.

4.1.2. Switching dynamics and complex periodic orbits

In the previous section we introduced the concept of saddle periodic orbits. Here we will characterize networks of such saddle states in state space and how they promote encoding of inputs as complex periodic orbits. For illustration, we first consider a small network with $N = 3$ neurons. For parameters $\tau = 0.64$, $c = 0.11$ and $I = 1.05$, this system exhibits saddle states in which there is one cluster with two synchronized neurons with identical states, e.g. $x_1(t) = x_2(t)$ for all times t, and one singleton consisting only of one neuron with state variable component x_3. Such states are also polysynchronous states in the sense of Izhikevich, see reference [64], with permutation symmetry $S_2 \times S_1$. That is, for the example above, their are $\frac{3!}{2!1!} = 3$ saddle states that differ only in their neuron labels but are otherwise identical, in particular tracing out (up to permutation) the same trajectory in state space and exhibiting the same stability properties. For a simple representation, we denote each saddle as a cluster vector

$$(c_1, c_2, c_3) = S_{c_1c_2c_3} \quad \text{with } c_i \in \{a, b\}, \tag{4}$$

where each vector component i indicates if neuron i is part of a cluster, labeled as 'a', or is a singleton, label as 'b'. Therefore, for this specific system, the emerging three saddle states are denoted S_{aab}, S_{aba}, and S_{baa}.

Direct numerical simulations show that these three saddle states form a robust heteroclinic network, i.e. the dynamics will not leave the network after sufficiently small perturbations, but rather switch between states in it. As shown in figure 6, the stability of these saddle states is such that, given a one-time perturbation $\Delta = (\Delta_1, \Delta_2, \Delta_3)$ to the phases of all neurons in a initial saddle state S_{aab}, without loss of generality, either triggers the switch

$$S_{aab} \xrightarrow{(\Delta_1 > \Delta_2, \Delta_3)} S_{baa} \quad \text{or} \quad S_{aab} \xrightarrow{(\Delta_1 < \Delta_2, \Delta_3)} S_{aba}, \tag{5}$$

independently of the perturbation component δ_3 in the singleton direction. Equation (5), thus, defines a basic transition rule between saddle S_{aab} and the other two saddle states. We remark that consistently permuting the indices in equation (5) yields the transition rules starting from any of the two other saddles in this network of states, $S_{aba} \xrightarrow{(\Delta_1 > \Delta_3)} S_{baa}$, $S_{aba} \xrightarrow{(\Delta_1 < \Delta_3)} S_{aab}$, $S_{baa} \xrightarrow{(\Delta_2 > \Delta_3)} S_{aba}$ and $S_{baa} \xrightarrow{(\Delta_2 < \Delta_3)} S_{aab}$. Together, these saddle states with the six possible heteroclinic connections form a (fully connected) network of states, see figure 7, i.e. all states can be reached after one state-switch. In general, i.e. for larger N and other parameter settings, heteroclinic networks are typically not fully connected. Rather, only a small subset of other saddle states is

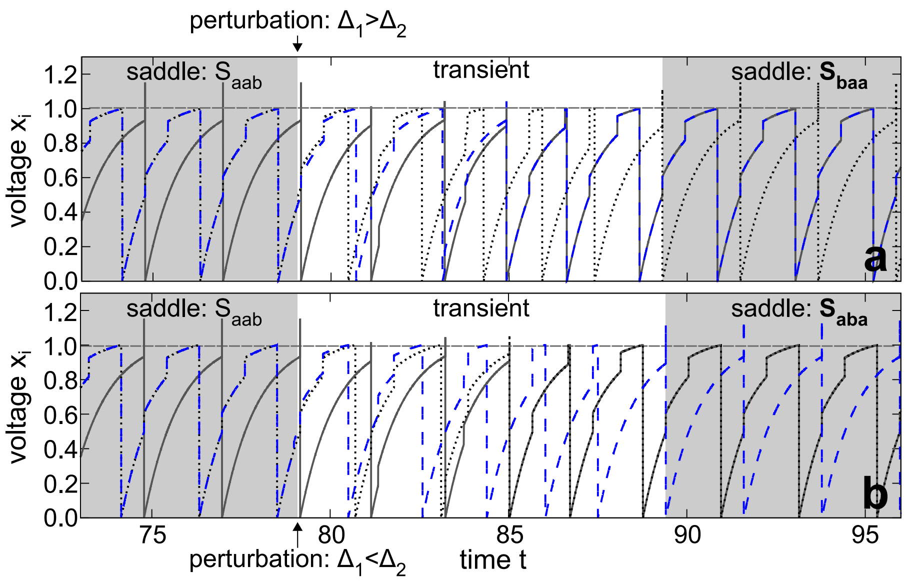

Figure 6. Heteroclinic network response to an instantaneous perturbation. After the system converged to the saddle periodic orbit S_{aab}, a random vector $\Delta = (\Delta_1, \Delta_2, \Delta_3)^\mathsf{T}$ is added to slightly perturb all neurons at time about $t = 79$. (a) If $\Delta_1 > \Delta_2$ the transition $S_{aab} \to S_{baa}$ is observed. (b) If $\Delta_1 < \Delta_2$ the transition $S_{aab} \to S_{aba}$ is observed. In both cases the perturbation on unit 3 decays to zero after one cycle (due to supra-threshold input only). The observed saddle-to-saddle transition occurs on a heteroclinic orbit. Solid, dashed and dotted lines indicate voltages $x_1(t), x_2(t)$, and $x_3(t)$. Display as in this figure, parameters are $I = 1.05, \tau = 0.64, \varepsilon = 0.11$.

reachable from any starting saddle state, see figure 4(b). However, this simple example of $N = 3$ units constitutes an instructive example, because many properties of a larger network, including the number of orbits and states increase combinatorially with the system size, often supra-exponentially. For example the number of saddle states (#saddles) increase as

$$\#\text{saddles}_{m,n,\mu} = \frac{N!}{(n!)^{m-1}(\mu)!}\,\frac{1}{m}, \tag{6}$$

that is, the number of ways we can distribute N elements into $m-1$ sets of size n and one set of size μ, and those sets over a circle. We assume the saddle states have a number $m-1$ cluster of size n and one cluster of size μ.

Above we described how an instantaneous, one-time perturbation induces specific saddle-to-saddle switches in a heteroclinic system. But how to encode time continuous signals using heteroclinic networks? As shown in [20], for sufficiently weak symmetry-breaking input signals, nearly periodic orbits visiting the vicinity of a sequence of (formerly) saddle orbits is established after a transient. The orbit reached is fully determined by the initial network state and its transition rule.

As an example, we schematically depict in figure 7(b) and show in figure 8 how a small, temporally constant symmetry breaking input signal $\xi = (\xi_1, \xi_2, \xi_3)$ with $\xi_3 > \xi_2 > \xi_1$ induces a complex periodic orbit visiting the vicinity of two saddle orbits,

$$S_{aab} \xrightarrow{(\xi_2>\xi_1)} S_{aba} \xrightarrow{(\xi_3>\xi_1)} S_{aab}, \tag{7}$$

independently of details of the initial condition. Applying this continuous signal is equivalent to recursively applying the corresponding one-time perturbation with $\Delta_3 > \Delta_2 > \Delta_1$. This happens because the right-hand side of equation (2) is a simple integration of all sources of current, i.e. only the total difference of integrated current between two neurons in a given interval matters for the effective perturbation. Stated differently, the overall effective differences in the rates of change of the x_i matter.

Which information about the input signal is encoded in this resulting orbit? Observing the orbit without any knowledge about the input signal, we find: after the first switch, that $\xi_2 > \xi_1$ (see transition rules); similarly, after the second switch, that $\xi_3 > \xi_1$. Notice that, it is not known whether ξ_3 is larger or smaller than ξ_2. In fact, the perturbation vector with $\xi_2 > \xi_3 > \xi_1$ yields the same periodic sequence (7). Therefore, this system computes the two strongest inputs out of three, a two-WTA computation over three inputs. The resulting percept (distinguishable set of input) is $\{\xi_3, \xi_2\} > \xi_1$, meaning that both ξ_3 and ξ_2 are larger than ξ_1 but no relation between ξ_2 and ξ_3 is known from observing that complex periodic orbit. Therefore, given a general input signal (ξ_1, ξ_2, ξ_3), the system sorts the set of inputs into two groups, the $k = 2$ strongest inputs and the

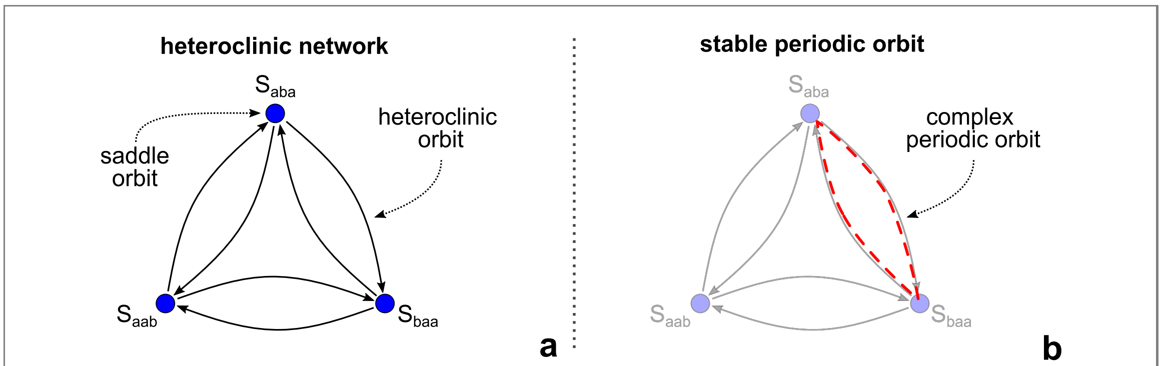

Figure 7. Heteroclinic network and signal-induced complex periodic orbit. (a) A graph representation of a heteroclinic network composed of three saddles and six heteroclinic orbits. ($I = 1.05, \tau = 0.64, \varepsilon = 0.11$). Nodes represent saddles, directed links represent heteroclinic orbits. (b) Input signals with $\xi_1 > \xi_2 > \xi_3$ induce complex (stable) periodic orbit approaching the vicinity where formerly two heteroclinic connections where located, here represented as a walk on the graph.

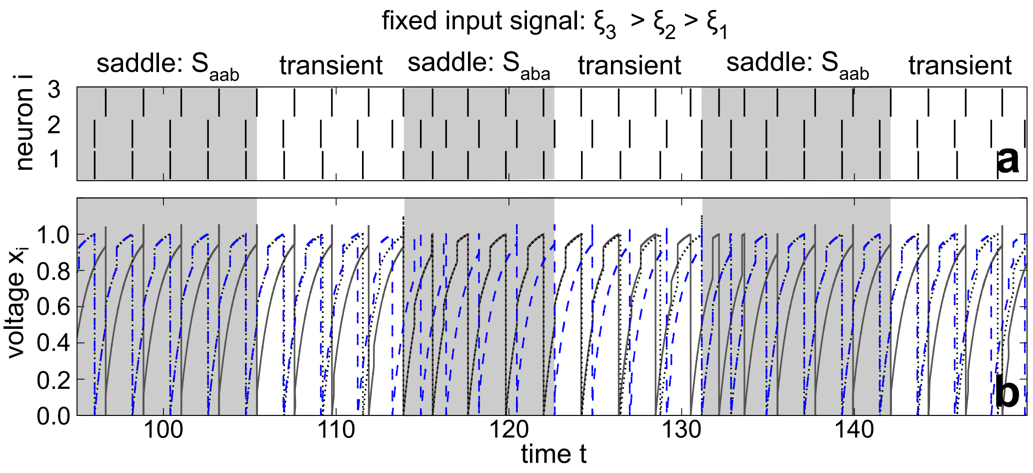

Figure 8. Fixed inputs generating complex periodic orbits $N = 3$. A fixed input signal $\xi = (0, 10^{-5}, 2 \times 10^{-5})$, see equation (2), induces a long periodic orbit approaching the vicinity of two (formerly) saddle states, resembling a cyclic sequence of two states. Gray background represents times in the vicinity of saddles. (a) Each dash represents the time in which the corresponding neuron sent a spike. (b) Each line represents the voltage dynamics of one of the $N = 3$ neurons composing the system. ($I = 1.05$, $\tau = 0.64, \varepsilon = 0.11$).

$N - k = 1$ weakest input. By permuting this result, we find the complete list of possible percepts,

$$\text{for} \quad \{\xi_1, \xi_2\} > \xi_3: \quad S_{aba} \to S_{baa} \to S_{aba}; \tag{8}$$

$$\text{for} \quad \{\xi_3, \xi_2\} > \xi_1: \quad S_{aab} \to S_{aba} \to S_{aab}; \tag{9}$$

$$\text{for} \quad \{\xi_1, \xi_3\} > \xi_2: \quad S_{baa} \to S_{aab} \to S_{baa}. \tag{10}$$

We note that, in particular for larger networks, the observed complex periodic orbit, not the percept, also depends to some degree on the saddle state the system resides in initially.

In summary, reiterated one-time perturbations or continuous signals serving as inputs are encoded in the NN as individual transitions between two saddle states and complex periodic orbits, respectively. Continuous signals enable computing a k-WTA function over the input signals' components. Importantly, this result is not singular to this small system, but a general feature of symmetrical heteroclinic networks of the type presented here. Moreover, systems of (non-spiking) coupled phase oscillators equally exhibit such dynamics, yet for them already an instantaneous symmetry-breaking perturbation induces a repeated switching process among saddles (that slows down with time), see e.g. [56].

Let us generalize the computational mechanism to a NN of size $N \geqslant 3$. The basic building block of symmetric heteroclinic networks of the type studied here, are the symmetry-class preserving switches between saddles. That is, the initial and final states (and their stability properties) are identical up to an index permutation. This constraint leads to saddle states that typically exhibit a total number m of different clusters, i.e. groups of identically synchronized neurons. All but one of these clusters are of the same size n, from which only one is not

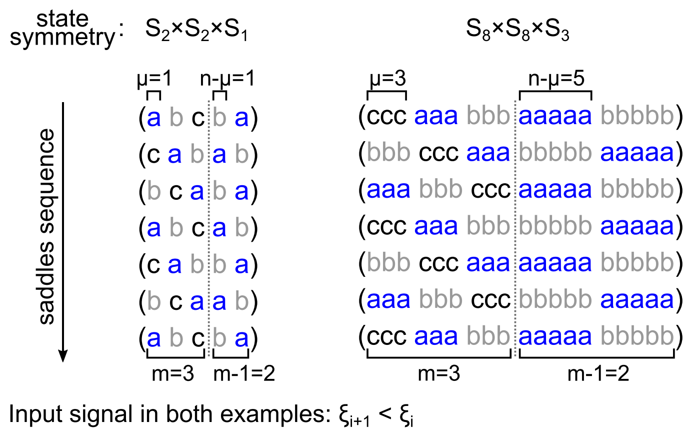
state symmetry:
S2×S2×S1
S8×S8×S3
μ=1
n-μ=1
μ=3
n-μ=5
saddles sequence
(a b c b a)
(c a b a b)
(b c a b a)
(a b c a b)
(c a b b a)
(b c a a b)
(a b c b a)
(ccc aaa bbb aaaaa bbbbb)
(bbb ccc aaa bbbbb aaaaa)
(aaa bbb ccc aaaaa bbbbb)
(ccc aaa bbb bbbbb aaaaa)
(bbb ccc aaa aaaaa bbbbb)
(aaa bbb ccc bbbbb aaaaa)
(ccc aaa bbb aaaaa bbbbb)
m=3
m-1=2
Input signal in both examples: ξi+1 < ξi

Figure 9. Two sequences of states along complex periodic orbits. For input signals $\xi_{i+1} < \xi_i$ and initial states (*ababa*) and (*cccaaabbbaaaaabbbbb*), these sequences are established. In both examples, none of the m sets of μ elements (right of dotted line) ever synchronize among themselves, neither do the $m-1$ sets of $n-\mu$ elements (left of dotted line). Synchronization and desynchronization (comparisons) occurs across set types, either to the left or to the right of the dotted line.

reset by incoming pulses ('unstable cluster'[4]), and a single smaller cluster of size μ, which is always reset by incoming pulses. The total number of neurons in the network then satisfies $N = (m-1)n + \mu$ where $n > \mu$. For the symmetry of such saddle states to be preserved, the 'unstable' cluster must break into two, leaving a new stable small cluster of size μ and the remaining $n-\mu$ neurons must synchronize with the original small cluster, thus being of size n again. After an overall shift in voltage differences between clusters (in relation to the smaller cluster), an originally stable cluster becomes unstable, thus marking the arrival at another saddle state. Thus, each saddle approach reveals which are the k strongest (or the k weakest) input components via the splitting dynamics of the unstable cluster. For example, states with symmetry $S_2 \times S_2 \times S_1$ or $S_8 \times S_8 \times S_3$ could exhibit, respectively, the transition rules

$$S_{aabbc} \xrightarrow{(\xi_{i+i}<\xi_i)} S_{cbaab}, \tag{11}$$

$$S_{aaaaaaaabbbbbbbbccc} \xrightarrow{(\xi_{i+1}<\xi_i)} S_{cccbbbbbbaaaaaaaabbb}, \tag{12}$$

where a labels the unstable cluster and b and c label the stable clusters. As shown for our example with $N = 3$, equations (11) and (12) implicitly encode all possible switching rules between saddles in those networks. In other words, in the first example ($N = 5$, $m = 3$, $n = 2$, and $\mu = 1$), at each approached saddle, the strongest input components over the cluster labeled a is computed whereas in the second example ($N = 19$, $m = 3$, $n = 8$, and $\mu = 3$), the strongest three input components over the cluster labeled a are computed via the splitting of the eight neurons composing this cluster.

What do such networks compute on the systems level, i.e. by combining all the individual computations (switches) along a complex periodic orbit? Let us first consider the two switching rules 11 and 12. For inputs signal $\xi_{i+1} < \xi_i$, the orbits illustrated in figure 9 are achieved.

In the most general case of the class of systems studied here, and exemplified in figure 9, each orbit representing a percept is composed of saddles with $m-1$ clusters of sizes n and one of size μ. These states can be also characterized by m sets of μ neurons that are (almost) always in-set synchronized (during the whole complex periodic orbit) and other $m-1$ sets of $n-\mu$ neurons that are (almost) always synchronized in-set (also during the whole complex periodic orbit). Therefore, the overall computation is restricted to the comparison between each of the m sets of μ elements to the $m-1$ sets of $n-\mu$ elements, which repeatedly synchronize and desynchronize across set types conform the system state-switching rules, see figure 9. In practice, we calculated the computed partial rank order by counting the overall number of elements in each of the types of sets, that is, the computed k is either equal to

$$k = m \times \mu, \tag{13}$$

if the neurons receiving the μ largest inputs over the unstable cluster repeatedly move to form the new small cluster of size μ, or

$$k = (m-1) \times (n-\mu), \tag{14}$$

[4] We remark that mathematically, only states or sets in state space can be stable or unstable and we here refer to one cluster, i.e. subset of components of the state vector, to be 'unstable' meaning that the saddle periodic orbit is unstable to desynchronizing these components.

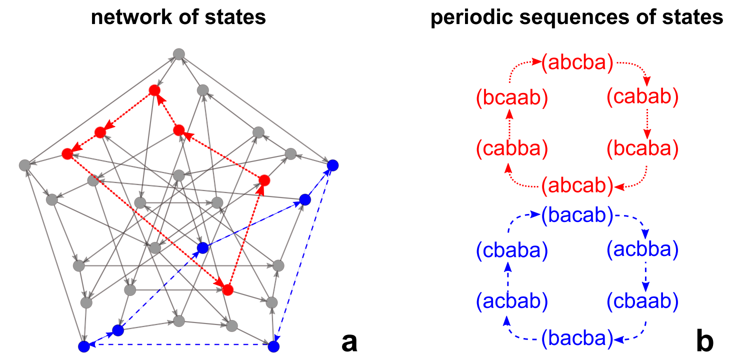

Figure 10. Two sequences of states representing the same percepts. For $N = 5$ and parameters $I = 1.04$, $\tau = 1.59$ and, $\varepsilon = 0.025$, the system exhibit a network of state with transition rule as in equation (11). The resulting network of states is shown in panel (a). For inputs with $\xi_{i+1} < \xi_i$ for all i, one of two periodic orbits approaching one of the two sequences in panel (b) is performed, in doted or dashed arrows. All other states (not present in the sequences) lead to a short transient toward one of the sequences.

if the neurons receiving the smallest inputs move accordingly (this is a system property). For example, in the above example of $N = 5$ network we have $\mu = 1$ and $m = 3$, resulting in $k = m \times \mu = 3$ and for our example with $N = 19$, $\mu = 3$ and $m = 3$, resulting in $k = 9$.

4.1.3. Percept multiplicity and decoding complex periodic orbits

A computationally interesting aspect of all systems computing a k-WTA function is the combinatorial scaling of the number of percepts with the system size. Specifically, the number P_{Nk} of percepts for a system with N inputs computing the largest k inputs is given by

$$P_{Nk} = \frac{N!}{k!(N-k)!}, \tag{15}$$

i.e. the number of ways we can sort N inputs into a groups of k inputs and a another group containing the $N - k$ inputs. Nevertheless, the orbit representation of a percept (spike train combinations) is not always unique. On the contrary, the number of orbits representing the same percept $M_{m,n,\mu}$ increases combinatorially as

$$M_{m,n,\mu} = \frac{((m)(n-\mu))!}{((n-\mu)!)^m}\frac{1}{m} \times \frac{((m+1)\mu)!}{(\mu!)^{m+1}}\frac{1}{(m+1)}, \tag{16}$$

with the system size $N = (m-1)n + \mu$, see reference [65]. Which orbit is established depends solely on the initial state and the fixed input, for an example, see figure 10. Basically, this is the number of ways we can distribute $m \times \mu$ elements inside m sets of size μ and those sets over a circle multiplied by the ways we can distribute $(m-1) \times (n-\mu)$ elements in $m-1$ sets and those sets over a second circle.

This potentially large number of orbits representing the same percept could in principle hinder a heteroclinic computing feasibility by requiring readouts of high complexity, large size, or requiring an combinatorially increasing time for decoding what the complex periodic orbit may symbolize. Recent research [65] shows that this is not the case and that readouts growing linearly with the number of neurons on the system exhibiting the heteroclinic network are possible to construct. We now describe the general building blocks of such decoding schemes.

An effective readout, thus, must cope with the exponential increase of percepts with system size and, at the same time, with the exponential increase of orbits representing the same percept (multiplicity). The solution is twofold. First, we identify properties common to all orbits representing the same percept. It was shown in [65] that one such marker is the small cluster and its composing neurons, because either only neurons receiving the k strongest inputs or only the $N - k$ receiving the weakest inputs ever assume such position (independent of system details). Thus, in principle, any readout capable of discerning between a set of (almost) concurrent pulses of size n and another set of (almost) concurrent pulses of size μ should be capable of decoding the percept. Second, due to the NN symmetry, all orbits representing percepts are identical after a proper neuron index permutation, thus the connectivity to a readout can also be generated via such a permutation. Together, this solution provides an overall readout whose size or complexity increase linearly with the system size and generate a simple binary output, i.e. spike trains emitted, representing an ‘on’ state and no spikes emitted, representing an ‘off’ state.

Figure 11 shows one simple decoder that uses a suitable ratio between inhibitory connections and excitatory ones to detect the percepts. It is composed of N neurons, one for each neuron in the main system. Neurons with the same index have a excitatory connection, neurons with different indices have inhibitory connections.

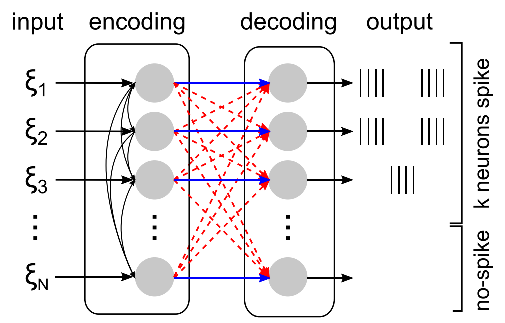

Figure 11. Schematic of a neural decoder for heteroclinic computing. Blue solid arrows represent excitatory connections; red dashed arrows represent inhibitory connections. If sufficiently many inhibitory pulses arrive at the same time as an excitatory one, they prevent a output pulse; if less than n inhibitory pulses arrive at (almost) the same time as an excitatory one, a pulse is elicited.

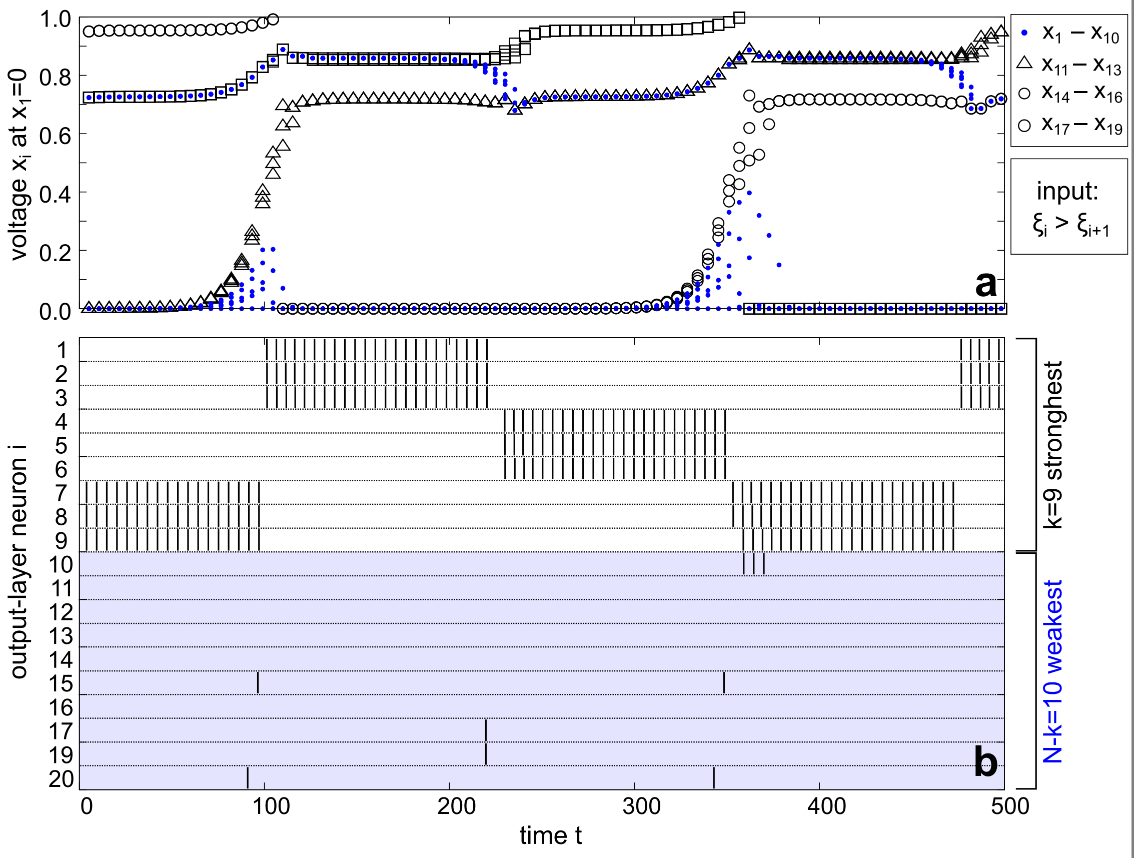

Figure 12. Heteroclinic dynamics and decoder output. A network of $N = 19$ neurons and saddle state symmetry $S_8 \times S_8 \times S_3$, encoding an $(k = 9)$-WTA function. (a) The voltages x_i of all neurons in the main system sampled whenever $x_1(t) = 0$. It shows the dynamic cluster formation and reformation. (b) The output of the decoder. The overwhelming majority of spikes are sent by the neurons receiving the $k = 9$ strongest inputs. Network parameters: $I = 1.04$, $\tau = 1.59$, $\gamma = 1$ and $\varepsilon = 0.0058$. Readout parameters: $I = 1.04$, $\tau = 1.61$, $\gamma = 1.8$, inhibitory coupling strength $\varepsilon^- = -0.1$, excitatory coupling strength $\varepsilon^+ = 1.4$ and $\tau_{\text{dec}} = 0.29$.

The coupling strengths to the readout neuron are such that if μ or less pulses arrive (almost) at the same time at a neuron in the readout, a pulse is generated; if more than μ (almost) synchronized pulses arrive, no pulse is generated. The readout layer has no internal (recurrent) connections. Because it is composed of the same neuron model as the main system, the readout adds only three new parameters to the system, the global value of the inhibitory ε^- and excitatory ε^+ coupling strengths, and a small delay for the excitatory connections τ_{dec} to guarantee that all inhibitory pulses are computed before the (only) excitatory one (for almost synchronized neurons).

In figure 12, we present an example applied to a network of $N = 19$ neurons exhibiting state symmetry $S_8 \times S_8 \times S_3$ and transition rules as in equation (12). Apart from a small number of false positives (spikes where theory predicts none in an ideal, noiseless), three sets of three neurons clearly fire the most, thereby, decoding the signal. As demonstrated before [65], one more layer with a single excitatory connection per neuron, between neurons with the same index, may serve as a high-pass filter and, thus, exploit the difference in output frequencies to avoid any false positive.

4.2. Dynamical inhibition

As an alternative to k-WTA via heteroclinic dynamics, reconfigurable k-WTA may be implemented by fully symmetrical networks of inhibitory neurons. Specifically, recent work [29] demonstrated that proportional inhibition is suitable to create reconfigurable computational units, where k can be nearly freely chosen by varying one parameter. In the example we discuss here, we set $\varepsilon(x_i)$ in equation (2) to

$$\varepsilon(x_i) = -cx_i, \tag{17}$$

where c is a global constant between zero and one and x_i is the voltage of the neuron receiving the inhibitory pulse. In this model, $\tau = 0$ and $I > 1$, such that each neuron in isolation becomes oscillatory by repeatedly reaching the spiking threshold at value $x^\theta = 1$. In contrast to other implementations that require a careful parameter tuning to select k, in these networks of inhibitory neurons k is a simple monotonic function of the global scaling coupling constant c (for fixed input components differences), and thereby reconfigurable.

The overall dynamics is driven by two coacting effects, see figure 13. The first is a compression of voltage differences $x_i - x_j$ after a spike is elicited. Each time a neuron is reset, its inhibitory spike will decrease the voltage of all other neurons. Because this decrease is proportional to the voltage of the receiving neuron, all voltage differences between neurons will also decrease (not including the neuron emitting the spike). The second is the voltage separation caused by the reset itself. Apart from very large c values, the voltage differences between the neuron emitting the spike and all other neurons will typically increase because, due to the concavity (down) of the voltages' curves, more time is spent close to $x_i = 1$ than zero. Figure 13 shows the compression and separation of voltage differences taking place for a network of $N = 4$ neurons. At the moment neuron one (blue solid line) reaches the threshold and is reset to zero, all other neurons are inhibited and their voltages reduced to about 70%. Their voltages differences are thus also reduced to 70% of the original.

In the absence of external inputs, the system is symmetrical but its initial condition (voltage of each neuron) generically is not. The intrinsic frequency of all neurons are identical. Under these conditions, the system self-organizes in such manner that all neurons spike orderly and sequentially, see figure 14. The order itself is defined entirely by the initial voltage order, i.e. neurons with larger initial voltages fire first. Intuitively, such an ordering should emerge, because if the frequency of all neurons are the same (same speed), no neuron can ever overtake another, therefore all neurons spike sequentially. Our ongoing work [66] indicates that not only neurons spike sequentially, but also the inter-spike interval is the same, independent of the initial neuronal voltages, because it is the only non-trivial stable attractor. Pulses arriving too late or too early would trigger a self-correction, as they would either cause a larger decrease in voltage or smaller, respectively, than if it would arrive in time, see the first reset illustrated in figure 14(b). The exact inter-spike interval is a (parameter dependent) system property and variations of the inter-spike interval duration decrease exponential with each cycle.

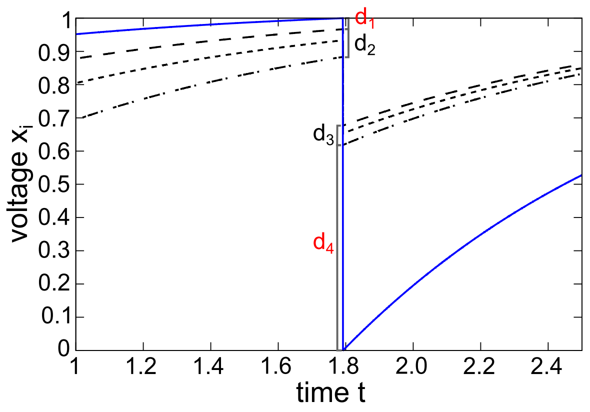

Figure 13. Dynamic compression and separation of voltages differences. The voltage x_i of each of the $N = 4$ neurons as a function of time. The smallest voltage difference d_1 between the neuron being reset and the others, measured before the reset is smaller than d_4 measured after the reset, a separation occurs and $d_4 > d_1$. The largest voltage difference d_2 between the neurons not being reset measured before the reset is larger than this difference d_3 measured after the reset, a compression occurs and $d_3 < d_2$. ($I = 1.04$, $\tau = 0$, $c = -0.3$).

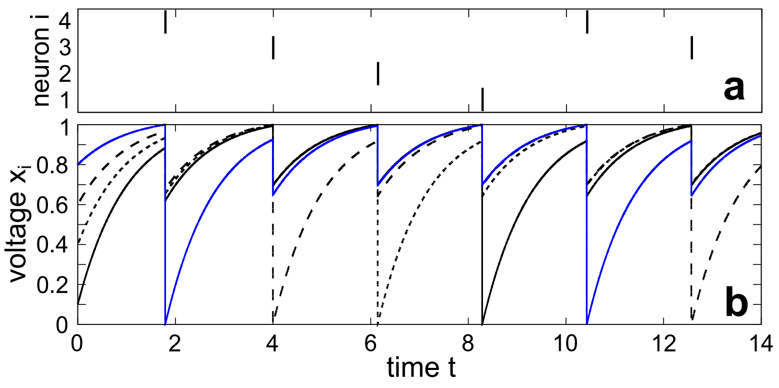

Figure 14. Self-organized (almost) equally spaced spike sequence. (a) The spike sequence generated by a network of $N = 4$ neurons in the absence of input signals. (b) The voltage x_i of all neurons as a function of time. The initial voltages are set as shown at $t = 0$. ($I = 1.04, \tau = 0, c = -0.3$).

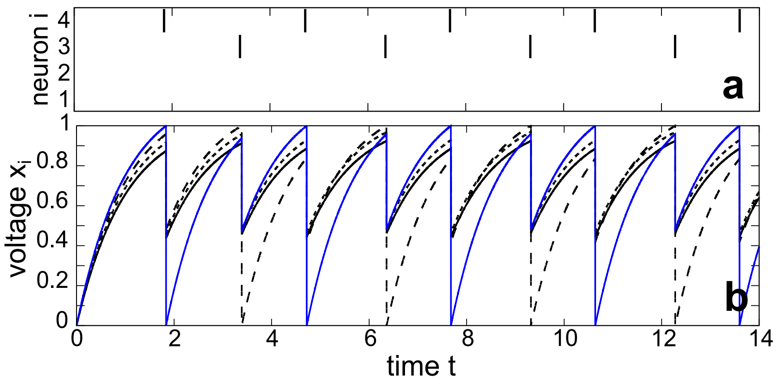

Figure 15. Two-WTA over four input signals. (a) The spike sequence generated by a network of $N = 4$ neurons in for input signals ξ_4. (b) The voltages x_i of all neurons as a function of time. The initial voltages are set as shown at $t = 0$. ($I = 1.04, \tau = 0, c = -0.3$).

Even more interestingly, k-WTA computations may take place if a subset of k neurons, due to external input signals, are able to overtake the other $N - k$ neurons, and prevent the spikes from the later group. Although overtaking has been observed an explained in a class of spiking neural oscillator networks with symmetry [67], it remains unclear under which conditions overtaking may occur in symmetry-broken systems, depending on the input current differences $\xi_i - \xi_j$ and the value of the network parameter c. The simplest case occurs when input currents ξ_i exhibit large differences $\xi_i - \xi_j$, what would, in turn, induce neurons to have large differences in their intrinsic frequencies. If the differences are large enough, independently of c, one or more neurons (the k winners) may repeatedly overtake the others (the $N - k$ losers) by simply being much faster. A second, more reliable way is to set $c > 0$, which allow a broader range of input differences, also only slightly faster neurons, to overtake others. The resulting voltage separation from the compression combined with the voltages set back from the resets promotes overtaking by slightly faster neurons. See figure 15 for an example. When the computation is taking place, k neurons fire in sequence, while the other $N - k$ neurons are kept sub-threshold, even if, at times, they may stay always at large voltages close to $x_i = 1$.

The computed k is controllable, as shown numerically [29], and analytically predictable, see reference [66]. The precise combination of the c value and the input differences $\xi_i - \xi_j$ defines which k is computed by the system. Interestingly, as shown in figure 16 (calculated numerically), the computed k's appear as horizontal bands in parameter space, thus computing the same k for a broad range of input differences. This is fortunate, because if the interval of possible input differences is fixed, it is possible to reliably reconfigure the system computation, the k, by simply adjusting the single global parameter c. For our specific example with networks size of $N = 4$, up to four neurons may spike as a response to an input signal, so $k \in \{1, 2, 3, 4\}$.

Another particular feature of this system class is the computation's independence of the network size. That is, the computed k value only depends on c and the $\Delta\xi$ and not on N. This implies that not only the dynamics but the exact computation is preserved after adding or removing neurons from the network (up to removing $N - k$ neurons). This is possible because the inhibited neurons effectively do not participate in the global network dynamics, as spiking NNs only exchange information via spikes. The same feature that provides such

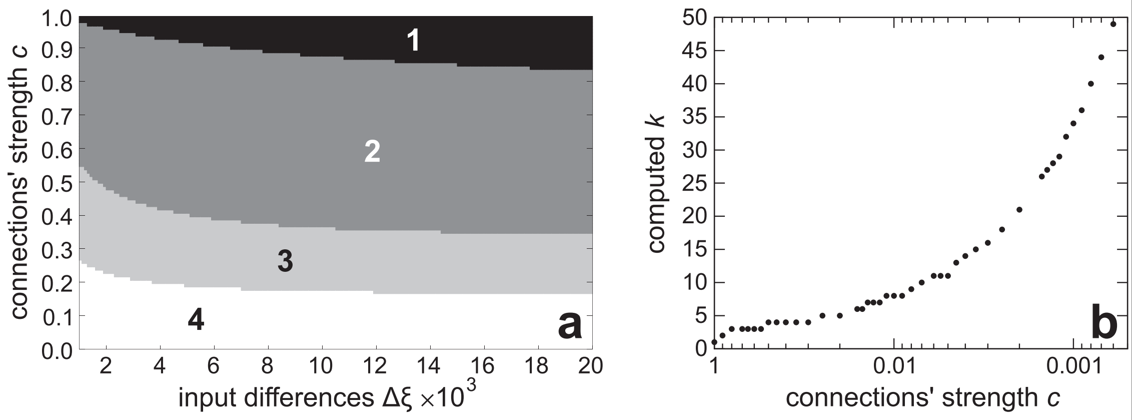

Figure 16. Robust and reconfigurable computation shown in parameter space. Computed k as a function of the coupling parameter c and input signal differences $\Delta\xi$ for networks of size $N = 4$ and $N = 50$. (a) The computed k as a function of c and the input difference $\Delta\xi$ in an $N = 4$ network. The computed k appears in broad bands, showing robustness in parameter space for small k. ($I = 1.04$, $V_i(0) = 0$ for all i). (b) The computation of larger k requires an ever increasing precision on c. In (b), inputs were fixed and linearly distributed between 0 and 0.003. ($\Delta\xi = 6 \times 10^{-5}$, $I = 1.04$ and $V_i(0) = 0$ for all i).

resilient computation, imposes a practical limit on the possible k values computable. As show in figure 16(b), selecting larger k requires an ever increasing precision on c. From another perspective, because the computed k is independent of the network size, the results in figure 16(a) must hold for larger N values apart from the region marked as $k = 4$, which must nest all regions computing $k > 3$.

Finally, this approach does not require a readout, because only the neurons receiving the k strongest inputs spike. Therefore, already in a usable form either for interpretation or for transmission for further processing.

5. Discussion

Bio-inspired computing is an ever-growing field branching in a variety of directions, from robotics to data analysis and neurosciences. An exhaustive review of bio-inspired computing, off course, is out of the scope of this work, as it would not be feasible in any single article. By looking back toward some of its core roots, we outlined, compared and discussed some of its major branches.

The two first sections of this article attempts to convey how complementary different bio-inspired computing concepts can be. For example, ML approaches are deeply rooted in layered networks, for which powerful learning rules are available. Pattern generators, in contrast, may be more suitably realized by recurrent networks, for which the emergence of collective nonlinear dynamics is an intrinsic property. Neither approach, though, could optimally replace the other.

The third section is dedicated to circuits of spiking neurons. We have highlighted how the same approach of studying simplified neuronal models may help not only unveiling the underlying mechanism promoting computation on natural systems but also implementing artificial computing models. This acquired knowledge, in turn, may inspire unprecedented approaches to artificial computing.

We dedicated section four to present, in some depth, two NN approaches to implement discrete decision-making via k-WTA dynamics, namely heteroclinic computing and dynamical inhibition. Heteroclinic computing exploits the dynamics near networks of unstable states to compute a partial rank order of the inputs. Related works have suggested heteroclinic dynamics may play a fundamental role in modeling and encoding information in the brain [11, 18, 68–71]. We analyzed fundamental aspects of this encoding, in particular complex multi-dimensional spike trains, and how to decode it to a human friendly representation with simple, low-complexity decoders. Dynamical inhibition provides a novel way to implement k-WTA, where neuronal oscillators that have (at least) slightly higher frequencies, repeatedly inhibit some neurons with lower frequencies, such that only the former spike. The inhibitory coupling proportional to the current voltage $x_i(t)$ of the neuron receiving the spike at time t, yield systems that are capable of computing the same function for a broad range of input differences.

These implementations add to a broader context of dynamical k-WTA computation, for example via bump states [72], dynamical neural fields [73] or cross inhibition [74]. Even though they all implement an identical set of k-WTA functions, each paradigm brings its own particular features. Heteroclinic computing exhibit sensitivity to arbitrarily small inputs (down to the noise level and parameters precision), while dynamical inhibition provides re-configurable computations (k selection) via a single global parameter. Alternatively, dynamical neural field implementations not discussed here provide soft-WTA computations that are robust to different sources of perturbations, because they rely on the average activities of local and global populations [72, 73, 75].

We hope this article serves as a brief introduction to some of the variety and richness of bio-inspired computing models and how they may complement each other. In particular, we put in perspective the recent results on discrete decision making via k-WTA with dynamical systems. We believe that such systems are suitable as NN implementations, also in hardware, as they naturally compute discrete (binary) functions out of multi-dimensional and multi-modal analogue signals and, thus, may well play a role in the next generation of intelligent machines.

Acknowledgments

Partially supported by the German Research Foundation (Deutsche Forschungsgemeinschaft, DFG) under Project Number 419424741 and under Germany's Excellence Strategy—EXC-2068—390729961—Cluster of Excellence Physics of Life at TU Dresden, and the Saxonian State Ministry for Science, Culture and Tourism under Grant Number 100400118.

Data availability statement

All data that support the findings of this study are included within the article (and any supplementary files).

Appendix A. Piecewise analytical integration and initial conditions

All simulations in this article have been performed via a piecewise analytical integration of equation (2). Notice that, between any two discontinuous evens, that is, pulse receptions and neurons' resets, the neurons are effectively uncoupled and this equation has an analytical solution. When uncoupled, because $V(t)$ is a monotonically increasing function of t, until it reaches a fixed threshold (where it resets), each neuron has a well defined oscillation period T and we can, thus, map the voltages into an internal 'clock' variable $\rho \in [0, T[$, which is also reset to zero each time $V \to 0$, see [76] for details. To update the network state for a time Δt we first map the starting voltage V_0, in the considered interval, into the internal 'clock', which for our IFN is given by

$$\rho_0 = \frac{1}{\gamma}\log\left(\frac{I}{I - \gamma V_0}\right); \tag{A.1}$$

second, we advance the 'clock' Δt units of time,

$$\rho_+ = \rho_0 + \Delta t; \tag{A.2}$$

and third, we map the result back to the state variable V via

$$V_+ = \frac{I}{\gamma}(1 - \exp^{-\gamma\rho_+}), \tag{A.3}$$

where V_+ is the voltage at the end of the time interval. Thus, by computing the times in which such events take place, we can integrate the system analytically for the time step between two consecutive events. After each of these time steps (at the event times), we compute the appropriate contribution from the arriving pulses (see equation (2)) or resets ($V \to 0$). Altogether, these update sequence yields the system's time evolution.

To keep this stepwise analytical approach also when noise is applied, thus avoiding numerical integration, we approximate a Gaussian noise source by only evaluating its effects after small intervals of time δt, such that the overall dynamics is still a sequence of discrete events. To avoid the effects of numerical synchronization, which could lead to large deviations in the system's dynamics (which relies on synchronization or lack thereof), the sampled times at which the noise is evaluated are not fixed, but drawn from a Poisson distribution (independently for each neuron), yielding a variable time step with a well defined mean value. The noise contribution η after an interval δt is modeled as:

$$\eta = \sqrt{\delta t} N_{\text{rand}}(\sigma, 0), \tag{A.4}$$

where $N_{\text{rand}}(\sigma, 0)$ is a random number drawn from a normal distribution with variance σ and centered at zero. In practice, our algorithm simply checks for the next event among: neurons reaching the threshold, a pulse arriving in a neuron or a scheduled noise update. After that, the new state is analytically updated until the time of the next event, and the contribution of the event itself is computed.

Concerning the initial conditions, the initial 'voltages' in our simulations are typically close to the orbit being investigated and can be read from each figure at time $t = 0$ if not stated otherwise. Notice though, that for heteroclinic computing the exact value of the initial voltage differences between clusters are not as important for the initial condition, because the stability properties relies on the 'in-cluster' voltage differences, i.e. starting with the correct clustering will typically lead to the correct orbit in a self-organized manner (part of the stable manifold) [57].

ORCID iDs

Fabio S Neves https://orcid.org/0000-0001-5583-1328

References

[1] Lynn C W and Bassett D S 2019 The physics of brain network structure, function and control *Nat. Rev. Phys.* **1** 318–32
[2] Amit D J 1989 *Modeling Brain Function: The World of Attractor Neural Networks* (Cambridge: Cambridge University Press)
[3] Chialvo D R 2010 Emergent complex neural dynamics *Nat. Phys.* **6** 744–50
[4] Rolls E T and Deco G 2010 *The Noisy Brain: Stochastic Dynamics as a Principle of Brain Function* (Cambridge: Cambridge University Press)
[5] Emmert-Streib F, Yang Z, Feng H, Tripathi S and Dehmer M 2020 An introductory review of deep learning for prediction models with big data *Front. Artif. Intell.* **3** 4
[6] Fahle S, Prinz C and Kuhlenkötter B 2020 Systematic review on machine learning (ML) methods for manufacturing processes—identifying artificial intelligence (AI) methods for field application *Procedia CIRP* **93** 413–8
[7] Sharma N, Sharma R and Jindal N 2021 Machine learning and deep learning applications-a vision *Glob. Transit. Proc.* **2** 24–8
[8] Bishop C M 2006 *Pattern Recognition and Machine Learning* (*Information Science and Statistics*) (Berlin: Springer)
[9] Campbell M, Hoane A J and Hsu F-h 2002 Deep blue *Artif. Intell.* **134** 57–83
[10] Silver D *et al* 2017 Mastering the game of go without human knowledge *Nature* **550** 354–9
[11] Rabinovich M I, Varona P, Selverston A I and Abarbanel H D I 2006 Dynamical principles in neuroscience *Rev. Mod. Phys.* **78** 1213
[12] Mastrandrea R, Gabrielli A, Piras F, Spalletta G, Caldarelli G and Gili T 2017 Organization and hierarchy of the human functional brain network lead to a chain-like core *Sci. Rep.* **7** 4888
[13] Huang Z J and Paul A 2019 The diversity of GABAergic neurons and neural communication elements *Nat. Rev. Neurosci.* **20** 563–72
[14] Hopfield J J 1982 Neural networks and physical systems with emergent collective computational abilities *Proc. Natl Acad. Sci.* **79** 2554–8
[15] Bohte S M 2004 The evidence for neural information processing with precise spike-times: a survey *Nat. Comput.* **3** 195–206
[16] Rabinovich M, Volkovskii A, Lecanda P, Huerta R, Abarbanel H D I and Laurent G 2001 Dynamical encoding by networks of competing neuron groups: winnerless competition *Phys. Rev. Lett.* **87** 068102
[17] Ashwin P and Borresen J 2005 Discrete computation using a perturbed heteroclinic network *Phys. Lett.* A **347** 208–14
[18] Rabinovich M, Huerta R and Laurent G 2008 Transient dynamics for neural processing *Science* **321** 48–50
[19] Buckley C L and Nowotny T 2011 Transient dynamics between displaced fixed points: an alternate nonlinear dynamical framework for olfaction *BMC Neurosci.* **12** P237
[20] Neves F S and Timme M 2012 Computation by switching in complex networks of states *Phys. Rev. Lett.* **109** 018701
[21] Lukoševičius M and Jaeger H 2009 Reservoir computing approaches to recurrent neural network training *Comput. Sci. Rev.* **3** 127–49
[22] Barbier T, Teuliere C and Triesch J 2021 Spike timing-based unsupervised learning of orientation, disparity, and motion representations in a spiking neural network *Proc. IEEE Computer Society Conf. Computer Vision and Pattern Recognition Workshops* pp 1377–86
[23] Martin L, Sándor B and Gros C 2016 Closed-loop robots driven by short-term synaptic plasticity: emergent explorative vs limit-cycle locomotion *Front. Neurorobot.* **10** 12
[24] Ijspeert A J 2008 Central pattern generators for locomotion control in animals and robots: a review *Neural Netw.* **21** 642–53
[25] Gerstner W and Kistler W M 2002 *Spiking Neuron Models: Single Neurons, Populations, Plasticity* (Cambridge: Cambridge University Press)
[26] Paugam-Moisy H and Bohte S 2012 *Computing with Spiking Neuron Networks* (Berlin: Springer) pp 335–76
[27] Maass W 1999 Neural computation with winner-take-all as the only nonlinear operation *NIPS'99: Proc. 12th Int. Conf. Neural Information Processing Systems* (Cambridge, MA: MIT Press)
[28] Wordsworth J and Ashwin P 2008 Spatiotemporal coding of inputs for a system of globally coupled phase oscillators *Phys. Rev.* E **78** 066203
[29] Neves F S and Timme M 2020 Reconfigurable computation in spiking neural networks *IEEE Access* **8** 179648–55
[30] Hertz J, Krogh A and Palmer R G 2018 *Introduction to the Theory of Neural Computation* (Boca Raton, FL: CRC Press)
[31] Sun R (ed) 2008 *The Cambridge Handbook of Computational Psychology* (*Cambridge Handbooks in Psychology*) (Cambridge: Cambridge University Press)
[32] McCulloch W S and Pitts W 1943 A logical calculus of the ideas immanent in nervous activity *Bull. Math. Biophys.* **5** 115–33
[33] Hebb D O 1949 *Organization of Behavior: A Neuropsychological Theory* (New York: Wiley)
[34] Elman J L 1993 Learning and development in neural networks: the importance of starting small *Cognition* **48** 71–99
[35] Rumelhart D E and McClelland J L (ed) 1986 *Parallel Distributed Processing: Explorations in the Microstructure of Cognition* (*Foundations*)vol 1) (Cambridge, MA: MIT Press)
[36] Rosenblatt F 1958 The perceptron: a probabilistic model for information storage and organization in the brain *Psychol. Rev.* **65** 386–408
[37] Rumelhart D E and McClelland J L 1987 *Learning Internal Representations by Error Propagation* (Cambridge, MA: MIT Press) pp 318–62
[38] Mozer M 1989 A focused backpropagation algorithm for temporal pattern recognition *Complex Syst.* **3** 349–81

[39] Maass W, Natschläger T and Markram H 2002 Real-time computing without stable states: a new framework for neural computation based on perturbations *Neural Comput.* **14** 2531–60

[40] Jaeger H 2001 The 'echo state' approach to analysing and training recurrent neural networks *Technical Report, GMD Report 148* (German National Research Center for Information Technology)

[41] Jaeger H and Haas H 2004 Harnessing nonlinearity: predicting chaotic systems and saving energy in wireless communication *Science* **304** 78–80

[42] Tanaka G, Yamane T, Héroux J B, Nakane R, Kanazawa N, Takeda S, Numata H, Nakano D and Hirose A 2019 Recent advances in physical reservoir computing: a review *Neural Netw.* **115** 100–23

[43] Larger L, Soriano M C, Brunner D, Appeltant L, Gutierrez J M, Pesquera L, Mirasso C R and Fischer I 2012 Photonic information processing beyond turing: an optoelectronic implementation of reservoir computing *Opt. Express* **20** 3241–9

[44] Brunner D, Soriano M C and Van der Sande G 2019 *Photonic Reservoir Computing* (Berlin: de Gruyter and Co)

[45] Appeltant L, Soriano M C, Van der Sande G, Danckaert J, Massar S, Dambre J, Schrauwen B, Mirasso C R and Fischer I 2011 Information processing using a single dynamical node as complex system *Nat. Commun.* **2** 1–6

[46] Jaeger H 2017 Controlling recurrent neural networks by conceptors arXiv:1403.3369v3 [cs.NE]

[47] Bucher D 2009 Central pattern generators *Encyclopedia of Neuroscience* ed L R Squire (New York: Academic) pp 691–700

[48] Ijspeert A J, Crespi A and Cabelguen J-M 2005 Simulation and robotics studies of salamander locomotion: applying neurobiological principles to the control of locomotion in robots *Neuroinformatics* **3** 171–96

[49] Steingrube S, Timme M, Wörgötter F and Manoonpong P 2010 Self-organized adaptation of a simple neural circuit enables complex robot behaviour *Nat. Phys.* **6** 224–30

[50] Thorpe S J 1990 Spike arrival times: a highly efficient coding scheme for neural networks *Parallel Processing in Neural Systems* pp 91–4

[51] Kirst C and Timme M 2009 How precise is the timing of action potentials? *Front. Neurosci.* **3** 9

[52] Cofré R, Maldonado C and Cessac B 2020 Thermodynamic formalism in neuronal dynamics and spike train statistics *Entropy* **22** 1330

[53] Ashwin P and Borresen J 2004 Encoding via conjugate symmetries of slow oscillations for globally coupled oscillators *Phys. Rev.* E **70** 026203

[54] Uhlhaas P J, Pipa G, Lima B, Melloni L, Neuenschwander S, Nikolić D and Singer W 2009 Neural synchrony in cortical networks: history, concept and current status *Front. Integr. Neurosci.* **3** 17

[55] Maass W 2000 On the computational power of winner-take-all *Neural Comput.* **12** 2519–35

[56] Krupa M 1997 Robust heteroclinic cycles *J. Nonlinear Sci.* **7** 129–76

[57] Neves F S and Timme M 2009 Controlled perturbation-induced switching in pulse-coupled oscillator networks *J. Phys. A: Math. Theor.* **42** 345103

[58] Neves F S 2010 Universal computation and memory by neural switching *PhD Thesis* (Division of Mathematics and Natural Sciences Georg-August-University, Göttingen)

[59] Timme M, Wolf F and Geisel T 2003 Unstable attractors induce perpetual synchronization and desynchronization *Chaos* **13** 377–87

[60] Ashwin P and Timme M 2005 Unstable attractors: existence and robustness in networks of oscillators with delayed pulse coupling *Nonlinearity* **18** 2035–60

[61] Ernst U, Pawelzik K and Geisel T 1998 Delay-induced multistable synchronization of biological oscillators *Phys. Rev.* E **57** 2150

[62] Timme M, Wolf F and Geisel T 2002 Prevalence of unstable attractors in networks of pulse-coupled oscillators *Phys. Rev. Lett.* **89** 154105

[63] Kirst C, Geisel T and Timme M 2009 Sequential desynchronization in networks of spikin neurons with partial resets *Phys. Rev. Lett.* **102** 068101

[64] Izhikevich E M 2006 Polychronization: computation with spikes *Neural Comput.* **18** 245–82

[65] Neves F S and Timme M 2021 Decoding complex state space trajectories for neural computing *Chaos* (accepted)

[66] Börner G, Neves F S and Timme M Self-correctiing dynamics in reconfigurable spiking neural circuits (in preparation, expected date 2022)

[67] Kielblock H, Kirst C and Timme M 2011 Breakdown of order preservation in symmetric oscillator networks with pulse-coupling *Chaos* **21** 025113

[68] Laurent G, Stopfer M, Friedrich R W, Rabinovich M I, Volkovskii A and Abarbanel H D 2001 Odor encoding as an active, dynamical process: experiments, computation, and theory *Annu. Rev. Neurosci.* **24** 263–97

[69] Afraimovich V S, Rabinovich M I and Varona P 2004 Heteroclinic contours in neural ensembles and the winnerless competition principle *Int. J. Bifurcation Chaos* **14** 1195–208

[70] Huerta R, Nowotny T, García-Sanchez M, Abarbanel H D I and Rabinovich M I 2004 Learning classification in the olfactory system of insects *Neural Comput.* **16** 1601–40

[71] Bick C and Rabinovich M I 2009 Dynamical origin of the effective storage capacity in the Brain's working memory *Phys. Rev. Lett.* **103** 218101

[72] Laing C R and Chow C C 2001 Stationary bumps in networks of spiking neurons *Neural Comput.* **13** 1473–94

[73] Sandamirskaya Y 2014 Dynamic neural fields as a step toward cognitive neuromorphic architectures *Front. Neurosci.* **7** 276

[74] Chen Y 2017 Mechanisms of winner-take-all and group selection in neuronal spiking networks *Front. Comput. Neurosci.* **11** 20

[75] Sandamirskaya Y, Richter M and Schöner G 2011 A neural-dynamic architecture for behavioral organization of an embodied agent *IEEE Int. Conf. Development and Learning (ICDL)* vol 2 pp 1–7

[76] Mirollo R E and Strogatz S H 1990 Synchronization of pulse-coupled biological oscillators *SIAM J. Appl. Math.* **50** 1645–62