JACOBS
UNIVERSITY

Fatemeh Hadaeghi, Xu He, Herbert Jaeger

Unconventional Information Processing Systems, Novel Hardware: A Tour d'Horizon

Technical Report No. 36
July 2017

Dept. of Computer Science and Electrical Engineering

Unconventional Information Processing Systems, Novel Hardware: A Tour d'Horizon

Fatemeh Hadaeghi, Xu He, Herbert Jaeger

Jacobs University Bremen gGmbH
Campus Ring 12
28759 Bremen
Germany

E-Mail: h.jaeger@jacobs-university.de
http://minds.jacobs-university.de/

Abstract

This report provides a wide-angle survey on computational paradigms which have a possible bearing on the development of unconventional computational substrates and hardware devices. Such unconvential substrates and devices have some properties that alienate them from the classical Turing model of computation. Among other challenging characteristics, they are non-digital, unclocked, low-precision, exhibit static and dynamic parameter drift, and may have limited lifetime. Such properties are shared with biological computing systems – brains, but not only brains – , so this survey includes ideas and insights from neuroscience and the natural computing field.

Contents

1 Introduction

This report is a deliverable within the NeuRAM3 H2020 project (http://neuram3.vsos.ethz.ch/). In its present version, the manuscript is essentially a minimally commented material collection. The emphasis lies on computational paradigms, algorithms and architectures, unconventional candidate physical substrates and technological challenges in designing an information processing system. Even at this premature stage the report illustrates how much thinking has already been invested in a wide span of disciplines. The final version will be significantly more detailed and didactic, aiming at becoming a self-contained interdisciplinary study reference where material scientists and novel device engineers can conveniently and thoroughly instruct themselves about the multi-faceted worlds of computing theory, cognitive computing, and neural computing; and where, conversely, researchers in the wider computational sciences find an accessible introduction to the world of novel computational substrates. Thus, please, consider this report only as a first instalment of a future delivery, and be forgiving when you spot its imperfections, of which there are many.

The principal purpose of our efforts is to provide a comprehensive overview of existing information processing paradigms (IPPs) which already have seen physical realizations or at least there appear chances for them. This excludes some IPPs of purely theoretical interest in complexity theory and hypercomputing [Burgin and Dodig-Crnkovic, 2013], for instance oracle machines or abstract machines that command on infinite-precision arithmetics or arbitrary speedups. Since the word "computing" is too tightly connected to certain classical concepts of mathematics and computer science, in order to address models of computation or simply physically reliable techniques of problem solving, we preferred to explore them as "models of information processing" rather than models of computing. The term "information processing paradigm (IPP)", therefore, accounts for the abstract models of information processing as diverse as the Turing model, quantum computing, analog signal processing, and many others. It is worth mentioning that we define IPPs intensionally, by their "idea", not extensionally, by the problems that can be solved by them. Specifically, many would argue or believe that any physically realizable IPP can be simulated by a Turing Machine (TM), and conversely, TMs can be "coded into" most – but not all – of currently researched IPPs. We also note that some IPPs have been originally defined in terms of abstract IP operations or tasks, whereas others have first been defined in terms of (or have been motivated by) particular physical substrates. We are explicitly focusing on models of information processing that are physically realizable. We focus what can be done under real-world constraints like bounded time and speed, noise, limited precision, device mismatch, etc. Here, our emphasis on physical realizability of IPPs, though technologically motivated, is prone to shed light on - and be informed by - biological neural systems.

2 A review of (potentially) realizable IPPs

In this section we provide an overview of existing information processing paradigms which already have seen physical realizations or where at least there appear chances for them.

2.1 Stochastic Computing

Stochastic computing (SC) is a paradigm that uses random binary bit streams for computation [Han and Orshansky, 2013]. SC was first introduced in the 1960s for logic circuit design [Poppelbaum et al., 1967; Gaines et al., 1969], but its origin can be traced back to von Neumanns seminal work on probabilistic logic [Von Neumann, 1956]. A recent review on SC is given in Alaghi and Hayes [2013]. In SC, real numbers are represented by random binary bit streams that are usually implemented in series and in time. Information is carried on the statistics of the binary streams. For instance, the number 1/3 is represented by a bit stream with a percentage of 33 percent of randomly placed 1's. In this way, multiplication, for instance, can be done by merging two bit streams with an AND gate. From a neuroscience perspective, the question whether brain computations are inherently deterministic or stochastic is obviously of fundamental importance. Numerous experimental data highlight inherently stochastic aspects of neurons, synapses and networks of neurons on virtually all spatial and temporal scales that have been examined [Habenschuss et al., 2013]. A clearly visible stochastic feature of brain activity is the trial-to-trial variability of neuronal responses, which also appears on virtually every spatial and temporal scale that has been examined [Faisal et al., 2008]. This variability has often been interpreted as side-effect of an implementation of inherently deterministic computing paradigms with noisy elements, and it has been attempted to show that the observed noise can be eliminated through spatial or temporal averaging. However, more recent experimental methods, which make it possible to record simultaneously from many neurons (or from many voxels in fMRI), have shown that the underlying probability distributions of network states during spontaneous activity are highly structured and multimodal, with distinct modes that resemble those encountered during active processing. A theoretical foundation for this hypothesis has been provided in Habenschuss et al. [2013] by showing that even very detailed models for cortical microcircuits, with data-based diverse nonlinear neurons and synapses, have a stationary distribution of network states and trajectories of network states to which they converge exponentially fast from any initial state.

2.2 Approximate Computing

As opposed to stochastic / probabilistic computing, Approximate Computing (AC) is satisfied with low-precision versions of deterministic algorithms [Han and Or-

shansky, 2013]. The distinctive feature of AC is that it does not involve assumptions on the stochastic nature of any underlying processes implementing the system. It does, however, often utilize statistical properties of data and algorithms to trade quality for energy reduction. Approximate computing, hence, employs deterministic designs that produce imprecise results. Energy-minimization rationale behind approximate computing has been pointed out in Han and Orshansky [2013] through designing energy-efficient elementary transistor circuits for approximate adders and gates.

2.3 Natural Computing

Natural Computing (NC) is a paradigm introduced to encompass three classes of methods: 1) those that take inspiration from nature for the development of novel problem-solving techniques; 2) those that are based on the use of computers to synthesize natural phenomena; and 3) those that employ natural materials (e.g., molecules) to compute [Teuscher, 2014]. It is also known as non-von-Neumann computing, nonstandard computing, non-classical computing and unconventional computing and includes quantum computing, optical computing, molecular computing, and chemical computing with the goal to go beyond traditional computing technologies and paradigms. In a book entitled by "Physical Computation: A Mechanistic Account", Piccinini articulates this mechanistic account of concrete computation over sixteen chapters, weaving together ideas and evidence from several fields including philosophy of science, computer science, AI, biology, and cognitive neuroscience. According to Piccinini, a physical system is a computing system just in case (a) it is a mechanism, (b) it performs teleological functions, (c) at least one of its teleological functions consists in manipulating vehicles in accordance with a rule that is sensitive to differences between different portions of the vehicleswhere a vehicle is a physical variable whose values can enter a system, be changed by the system, and exit the system. Clause (c) is the distinguishing feature of concrete computing mechanisms, and allows one to see that computation does not require vehicles to be representations. Clause (b) emphasises that computing systems perform functions; in particular, they contribute to the goals of organisms. Clause (a) says that computing systems are concrete entities consisting of spatiotemporally organized and interacting components that produce phenomena [Piccinini, 2015]. In this book and the other review papers in this category, hardware bottlenecks (e.g, physical space, energy, precision, design, and manufacture and verification effort) are scarcely discussed.

de Castro [2007] provides an overview of the fundamentals of natural computing, emphasizing the biological motivation. In the introduction and part I of this book, De Castro [2006], the theme "from nature to computing and back again" is explored through the following guiding concepts: connectivity, stigmergy, adaptation (learning and evolution), feedback, self-organization, complexity, emergence and reductionism, buttom-up vs. top- down, determinism, chaos and fractals.

Then, with focus on evolutionary computing, computing inspired by nature is introduced. In part II, different mathematical/ modeling approach to simulate and emulate natural computing are discussed (e.g, differential equations, cellular automata, L-systems, fractional Brownian motion, iterated dynamical systems). And finally, at part III, computing with new natural materials (DNA computing and quantum computing) are broadly discussed. Kari and Rozenberg [2008] is another book on the same themes and a similar organization. MacLennan [2004] argued for a broadened defnition of computation, which includes continuous representations and processes, on the basis that computation is a matter of what is being accomplished (manipulation of abstract form independently of material substrate), rather than of how it is accomplished (digital or analog technology).

Natural Computing is more of a general attitude or perspective to be taken on computing, rather than a technical approach or formal theory. Many of themes of interest bundled in NC are technically and scientificially carried by their own dedicated research communities, independent from and outside of the "paradigm" of NC. In the following subsections we will present most of these well-cirumscribed approaches in more detail.

2.4 Self organizing IP systems

2.4.1 Cellular automata

A *cellular automaton* is a discrete model studied in computability theory, mathematics, physics, complexity science, theoretical biology and microstructure modeling. Cellular automata are also called cellular spaces or tessellation automata. Wolfram [1983, 1984] and Wolfram et al. [1986] lay foundations for studying self-organization and pattern formation in and with CA. Ermentrout and Edelstein-Keshet [1993] give a review on biologically motivated CA. Mitchell et al. [1996] review the large literature on computation in cellular automata, and the even larger literature on decentralized parallel computation in general.

Tzionas et al. [1997] suggests to use cellular automata in path planning of robots. Behring et al. [2001] gives an application example of CA in efficient computation of an optimal collision free path from an initial to a goal configuration on a physical space cluttered with obstacles. Stoy [2006] provides another example of an application of CA to control a distributed system of mechanically coupled modules connected in time-varying ways (self-reconfigurable robot). Looking into the possibility that life could emerge from the interaction of inanimate artificial molecules, Langton [1986] suggested that there is a strong possibility that the "molecular logic" of life can be embedded within cellular automata.

Using genetic algorithms (GAs) to evolve CAs to perform computations such as p=0.5 density classification has been investigated by Mitchell et al. [1994]. Fates and Morvan [2004] focus on the study of one-dimensional (1D) asynchronous CA with two states and nearest-neighbors. They describe a general-purpose scheme to quantify the robustness of a CA to asynchronism. Darabos et al. [2007] inves-

tigate the performances of collective task-solving capabilities and the robustness of complex networks of automata using the density and synchronization problems as typical cases (there is also an instructive comparison between robustness in random networks and scale-free small world networks).

Itoh and Chua [2009] develop a memristor cellular automaton and a memristor discrete-time cellular neural network which can perform some logical operations, image processing operations and complex behaviors. Adamatzky and Chua [2011] designed a minimalistic model of a regular network of memristors using CA. A hardware implementation of CA was provided in [Gibson et al., 2015]. In this study the potential for speeding up CA execution using multi-core CPUs and GPUs is investigated.

2.4.2 Reaction-diffusion systems

Reaction-diffusion systems involve constituents locally transformed into each other by chemical reactions and transported in space by diffusion. They arise, quite naturally, in chemistry and chemical engineering but also serve as a reference for the study of a wide range of phenomena encountered beyond the strict realm of chemical science such as environmental and life sciences [De Wit, 1999; Nicolis et al., 1977; Goldbeter, 1997; Marchese, 2002; Adamatzky and Wuensche, 2006; Bessler, 2005]. This paradigm is widely used in biological modeling to study the wave propagation in excitable tissues. Ventricular fibrillation in 2D or 3D models of cardiac tissue is an instructive example [Holden et al., 2013].

A special case of reaction-diffusion dynamics in electronic hardware is presented by Oya et al. [2007], which proposes a computational microchip of coupled single-electron devices which can self-organize into Voronoi boundary patterns.

2.4.3 Artificial chemistry

An *artificial chemistry* is a chemical-like system that consists of objects, called molecules, which interact according to rules resembling chemical reaction rules. Successful models of this kind would provide improved ways of describing ecological networks with feedback, co-evolution, and phenotypic plasticity.

Proposing the hypothesis that a distinguishing feature of adaptive systems is a self-referential loop between combinatorial objects and the functions they encode, Fontana [1991] developed ALChemy as a simple model to isolate this feature and to study the consequences. Such a loop was represented through a language. To address the question how groups emerge from local individual interactions, Dittrich et al. [2000] developed the *Seceder model* and investigated group formation through collision rules. Dittrich et al. [2001] give the first formal introduction to artificial chemistry. The authors argue that artificial chemistries are "the right stuff" for the study of pre-biotic and bio-chemical evolution, and they provide a productive framework for questions regarding the origin and evolution of organizations in general. They also address the application of artificial chemistry

in modeling, information processing and optimization. They introduced different kinds of artificial chemistry with systems ranging from those that are well-stirred and have no concept of location, to those in which molecules move in a very similar way to molecules in nature. Ono and Ikegami [2001] demonstrated a model of self-maintaining proto-cells through simulating the organization of membranes and their role in the interaction with metabolic reactions within the cell. Instead of modeling the membrane at an atomic or molecule level, they used an abstract model for the membrane.

Ziegler and Banzhaf [2001] demonstrated that an artificial chemistry could be used to control a small Khepera robot.

Hutton [2002] proposed a novel system of reactions in an artificial chemistry environment in which self-replicating molecules can spontaneously form under the right conditions. Interactions between the replicators are shown to result in mutated versions that can out-perform their parents. They also showed how such artificial chemistries can be implemented as a cellular automaton. Suzuki et al. [2002] give another example of modeling a system in an abstract chemistry framework to investigate the population dynamics of a tritrophic interaction mediated by herbivore-induced plant volatiles that attracted carnivorous natural enemies of herbivores.

Addressing the question of what features must be present in a system if it is to lead to indefinitely continuing evolutionary change, Hutton [2007] presented a unit of evolution: a self-reproducing cell in a two-dimensional artificial chemistry. The cells have a strip of genetic material that is used to produce enzymes, each catalysing a specific reaction that may affect the survival of the cell. The enzymes are kept inside the cell by a loop of membrane, thus ensuring that only the cell that produced them gets their benefit. A set of reaction rules, each simple and local, allows the cells to copy their genetic information and physically divide. Dittrich and Di Fenizio [2007] outline a theory to deal with systems consisting of many interacting components. They introduce the concept of a chemical organization as a closed and mass-maintaining set of components to map a complex (reaction) network to the set of organizations. Then, they try to connect dynamics with the set of organizations, which allows to map a movement of the system in state space to a movement in the set of organizations. Yamamoto et al. [2013] surveys the current state-of-the-art in the techniques for parallelizing artificial chemistries on GPUs, with focus on their stochastic simulation and their applications in the evolutionary computation domain.

2.4.4 Artificial immune systems / immune computing /computational immunology

Viewed as an information processing system, the natural immune system of organisms performs many complex tasks in a parallel and distributed computing fashion. These tasks include distinguishing between self and nonself, neutralization of nonself pathogens (viruses, bacteria, fungi, and parasites), learning, memory, associa-

tive retrieval, self-regulation, and fault-tolerance. De Castro and Timmis [2002b] give an introduction to artificial immune systems (AIS) as adaptive systems inspired by the biological immune system which are applied to problem solving. Dasgupta et al. [2003] survey major approaches in this field; e.g. artificial immune networks, aiNet, the Negative Selection Algorithms.

As an introduction, DasGupta [1993] illustrates different immunological mechanisms and their relation to information processing. From an information processing perspective, the immune system is a highly parallel system. It provides an excellent model of adaptive processes operating at the local level and of useful behavior emerging at the global level. Moreover, it uses learning, memory, and associative retrieval to solve recognition and classification tasks. To address the problems of protecting computers against viruses, Kephart et al. [1994] followed these ideas and designed an immune system for computers and computer networks. Like the vertebrate immune system, the system develops antibodies to previously unencountered computer viruses or worms and remembers them so as to recognize and respond to them more quickly in the future. They have proposed to use selective proliferation and self-replication (negative selection principle) for quick recognition and response. Later, Hunt and Cooke [1996] have applied the known rules of immune system to design a framework to solve simple pattern recognition tasks. The recognition of promoters in DNA sequences has been reported in this study. Since the system is self-organizing, it does not require effort to optimize system parameters. It seems to be the first immune system inspired unsupervised machine learning method.

The immune system exhibits many properties that we would like to incorporate into artificial computing systems: It is diverse, distributed, error tolerant, dynamic, self-monitoring (or self-aware) and adaptable. To do this, Hofmeyr and Forrest [2000] proposed an ARTIS (ARTificial Immune System) model in which the distributed environment is presented as a graph which provides the notion of locality in migration. Each node (detector) is modelled with a binary string and an r-contiguous bits rule is used as matching rule. The influence of the threshold in matching rule, locality in connection and learning mechanism then were investigated in computer security, in the form of a network intrusion detection system called LISYS.

Timmis et al. [2000] have proposed an immune system inspired supervised machine learning method as a simple classification tool. The modeling strategy is rule based in which the rules are motivated from concepts of immune systems. Here, each B cell object can represent a data item which is being used for learning and is capable of responding to a stimulus closely matching that data item. They have used affinity matching to evaluate how well B cell and the pathogen (training data item) match. The system is called AIRS (Artificial Immune Recognition System).

Bradley and Tyrrell [2000] introduce many of the ingenious methods provided by the immune system to provide reliable operation and suggests how such con-

cepts can inspire novel methods of providing fault tolerance in the design of state machine hardware systems. Through a process of self/non-self recognition, the proposed hardware immune system will learn to differentiate between acceptable and abnormal states and transitions within the "immunised" system.

De Castro and Von Zuben [2002] propose a computational implementation of the clonal selection principle that explicitly takes into account the affinity maturation of the immune response. The general algorithm, named CLONALG, is derived primarily to perform machine learning and pattern recognition tasks and is adapted to solve optimization problems, emphasizing multimodal and combinatorial optimization.

De Castro and Timmis [2002a] present the adaptation of an immune network model (an optimization version of aiNet), originally proposed to perform information compression and data clustering, to solve multimodal function optimization problems.

For computer security applications, a self-adaptive distributed agent-based defense immune system based on biological strategies has been developed by Harmer et al. [2002] within a hierarchical architecture. Robustness of the immune system inspired algorithms are mainly discussed in example papers of such applications [Harmer et al., 2002; Hofmeyr and Forrest, 2000].

de Castro and Timmis [2003] have proposed a general framework for designing AIS. The framework is conceptually simple: it is composed of a formal methodology to represent the components of the system, a set of functions that evaluate the quality of each of these components in a given environment, and a set of algorithms that govern the overall behavior of the system.

Watkins et al. [2004] present two revised versions of AIRS by change the memory cell evolution and somatic hypermutation. The heart of the previously proposed AIRS algorithm was the process of evolving memory cells from a population of artificial recognition balls (ARBs). The primary mechanism for providing evolutionary pressure to the population of ARBs in the development of memory cells is the competition for system wide resources. It is noted that these changes have not affected the overall classification accuracy of the system and have improved efficiency.

Coello and Cortés [2005] have proposed an algorithm to solve multiobjective optimization problems (either constrained or unconstrained) using the clonal selection principle.

A survey paper [Garrett, 2005] tracks the development of AIS since its inception, and then attempts to make an assessment of its usefulness, defined in terms of distinctiveness and effectiveness. It concludes that researchers in both negative (and positive) detection, and various types of immune network are beginning to produce algorithms that are hybrids of AIS with other methods. This is perhaps an indication that generalist versions of current AIS methods alone are not powerful enough for some tasks.

Dasgupta [2006] gives a survey on AIS algorithms and applications. Some of

the mentioned algorithms are building models mimicking the mechanisms in the biological immune system (BIS) to better understand its natural processes and simulate its dynamical behavior in the presence of antigens/pathogens. Most of the AIS models, however, emphasize designing artifacts – algorithms using simplified models of various immunological processes and functionalities. This survey found that negative selection algorithms are widely used compared to other AIS approaches. After presenting a simple example of each of the three main types of AIS algorithms (clonal selection, immune network and negative selection), Timmis et al. [2008] gives a theoretical analysis for each of these types. This is the first time we see statistical learning basics in AIS. An evolutionary artificial immune system for multi-objective optimization which combines the global search ability of evolutionary algorithms and immune learning of artificial immune systems has been proposed in Tan et al. [2008].

The Dendritic Cell Algorithm (DCA) is an example of an immune inspired algorithm developed using a multi-scale approach [Greensmith and Aickelin, 2009]. This algorithm is based on an abstract model of dendritic cells (DCs). The authors investigate the role of DCA in a model of immune system. Whitacre [2010] and Whitacre and Bender [2010] offer a new perspective on the mechanics of evolution and the origins of complexity, robustness, and evolvability. They explore the hypothesis that degeneracy, a partial overlap in the functioning of multi-functional components, plays a central role in the evolution and robustness of complex forms. Since then, the term degeneracy is increasingly used in recent papers of immune inspired algorithms.

Recently, chaos based strategies are embedded into AIS to alleviate its premature convergence problem [Jordehi, 2015] .

2.4.5 Artificial Life

Artificial life (ALife) studies the fundamental processes of living systems in artificial environments in order to gain a deeper understanding of the complex information processing that define such systems. The topics are broad, but often include evolutionary dynamics, emergent properties of collective systems, biomimicry, as well as related issues about the philosophy of the nature of life and the use of lifelike properties in artistic works. Artificial life owes its two deepest intellectual roots to John von Neumann and Norbert Wiener. Von Neumann designed the first artificial life model (without referring to it as such) when he created his famous self-reproducing, computation-universal cellular automata [von Neumann, 1966]. At about the same time, Wiener started applying information theory and the analysis of self-regulatory processes (homeostasis) to the study of living systems [Adami, 1998; Langton et al., 1989].

Artificial life research seeks to synthesize the characteristics of life by artificial means, particularly employing computer technology. The essays collected in Boden [1996] explore such foundational themes as the nature of life, the relation between life and mind, and the limits of technology. Langton [1997] is another collection

of such papers in the first three issues of the Artificial Life journal.

Bedau et al. [2000] provides a structured list of key open problems in artificial life. One of the fundamental open problems in artificial life is to explain how robust, multiple-level dynamical hierarchies emerge solely from the interactions of elements at the lowest level. This is closely analogous to the problem in cognitive science of explaining how cognitive capacities ultimately emerge from the interactions of non-cognitive elements like neurons.

Bedau [2003] highlights the state-of-the-art (at that time) in artificial life with respect to dynamical hierarchies, molecular self-organization, evolutionary robotics, the evolution of complexity and language, and other practical applications. It also speculates about future connections between artificial life and cognitive science. Self-organization and self-replication are important issues which are addressed in this paper mainly through examples of cellular automata.

Subrata and Zomaya [2003] compare several well-known artificial life techniques to gauge their suitability for solving location management problems. Due to their popularity and robustness, a genetic algorithm (GA), tabu search (TS), and ant colony algorithm (ACA) are used to solve the reporting cells planning problem. The main message of this paper is that the power of artificial life techniques stems from their capability in covering large search spaces which makes them suitable for applications such as location management in mobile computing environments. Subrata et al. [2007] applies the same artificial life techniques to gauge their suitability for solving grid load balancing problems.

ALife models are not restricted to theoretical investigations but often are realized and tested in hardware, often but not always robotic systems (collection volume: [Adamatzky and Komosinski, 2009]). Steels [1993] explains three aspects of behavior-oriented robotics: 1., adaptation and learning, 2., ability to operate autonomously in dynamically changing environment (different task from AI) and 3., being bio-inspired. Then, it addresses some open problems in this area of research: 1., relation between the mechanisms used in behavior-oriented AI and knowledge-oriented AI, 2., formalization and theory formation.

The paradigm of Artificial Life (ALife) strongly differs from traditional modeling, by studying not only "life-as-we-know-it", but also "life-as-it-could-be" [Vidal, 2008]. So, in Vidal [2008], the authors propose to extend this modeling technique to any process and not just to life, leading to the more general distinction of processes-as-we-know-them and processes-as-they-could-be. They differentiated the two kinds of modeling: real-world modeling and artificial-world modeling.

2.5 Artificial Neural Networks

An Artificial Neural Network (ANN) is an information processing paradigm that is inspired by the way biological nervous systems, such as the brain, process information. The key element of this paradigm is the novel structure of the information processing system [Kohonen, 1988]. It is composed of a large number of highly

interconnected processing elements (neurones) working in unison to solve specific problems. ANNs learn from examples. An ANN is configured for a specific application, such as pattern recognition or data classification, through a learning process. Learning in biological systems involves adjustments to the synaptic connections that exist between the neurones. Over decades of researches on theoretical and practical aspects of artificial neural network, numerous architectures and learning algorithms have been proposed and tested on a wide range of applications. In the following we take a look at two variants/aspects of ANNs which are of particular interest for novel hardware: reservoir computing and artificial networks of spiking neurons.

2.5.1 Reservoir computing

Inspired by the brain's ability to process information, Reservoir Computing (RC) provides a framework to design, train and analyze recurrent neural networks (RNNs) for processing time dependent information [Lukoševičius and Jaeger, 2009]. Reservoir computing is an umbrella term which subsumes liquid-state machines Maass et al. [2002], echo state networks [Jaeger, 2001], Evolino [Schmidhuber et al., 2007] and as a special learning rule, Backpropagation Decorrelation [Schiller and Steil, 2005] for RNNs. A reservoir computer is composed of three major parts. The "input layer" feeds the input signal into a random, large, fixed recurrent neural network that constitutes the "reservoir". The input signal is non-linearly mapped into a higher dimensional space through the internal variables of this dynamical system (i.e., reservoir states). In the "output layer", a linear combination of the reservoir states is computed as the time dependent output of the reservoir. Depending on the task, randomly generated output to reservoir (all-to-all) connections may also be included in the architecture which are known as output feedback weights. Different from traditional RNN training strategies, the RC technique proposes to only adapt the output weights to minimize the mean square error between the target and the output signal. The values of input weights and reservoir coupling weights are not critical and can be selected a random within some predefined intervals to obtain the best performance. Regarding the fact that only the output connections are trained and the optimization of the output layer only consists in a linear regression, training algorithms are computationally efficient and straightforward.

The nonlinear expansion of the input signal into a high-dimensional (reservoir) signal space, plus ease of training enable reservoir computers to efficiently accomplish a large range of complex tasks on time dependent signals. Nonlinear channel equalization [Jaeger and Haas, 2004; Boccato et al., 2011], time series prediction [Jaeger, 2001; Li et al., 2012; Lukoševičius and Jaeger, 2009] , speech recognition [Triefenbach et al., 2010] and robot control [Antonelo et al., 2008b,a] are some of examples in which reservoir computing has been reported to performed well.

Computational efficiency, simplicity and flexibility of reservoir computing have led it to be one of the main candidate paradigms for analog information processing

[Duport et al., 2016; Appeltant et al., 2011a; Martinenghi et al., 2012; Larger et al., 2012; Hicke et al., 2013; Schrauwen et al., 2007].

2.5.2 Spike based neural computing

To address the question of "how can models of spiking neurons carry out computations?", a review of spiking neuron models, neural network with spiking neurons, learning in these networks and the challenges is presented in this section. Computing with spikes is a promising avenue for novel computing technologies at least for three reasons:

- brains do it – a superb proof of principle;
- spike-based neuromorphic microchips promise one or two orders of magnitude less energy consumption than traditional digital microchips;
- addressed spike event signals can be communicated in a nearly all-to-all manner on a microchip.

Mathematical models of spiking neurons. Ever since the pioneering work of Hodgkin and Huxley [1952], biological neuron models have consisted of ODEs representing the evolution of the transmembrane voltage and the dynamics of ionic conductances. It is only recently that discrete-time dynamical systems (iterated maps) have begun to receive attention as valid phenomenological neuron models [Ibarz et al., 2011]. An alternative to these dynamical systems models are so called Spike Response Models (SRMs) [Gerstner, 2001b; Grüning and Bohte, 2014; Maass et al., 2002]. In SRMs, the membrane potential is not computed through differential equations but as a sum of integral kernels. In principle, SRMs are equivalent to many classes of dynamical systems models, but offer a different insight into the processes in spiking neurons. In this section, these three categories of models are introduced.

Continuous dynamical system models. The dominants in this category are the so-called Hodgkin-Huxley models. However, from a computational perspective, the simulation of thousands of detailed Hodgkin-Huxley neurons calls for a supercomputer. Therefore, further simplification may be called for, which is what the integrate-and-fire family of models, and particularly its most popular member, the leaky integrate-and-fire (LIF) model, provide [Izhikevich, 2004]. This model is still conductance-based, but it dispenses with all variables except membrane voltage. In fact, the cell membrane behaves precisely as a capacitor that passively discharges through a resistor, representing the ionic channels, and may be charged by external currents (synaptic events, for example). The key to the neuroscientific interest of the model is its threshold-and-reset mechanism: if voltage reaches a threshold value, the capacitor instantaneously discharges to a reset level and

the neuron is said to have fired a spike [Ibarz et al., 2011]. The model can be significantly improved replacing the discharge resistor by a nonlinear component. The advantage of nonlinear LIF models over their linear counterparts is tightly linked to the saddlenode bifurcation that allows transition from silence to spiking and vice versa [Ibarz et al., 2011]. The exponential integrate-and-fire (EIF) model [Brette and Gerstner, 2005] and quadratic-integrate-and-fire model [Izhikevich, 2004] are examples.

Another examples of so-called Hodgkin-Huxley-ODE formulation for neuronal dynamics are FitzHugh-Nagumo models, Izhikevich models of spiking and bursting neurons [Izhikevich et al., 2003] and three-dimensional Hindmarch-Rose models [Hindmarsh and Rose, 1984; Izhikevich, 2000]. Reviews of such models from a dynamical system viewpoint are reported in Touboul [2008]; Touboul and Brette [2009].

Discrete dynamical system models We can trace the origin of most map-based neuron models to two main sources [Ibarz et al., 2011]:

- Event discretization. The action potential, a paradigmatic all-or-nothing event with a duration that is very short compared to the time scale of other neuronal processes, is a natural source of discretization. In some models, such as those of the family that begins with the classical McCullochPitts, going through Nagumo-Sato [Nagumo and Sato, 1972] and Aihara [Aihara et al., 1990], the only information considered relevant is whether a spike is present or absent during a certain time interval. In some instances [Medvedev, 2005], a map is derived directly by means of a Poincaré section from a continuous-time flow, the section being taken at spike initiation. Finally, the threshold-and-reset mechanism of integrate-and-fire models is a form of discretization: the flow of the variables during a spike is ignored, and the state of the system jumps from its pre-spike to its post-spike value. Incorporating this jump in a map-based formulation is straightforward.

- Discretization of ODE-based models through numerical integration. The discretized version of continuous (ODE) Izhikevich model is a handy example [Ibarz et al., 2011]. In the case of neuron models, however, the desire to use time steps (in numerical integration) as big as possible in order to simulate networks of thousands of neurons has led modelers to establish discrete equations on their own as their preferred neuron model. Besides, the map-based formulation has led to variants that could hardly have arisen from integration of an ODE; the chaotic Rulkov model is a case in point [Rulkov, 2002; Shilnikov and Rulkov, 2004; Rulkov, 2001]. The Courbage-Nekorkin-Vdovin (CNV) model [Courbage et al., 2007] and the Chialvo model [Chialvo, 1995] are another examples of inherently discrete designs.

As a step further in abstraction, small discrete subpopulations of neurons can

be used as basic representational units (called *netlets* in Harth et al. [1970]). Ibarz et al. [2011] gives a brief review of progressions of such abstracted models.

Stochastic models. In this view, a neuron (understood to include its incoming synapses) acts mathematically as filter or operator in that it maps incoming sets of spike trains into an output spike train. As mentioned before, they are known as Spike Response Models (SRMs). Ma et al. [2006]; Florian [2007]; Pecevski et al. [2011]; Buesing et al. [2011]; Gerstner [2001a]; Gerstner et al. [1993] have used this kind of models in their studies on neural networks.

Neural codes. The aim of artificial spiking neural networks is to carry out neural computation. However, firstly, the quantities relevant to the computation have to be expressed in terms of the spikes that spiking neurons communicate with. The earliest attempt seems to be Bialek et al. [1991] where a real-time stimulus signal has been reconstructed from the spike trains recorded from a single neuron by using a linear filter.

Although the nature of the neural code (or neural codes) is an unresolved topic of research in neuroscience, based on what is known from biology, a number of neural information encodings have been proposed [Grüning and Bohte, 2014]:

- **Binary coding**

  Binary coding is an all or nothing encoding: a neuron is either active in a given time interval, that is it fires one or several spikes in that time interval, or it is silent. Following a binary interpretation of the output spike trains, Gütig and Sompolinsky [2006] proposed a supervised synaptic learning to reveal the high capacity of neural systems to learn to decode information embedded in distributed patterns of spike synchrony. Later, Urbanczik and Senn [2009a] followed the idea and proposed a learning rule based on the gradient of a cost function to do a binary classification task.

- **Rate coding**

  Rate coding is another abstraction from the timed nature of spikes, in that only the rate of spikes in a interval is used as a measure for the information conveyed. Rate encoding is motivated by the observation that physiological neurons tend to fire more often for stronger (sensory or artificial) stimuli. It can be applied at the level of the single neuron, or at the interpretation of spike trains [Grüning and Bohte, 2014]. In the first case neurons are modeled directly as rate neurons which transfer at each time step real-valued input numbers – "rates" – into an output rate. Rate coding has been the notion behind standard artificial "sigmoidal neurons in technical contexts and cognitive science. For the interpretation of spike trains, a rate encoding (also called frequency encoding) has also been used to interpret the outputs of spiking neural networks, see for example Wade et al. [2010]; Urbanczik and Senn [2009b]; Glackin et al. [2011].

- **Temporal coding (and inter-spike interval coding)**

  In a fully temporal code, the encoding relies on the precise timing of all spikes. They are inspired by evidence in neuroscience that spike-timing can be remarkably precise and reproducible [Grüning and Bohte, 2014; Gerstner et al., 1996; Panzeri et al., 2001]. In a fully temporal code, timings are relevant to a certain given (internal or external) event (such as the onset of a stimulus or spike of a reference neuron). For example, a spike-train with three spikes at 10ms, 17ms and 21ms (relative to e.g. stimulus onset) should have a different meaning than a spike-train with three spikes at 8ms, 17ms and 27ms or a spike train with five spikes at 8ms, 10ms, 17ms, 21ms and 27ms etc [Grüning and Bohte, 2014]. If spike-trains with fixed mutual timings are distributed across a group of neurons, these patterns are referred to as a polychronous group [Izhikevich, 2006]. In their temporal coding framework, Rao and Sejnowski [2001] have proposed a temporal difference learning rule for prediction of input sequences. Moreover, a mechanism by which a temporal code can be generated through an interaction between an asymmetric rate code and oscillatory inhibition has been proposed in Mehta et al. [2002].

- **Latency coding (e.g., rank order coding)**

  Latency coding makes use of the timing of spikes, but not of the multiplicity of spikes. Information is encoded as the latency from a certain (internal or external) event until the first spike. This is motivated by the observation that important sensory stimuli tend to elicit a spike earlier in upstream neurons [Grüning and Bohte, 2014; Florian, 2012]. This encoding has for example been used in unsupervised learning [Natschläger and Ruf, 2009] and supervised learning methods like SpikeProp [Bohte et al., 2002] or the Chronotron [Florian, 2012] and others [Urbanczik and Senn, 2009b]. Closely related is rank-order coding, where information about a stimulus is encoded in the order in which neurons within a group emit their first spikes [Thorpe and Gautrais, 1998; Thorpe et al., 2001; Perrinet et al., 2001].

  A comparison of rate coding, mean inter-spike interval and rank order code, where the first ganglion cells to emit a spike are given a maximal weight has been presented in Van Rullen and Thorpe [2001] to do an image reconstruction (retina) task. Another example was provided in Brette [2015].

- **Predictive coding**

  Predictive spike-coding is a special case of a fully temporal encoding. Here, the notion is that the spiking mechanism effectively provides a mean to carry out an analog-to-digital and digital-to-analog conversion at the soma and the synapses of the neuron, respectively. Examples include greedy solutions [Lazar and Tóth, 2003; Rombouts and Bohte, 2010; Delbruck, 2008; Boerlin and Denève, 2011; Hosoya et al., 2005] where spike-generation at

the soma is associated with the subtraction of a temporal kernel from in the incoming current; at the synapse each spike then causes an excitatory postsynaptic potential (EPSP) in the target neuron such that the sum of EPSPs approximates the computed signal at the soma [Grüning and Bohte, 2014]. Adaptation mechanisms can change the EPSP, enabling the spiking neuron to adjust to changes in the dynamic range of the inputs [Bohte, 2012]. The adaptable dynamics of spiking neurons enables them to compute over multiple timescales [Bohte et al., 2002; Rombouts and Bohte, 2010].

Predictive spike-code was also applied to compute an implicit measure of the inference task in Boerlin and Denève [2011]. In this work, the authors proposed that probability distributions are inferred spike-per-spike in recurrently connected networks of integrate-and-fire neurons (i.e. deterministic neurons).

- **Probabilistic coding**

  Probabilistic spike-coding is concerned with efficiently carrying out inference using spiking neurons, and is mostly considered in a computational neuroscience context. There are two prominent flavours of spiking neural computation here: one where spiking neurons are considered that stochastically generate spikes in response to a given input [Beck et al., 2012; Ma et al., 2006], and one where spiking neurons are essentially deterministic. In both cases, an implicit measure of the inference task is computed, the first in a rate code and the latter in a predictive spike-code [Grüning and Bohte, 2014]. As an example, Beck et al. [2012] have introduced a very general approximate inference algorithm known as Variational Bayesian Expectation Maximization which can be naturally implemented within the linear probabilistic population coding (PPC) framework.

In parallel with those studies which propose novel neural information encodings based on experimental evidence, there are other mostly experimental studies which insist on the concurrent use of several codes that combine information across different spatiotemporal scales to explain processing in sensory cortices [Kayser et al., 2009; Gerstner, 2001a].

Computational power of a single spiking neuron. To answer the question of "what can a single spiking neuron computationally do?", all the papers which have been addressed above should be considered. Moreover, Rombouts and Bohte [2010] consider the actual neural spike-train as the fractional derivative, provided that the neural signal is approximated by a sum of power-law kernels (called power-law signal encoding). In addition to this line of arguments, there are other studies which have tried to teach a single spiking (or bursting) neuron to do some computational tasks [Maass, 2003; Kepecs and Lisman, 2003; Siettos and Starke, 2016; Maass, 1997]. An example is the tempotron model introduced in Gütig and

Sompolinsky [2006] where a supervised synaptic learning for a spiking neuron has been developed. In a follow-up paper, Urbanczik and Senn [2009a] have proposed a learning rule based on the gradient of a cost function for tempotron neuron model to implement a binary classification of input spike trains by an integrate-and-fire neuron. In addition, two supervised learning rules for spiking neurons with temporal coding of information (chronotrons) have been proposed in Florian [2012]; one that provides high memory capacity (E-learning), and one that has a higher biological plausibility (I-learning).
Regarding the fact that both synapse and dendrite are known to actively contribute to neural computation, the spiking neuron models would have to be complemented by more detailed models of synapses and dendrites [Grüning and Bohte, 2014; Major et al., 2013; Thalmeier et al., 2015; Letzkus et al., 2006; Markram et al., 1997]. In this regard, several studies have combined different types of synaptic plasticity with linear or non-linear (and passive or active) dendritic functions to propose new computational algorithms. Letzkus et al. [2006]; Kampa et al. [2007]; Ramakrishnan et al. [2013]; Roy et al. [2014, 2015]; Hussain et al. [2014] provide some examples. The results of Letzkus et al. [2006] suggest that synapse location within the dendritic tree is a crucial determinant of STDP, and that synapses undergo plasticity according to local rather than global learning rules.

Networks of spiking neurons. Networks of spiking neurons are studied in two main categories: 1., pattern formation, synchronization, chimera states and other "statistical physics" phenomena in networks of spiking neurons and 2., learning and memory in spiking neural networks. As the first group is beyond the scope of our study, we mention only a few examples: Brunel [2000]; Vogels and Abbott [2009]; Brody and Hopfield [2003], and focus our treatment on the second category.

Neural codes define the relationship between information and spiking patterns. The challenge remains to carry out useful computations with spiking neural networks. This requires learning methods to adapt weights in the network, and inseparable from adaptation mechanisms, also methods for designing network architectures (overviews: Grüning and Bohte [2014]; Abbott et al. [2016]; monograph: Gerstner et al. [2014]; Indiveri and Liu [2015] survey (mostly) spiking neural architectures with special regards to hardware implementations; Sheik et al. [2013] do a similarly oriented survey with a focus on robotics applications).

Plasticity. In the 1990s an increasing number of studies showed that real neurons were able to produce precisely timed spikes and a synaptic learning rule sensitive to the precise relative spike-timing was found, spike-timing-dependent plasticity (STDP) [Grüning and Bohte, 2014]. Since then, numerous variations of STDP have been proposed. For instance, Kistler and Van Hemmen [2000] proposed a synaptic plasticity mechanism through which given an input statistics, the stationary synaptic weights that result from the temporal correlations between the pre- and postsynaptic spikes are computed. Van Rossum et al. [2000] proposed another

version in which synaptic weights change in proportion to how correlated they are with other inputs onto the same postsynaptic neuron. Later, in an experimental study, Sjöström et al. [2001] demonstrated a form of cooperativity operating even when postsynaptic firing is evoked by current injection, and revealed a complex dependence of LTP and LTD on rate and timing. In order to provide a general framework for regulating neurotransmitter release probability, Senn et al. [2001] proposed an algorithm which reproduces the experimental data by modifying the probability of vesicle discharge as a function of the relative timing of spikes in the pre- and postsynaptic neurons. Using a biophysical model of a cortical neuron, Rao and Sejnowski [2001] proposed to implement a plasticity mechanism as a form of temporal difference learning for the prediction of input sequences. They showed that a temporal difference rule used in conjunction with dendritic backpropagating action potentials reproduces the temporally asymmetric window of Hebbian plasticity observed physiologically. Morrison et al. [2007] have introduced a weight dependent STDP rule; a multiplicative dependence on the synaptic weight for depression, and a power law dependence for potentiation. It has been also shown that the proposed plasticity model is robust against different spike pairing schemes. Florian [2007] presented an example of reward modulated STDP with eligibility trace; in this work, the neuron model is a stochastic spike response model of spiking neurons and learning is possible if the reward signal is delayed. The authors have evaluated the algorithm on the XOR problem with both rate coded and temporally coded input. Clopath et al. [2010] derived a phenomenological model of plasticity rule which also depends on the post- synaptic voltage. The plasticity rule behaves similar to a STDP rule in which triplets of spikes with pre-post-post or postpre-post timing evoke LTP. Bidirectional connections and highly connected clusters are possible (and dominant under a rate-coding procedure). Mitra et al. [2009] describe a general adaptation algorithms for spiking neurons which comprises STDP as a special case but also can "emulate" rate-based coding schemes, which in turn makes it possible to use this learning rule for supervised pattern learning tasks; and use this model to implement supervised learning on spiking VLSI microchips.

Zenke et al. [2013] have claimed that Hebbian learning has to be opposed by some kind of compensatory mechanism to preserve network stability. They, therefore, provided a brief review on homeostasis mechanism and then proposed a new plasticity rule to modulate the rate of LTD induction using the temporal average of the postsynaptic firing rates over a given time window.

Frémaux et al. [2013] provide a model of reward-modulated spike-timing-dependent plasticity based on extending the continuous temporal difference (TD) learning of Doya [2000].

To stabilize assembly formation and memory recall in a spiking recurrent network, Zenke et al. [2015] suggested that multiple plasticity mechanisms on different timescales are needed. They proposed a version of STDP at a timescale of several hundred milliseconds in a combination with triplet STDP with heterosynaptic and

transmitter-induced plasticity at a timescale of seconds (slow homeostatic regularization).

There exist numerous other examples which have proposed rewarded, homeostatic or other modifications of STDP [Seung, 2003; Chechik, 2003; Toyoizumi et al., 2004; Legenstein et al., 2005; Bohte and Mozer, 2007; Vasilaki et al., 2009; Lazar et al., 2009; Miner and Triesch, 2016; Babadi and Abbott, 2016; Turrigiano, 2012].

Feedforward networks of spiking neurons. Here we will give some examples of papers which have developed either supervised or unsupervised learning algorithms for feedforward networks of spiking neurons. The important point here is that most of them are trying to only do classification.

Bohte et al. [2002] proposed a supervised learning rule, SpikeProp, akin to traditional error backpropagation, in which latency coding (delayed synapses) are used. The results show that in comparison to rate coding, it needs fewer neurons. Another supervised synaptic weight association training (SWAT) algorithm has been proposed by Wade et al. [2010] for a multiple-in, multiple-out network topology with (only) a single trained neuron. Using rate coding, they have merged the Bienenstock-Cooper-Munro (BCM) learning rule with STDP. Glackin et al. [2011] also developed a supervised training algorithm that implements fuzzy reasoning in a spiking (rate-coded) neural network. Sporea and Grüning [2013] and Stromatias and Marsland [2015] are recent examples of supervised learning algorithms in feed-forward architecture.

A robust unsupervised clustering algorithm has been proposed in Natschläger and Ruf [2009] to compute radial basis functions (RBFs) by storing the information in the delay of spiking neurons in receiving temporally encoded inputs.

A reinforcement learning rule for networks of spiking neurons has been proposed by Xie and Seung [2004] in which the learning rule performs stochastic gradient ascent on the expected reward by correlating the fluctuations in irregular spiking with a reward signal. They showed that irregular spiking similar to that observed in biological neurons could be used as the basis for a learning rule that calculates a stochastic approximation to the gradient. The learning rule is derived based on a special class of model networks in which neurons fire spike trains with Poisson statistics [Grüning and Bohte, 2014].

Technically, many learning algorithms use the concept of an eligibility trace that captures in what way a given (local) synapse contributes to the (global) network output. Typically a (suggested) weight change computed as in STDP is taken as an eligibility trace. The eligibility trace is then converted into a real weight change depending on a global error or reward signal. This is the concept of reward modulation (or reward maximisation) and such approaches can be found in a number of algorithms [Farries and Fairhall, 2007; Legenstein et al., 2008; Florian, 2007; Grüning and Sporea, 2012]. Urbanczik and Senn [2009b] proposed to add a feedback about the population response to modulate synaptic plasticity

in addition to global reinforcement.

Recurrent networks with spiking neurons. An abstract winner-take-all (WTA) mechanism has been proposed by Coultrip et al. [1992]. They simulated a circuitry consisting of several excitatory neurons jointly innervating and receiving feedback from a common inhibitory interneuron. Excitatory cells in this simulation receive input during a cycle of naturally occurring rhythmic activity (the hippocampal theta rhythm), and the neuron receiving the strongest input activation is the first to reach its spiking threshold. Spiking excites the inhibitory cell, which in turn prevents other cells from responding. The result is the natural generation of a simple competitive or "winner-take-all" mechanism, allowing only the most strongly activated cell in a group or "patch" to respond with spiking activity.

To speed up computations in neural networks with recurrent connections, Jin and Seung [2002] studied a counterexample, a network consisting of integrate-and-fire neurons with self-excitation, all-to-all inhibition, instantaneous synaptic coupling, and constant external driving inputs. When inhibition and/or excitation is strong enough, the network performs a winner-take-all computation for all possible external inputs and initial states of the network. The computation is very fast: as soon as the winner spikes once, the computation is completed since no other neurons will spike. Tiňo and Mills [2006] extend the existing gradient-based algorithm for training feedforward spiking neuron networks, SpikeProp [Bohte et al., 2002], to recurrent network topologies, so that temporal dependencies in the input stream are taken into account. They showed that it is often possible to extract from trained recurrent spiking neural networks a discrete Moore machine by grouping together similar spike trains appearing in the recurrent layer.

Oster et al. [2009] analytically examined the ability of a spike-based WTA network to discriminate the statistics of inputs ranging from stationary regular to nonstationary Poisson events by using a simplified Markov model of the spiking network. Their work extends previous theoretical results showing that a WTA recurrent network receiving regular spike inputs can select the correct winner within one interspike interval. Using unsupervised STDP-based learning and lateral inhibition, Masquelier et al. [2009] provide a distributed coding scheme to do spatiotemporal pattern recognition. In this framework, each neuron learns one subpattern.

Stability and convergence of a WTA network has been studied in Rutishauser et al. [2011]. They use nonlinear contraction theory to establish conditions for stability in the fully nonlinear case. Rutishauser et al. [2012] is another example of competition in spiking networks through selective inhibitory synchrony.

Kasabov et al. [2013] applied a combination of rank order coding and STDP to do on-line pattern recognition. They claim that their proposed system is capable to be trained in supervised, unsupervised and semi-supervised manner. Brea et al. [2013] considered a recurrent network of stochastic spiking neurons composed of both visible and hidden neurons. They derived a generic learning rule that is

matched to the neural dynamics by minimizing an upper bound on the Kullback-Leibler divergence from the target distribution to the model distribution. They claim that their derived learning rule is consistent with spike-timing dependent plasticity in that a presynaptic spike preceding a post-synaptic spike elicits potentiation while otherwise depression emerges. Recently, DePasquale et al. [2016] presented a procedure for training recurrently connected spiking networks to generate dynamical patterns autonomously, to produce complex temporal outputs based on integrating network input, and to model physiological data. The procedure makes use of a continuous-variable network to identify targets for training the inputs to the spiking model neurons. Corneil et al. [2014] have presented a STDP-based framework extending a previous model [Nessler et al., 2013] which can simultaneously learn to abstract hidden states from sensory inputs and code transition probabilities between these states in recurrent connection weights [Kappel et al., 2014].

Chicca et al. [2014] present an overview of a diversity of analog VLSI spiking neuronal circuits and networks and argue that while many useful models of spiking electronic circuits have already been found, in order to reach fully autonomous neuromorphic intelligent systems significant further interdisciplinary research is needed.

Spiking deep belief networks. A potential solution to overcome the energy demands, communication overhead, and high response latencies of deep belief networks (DBNs) is to transform them into spiking neural networks. Accordingly there is an increasing number of papers which tries to spikify deep belief network [Matsugu et al., 2002; Esser et al., 2016b; Stromatias et al., 2015a; OConnor et al., 2015; Lee et al., 2016; Diehl et al., 2015; Cao et al., 2015].

Sampling and probabilistic inference. In order to enable Bayesian inference in a spiking neural network, Ma et al. [2006] proposed a new type of code which they call "probabilistic population codes" in a network of spiking neurons that stochastically generate spikes in response to a given input.

Boerlin and Denève [2011] propose that probability distributions are inferred spike-per-spike in recurrently connected networks of integrate-and-fire neurons. They used deterministic neurons and an implicit measure of the inference task which is computed in a predictive spike-code.

Pecevski et al. [2011] have shown how networks of spiking neurons carry out probabilistic inference through sampling in general graphical models. Using synaptic or dendritic structures, they have presented a neuronal realization for Markov chain model.

Buesing et al. [2011] showed that under some conditions the stochastic firing activity of networks of spiking neurons can be interpreted as probabilistic inference via Markov chain Monte Carlo (MCMC) sampling.

It has been suggested that soft winner-take-all (WTA) circuits, where pyrami-

dal neurons inhibit each other via interneurons, are a common motif of cortical microcircuits [Nessler et al., 2013]. Through theoretical analysis and computer simulations, Nessler et al. [2013] showed that Bayesian inference is enabled in these network motifs through STDP in combination with activity-dependent changes in the excitability of neurons. The fundamental components of this emergent Bayesian computation are priors that result from adaptation of neuronal excitability and implicit generative models for hidden causes that are created in the synaptic weights through STDP. In a similar study, Kappel et al. [2014] have shown that STDP in combination with lateral excitation in winner-take-all networks can approximate Hidden Markov model learning. In the same way, Bill et al. [2015] have proposed that the spiking dynamics of a recurrent network with lateral excitation and local inhibition in response to distributed spiking input can be understood as sampling from a variational posterior distribution of a well-defined implicit probabilistic model. Kappel et al. [2015] introduced stochastic aspects of synaptic plasticity and proposed a model to express how priors on weight distributions and connection probabilities can be merged optimally with learned experience.

Recently, Pecevski and Maass [2016] used a combination of STDP and an intrinsic plasticity (based on lateral inhibition) to realize probabilistic associations between neurons in arranged in modules. Steimer and Douglas [2013] is another example.

A source of randomness is at the core of many stochastic optimization algorithms, for instance simulated annealing or genetic algorithms. Generating good random numbers on digital circuits incurs a non-negligable computational cost. Mostafa et al. [2015] describe a VLSI implementation for a specially designed stochastic optimization scheme where the source of randomness roots in the incommensurable frequencies of physically realized oscillators, and demonstrate state-of-the-art performance on classical NP-complete optimization problems.

2.6 Chaos based IP

Chaos computing is the idea of using chaotic systems for computation. In particular, chaotic systems can be made to produce all types of logic gates and further allow them to be morphed into each other. An instructive example is Munakata et al. [2002] in which the logical AND, OR, NOT, XOR, and NAND operations (gates) and bit-by-bit arithmetic operations have been implemented by chaotic elements. In addition to this line of research, there are two other lines which either seek to characterize the condition for occurrence of chaos in a mathematical models such as cellular automata, lattices of coupled dynamical systems and recurrent neural networks or try to find chaos control strategies.

As an examples of the first category, Langton [1990] presents research on cellular automata which suggests that the optimal conditions for the support of information transmission, storage, and modification, are achieved in the vicinity of a phase transition. Later, it has been also influential for the “edge of chaos”

myth in reservoir computing. In another example, the capacity of a lattice of threshold coupled chaotic maps to perform computations (e.g., logic gates, encode numbers, and perform specific arithmetic operations, such as addition and multiplication) has been investigated [Sinha and Ditto, 1999]. In Chang et al. [1998] the effect of Gaussian noise on stabilization of chaotic dynamical systems is explored (Freeman's model of the olfactory system [Freeman, 1987]). In Bertschinger and Natschläger [2004], a study on reservoir networks has been done which shows that certain dynamical systems which are capable of doing complex computational tasks should operate near the edge of chaos, i.e. the transition from ordered to chaotic dynamics. Chen and Aihara [1995] introduced the Transient Chaotic Neural Network (TCNN) in which the transiently chaotic dynamics is utilized for global searching and self-organizing. In Kazem et al. [2013] a forecasting model based on chaotic mapping, firefly algorithm, and support vector regression (SVR) has been proposed to predict stock market price. A population-based optimization algorithm, Chaotic Evolution (CE), which uses ergodic properties of chaos has been developed in Pei [2014] to implement exploration and exploitation functions of an evolutionary algorithm.

Fradkov and Evans [2005] provide a survey of chaos control approaches. Using perturbation, feedback loops, neural networks, delay feedbacks, predictive control, fuzzy adaptive sliding mode control are among the proposed strategies [Ott et al., 1990; Pyragas, 1992; Shinbrot et al., 1993; Alsing et al., 1994; Polyak, 2005; Layeghi et al., 2008].

Chaos computing has recently received an increasing attention in hardware design studies. Ditto et al. [2008] exploit the controlled richness of nonlinear dynamics to obtain flexibly reconfigurable hardware. Muthuswamy [2010] constructed a physical chaotic circuit from four fundamental circuit elements the resistor, capacitor, inductor and the memristor. Iu et al. [2011] provided a systematic study on chaotic behavior in memristor circuits. In a tutorial reported in Buscarino et al. [2012], Chua has inserted memristors in a Hodgkin-Huxley model of excitable cells. This tutorial shows that synapses are locally-passive memristors, and that neurons are made of locally-active memristors. Buscarino et al. [2012] introduces a chaotic circuit based on the mathematical realistic model of the HP memristor. Zhang and Shen [2013] is another example in this line.

Adamatzky and Chen [2013] gives a collection of tributes to outstanding discoveries pioneered by Leon Chua in nonlinear circuits, cellular neural networks, and chaos.

2.7 Population based IP systems

Algorithms inspired from the collective behavior of species such as ants, bees, wasps, termite, fish, and birds are referred as swarm intelligence algorithms Talbi [2009]. Swarm intelligence originated from the social behavior of those species that compete for foods. The main characteristics of swarm-intelligence-based al-

gorithms are particles are simple and nonsophisticated agents, they cooperate by an indirect communication medium, and do movements in the decision space [Talbi, 2009]. Among the most successful swarm intelligence inspired optimization algorithms are ant colony and particle swarm optimization.The information processing strategy of swarm intelligence methods is to allow cells to stochastically and collectively swarm toward optima. This is achieved through a series of three processes on a population of simulated cells: 1) "Chemotaxis" where the cost of cells is derated by the proximity to other cells and cells move along the manipulated cost surface one at a time (the majority of the work of the algorithm), 2) "Reproduction" where only those cells that performed well over their lifetime may contribute to the next generation, and 3) "Elimination-dispersal" where cells are discarded and new random samples are inserted with a low probability. Early work by Liu and Passino [2002] considered models of chemotaxis as optimization for both E.coli and M.xanthus which were applied to continuous function optimization. This work was consolidated by Passino who presented the bacterial foraging optimization algorithm that included a detailed presentation of the algorithm, heuristics for configuration, and demonstration applications and behavior dynamics [Muller et al., 2002]. Other examples of such algorithms are Beni and Wang [1993]; Timmis et al. [2010]; Das et al. [2009]. Sayama [2009] developed a *swarm chemistry* framework that uses artificial swarm populations as chemical reactants. Different from common methods in artificial chemistry, reaction in swarm chemistry is spontaneously achieved by the emergence of a new spatiotemporal pattern of collective behavior through the kinetic interaction between multiple chemical species.

The main interest of real ants behavior is that simple ants using collective behavior perform complex tasks such as transportation of food and finding shortest paths to the food sources. ACO algorithms mimic the principle that using very simple communication mechanism, an ant colony is able to find the shortest path between two points [Talbi, 2009]. The basic idea in ant colony optimization algorithms (ACO) is to imitate the cooperative behavior of real ants to solve optimization problems. ACO metaheuristics have been proposed by M. Dorigo Dorigo et al. [2006]. They can be seen as multiagent systems in which each single agent is inspired by the behavior of a real ant. Dorigo et al. [2006]; Dorigo and Blum [2005]; Dorigo and Stützle [2009]; Bilchev and Parmee [1995] are highly cited papers in this area of research. Ant colony optimization algorithms have been applied to many combinatorial optimization problems, ranging from quadratic assignment to protein folding or routing vehicles and a lot of derived methods have been adapted to dynamic problems in real variables, stochastic problems, multi-targets and parallel implementations. It has also been used to produce near-optimal solutions to the travelling salesman problem. They have an advantage over simulated annealing and genetic algorithm approaches of similar problems when the graph may change dynamically; the ant colony algorithm can be run continuously and adapt to changes in real time. This is of interest in network routing and urban

transportation systems [Talbi, 2009].

2.8 Membrane computing

Membrane computing investigates computing models abstracted from the compartmentalized structure of living cells effected by membranes [Paun, 2012]. A generic membrane system (P-system) consists of cell-like compartments (regions) delimited by membranes which are placed in a nested hierarchical structure. Each membrane-enveloped region contains objects, transformation rules which modify these objects, as well as transfer rules, which specify whether the objects will be transferred outside or stay inside the region. Regions communicate with each other via the transfer of objects. The computation by a membrane system starts with an initial configuration, where the number (multiplicity) of each object is set to some value for each region (multiset of objects). It proceeds by choosing, nondeterministically and in a maximally parallel manner, which rules are applied to which objects. The output of the computation is collected from an a priori determined output region [Dassow and Paun, 1999; Păun, 2000; Păun and Rozenberg, 2002; Kari and Rozenberg, 2008; Păun, 2012]. Zhang et al. [2014] propose an approximation solution for a combinatorial optimization problem using P-systems. Many variant models have been studied, and interest has focused on proving computational universality for systems with a small number of membranes, for the purpose of solving NP-complete problems such as Boolean satisfiability (SAT) problems and the traveling salesman problem (TSP). P-systems may trade space and time complexities and less often use models to explain natural processes in living cells. The studies devise models that may at least theoretically be implemented on hardware. To date, P-systems are nearly all theoretical models that have never been applied in practical real-world tasks [Paun, 2006].

2.9 Collision-based computing

Collision-based computing is an implementation of logical circuits, mathematical machines, or other computing and information processing devices in homogeneous, uniform and unstructured media with traveling mobile localizations. A quanta of information is represented by a compact propagating pattern (gliders in cellular automata, solitons in optical systems, wave fragments in excitable chemical systems). Logical truth corresponds to presence of the localization, logical false to absence of the localization; logical values can also be represented by a particular state of the localization. When two or more traveling localizations collide, they change their velocity vectors and/or states. Post-collision trajectories and/or states of the localizations represent results of logical operations implemented by the collision.

Jakubowski et al. [1996] is a study which proposed to use solitons which transfer state information during collisions for computation / numerical simulation on 1D

cellular automata. Anastassiou et al. [2001] provides an experimental study to investigate the energy-exchange interactions between vector solitons and the effect of sequence collisions.

Jakubowski et al. [2002] is a review on solitons' potential as a computational element. They treat the discovery of solitons in cellular automata; give an abstract model for particle computation (particle machines); analyze the information transfer in collisions of (continuum) optical solitons and describe state transformations in collisions of Manakov (vector) solitons.

Adamatzky [2004] describes collision-based computing with signals represented by traveling wave fragments. Logical operations, or gates, are realized at sites of wave fragments collisions, which turn out to be sensitive to local perturbation.

Rand et al. [2004] showed that arbitrary computation is possible using collisions of Manakov solitons in a nonlithographic, nonlinear optical medium.

Adamatzky and de Lacy Costello [2007] studied interactions between localized, shape-preserving wave-fragments in the sub-excitable Belousov-Zhabotinsky medium. Potential applications: signal tuning and general programming of collision-based excitable chemical computers. Later, Adamatzky et al. [2011] showed that a vesicle filled with a Belousov-Zhabotinsky mixture can implement a range of basic logical functions. Adamatzky and Durand-Lose [2012] provide a survey of collision-based schemes, where particles/collisions are dimensionless.

In Zhou and Liao [2012], three dynamical systems are combined to design an image encryption algorithm. The algorithm is claimed to be fast, parallel and flexible to setting it to various precisions.

3 A selection of substrates

Here we highlight a choice of physical substrates which are currently being experimentally explored. Due to the exuberance of nature, new physical phenomena come into the compass of engineering curiosity almost daily. Our selection can only highlight a few instructive examples on which reservoir computing — being our homeground field — have been implemented. The basic idea of reservoir computing (RC) is to use a random excitable medium to expand an input signal into a higher-dimensional (nonlinear transform) signal space. The generality of this idea has invited to consider RC as a computational paradigm for use in "unconventional" physical or computational media that differ from neural network models or digital computing circuitry. Early didactic attempts to physically demonstrate reservoir behavior were made by producing waves on the surface of water [Fernando and Sojakka, 2003]. Since then, electrical circuits, optical media and chemical (molecular) substrates have been investigated to create a reservoir. We provide a review of such experimental studies to illustrate the progresses in this area and to address the computational bottlenecks which arise from specific hardware implementations. Moreover, to deal with challenges of computation on such unconventional substrates, several lines of potential solutions are presented

based on advances in other computational approaches in machine learning.

3.1 Electrical implementations

The idea of reservoir computing were first transferred back to an analog computing device: a mixed-mode VLSI neural network. In this study, earlier results of Bertschinger and Natschläger [2004] were reproduced showing that the readout with linear classifiers is especially successful when the network exhibits critical dynamics "at the edge of chaos" [Schürmann et al., 2004]. Later, implemented on an FPGA board, Schrauwen et al. [2008a] presented an application driven digital hardware exploration where they realized real-time, isolated digit speech recognition using a Liquid State Machine. They presented a novel hardware architecture based on serial processing of dendritic trees using serial arithmetic. Later, Antonik et al. [2015] demonstrated a standalone reservoir computer programmed on an FPGA board and applied it to the real-world task of equalisation of a nonlinear communication channel.

In order to reduce the large number of elements in the reservior, Appeltant et al. [2011b] proposed to implement a reservoir computer in which the usual structure of multiple connected nodes is replaced by a dynamical system comprising a nonlinear node subjected to delayed feedback. Through an electronic implementation, they experimentally and numerically reported excellent performance in a speech recognition and time series prediction benchmarks. The electrical substrate in this work is a combination of digital and analog blocks.

In 2012, [Appeltant et al., 2011b] developed the first memristor-based reservoir computing architecture and showed that simple pattern classification and associative memory tasks can be experimentally solved. Regarding the fact that memristors differ from resistors by possessing a memory, both synapses and neurons have been discussed as biological memristors [Gale, 2017]. However, in this study, the readout synaptic weights are trained by an evolutionary (genetic) algorithm and memristors only provide the required nonlinearity in the reservoir part. In a follow-up study, applying the reservoir computing approach to a regular structured reservoir, Bürger and Teuscher [2013] have shown that on a simple signal classification problem, compared to unstructured random reservoirs, regular structured reservoirs lead to better average performance as well as to higher variation tolerance. Snyder et al. [2013] have investigated the computational capabilities of random Boolean networks when used as the dynamical component in reservoir computing devices. They have found that computation (in terms of the balance of separability and fading memory of inputs) tends to be maximized at the critical connectivity. They claim that such critical dynamics might be desirable for heterogeneous substrates in RC. Other examples of experimental studies of memristor-based computing have been recently reviewed in Gale [2017]. This book chapter addresses two main issues: 1., how memristor spikes are a real-world memristor model of an inhibitory neuron and 2., how the complex emergent

behaviour from networks of memristors might resemble neural dynamics.

In another line of investigation, to facilitate understanding and control of atomic switch networks (ASN), Demis et al. [2016] developed a numerical model based on the synapse-like properties of individual atomic switches and the random nature of the network wiring. Then, to demonstrate their utility for computation, they subjected the simulated network to training within the framework of reservoir computing and showed initial evidence of the ASN acting as a reservoir which may be optimized for specific tasks by adjusting the input gain.

3.2 Optical realizations

In an early approach to photonic reservoir computing, networks of coupled semiconductor optical amplifiers (SOA) were used as the basic building blocks for a reservoir, and tested on standard RC benchmarks [Vandoorne et al., 2008, 2010, 2011].

Several groups investigated a non-standard optoelectronic reservoir architecture, namely an optical delay line that was pumped by modulated input through a single nonlinear input/feedback element. This approach yielded competitive performance on standard RC benchmark tasks and was worked out in detail and variations within European and national research projects [Appeltant et al., 2011b; Paquot et al., 2012; Larger et al., 2012]. Research along these lines is continuing to the day.

Vandoorne et al. [2014] presented an entirely passive silicon photonics reservoir. This chip can perform arbitrary Boolean logic operations with memory as well as 5-bit header recognition up to 12.5Gbit/sec, without power consumption in the reservoir.

Fiers et al. [2014] presented an optical architecture based on nanophotonic crystal cavities. In addition to possibility of integrating many neurons on one chip, the components are passive, which simplifies the design and reduces the power consumption. To assess the performance of this network, a photonic network has been trained to generate periodic patterns, using an alternative online learning rule called first-order reduced and corrected error. This system features a complex valued reservoir using the amplitude and phase of an optical (or optoelectrical) signal.

Most recently, Brunner et al. [2016] have started to explore all-optical reservoirs made from interacting nanoscale lasers.

3.3 Chemical realizations (DNA computing)

In addition to electrical and opto-electrical hardware implementations, there are other "even more unconventional" substrates for reservoir computating. Among them, DNA computing and computation using nano-scale semiconductor materials have been investigated to physically implement reservoir computing.

DNA computing is a form of parallel computing in that it takes advantage of the many different molecules of DNA to try many different possibilities at once. Goudarzi et al. [2013] have proposed and simulated a novel approach to DNA computing based on the reservoir computing paradigm. Using a network of oscillators built from deoxyribozymes, they extracted hidden features in a given input signal and computed a desired output. They tested the performance of this approach on two simple temporal tasks. Both have recursive time dependence and therefore require the reservoir to remember past inputs for some period of time, and both are simplified versions of a popular NARMA RC benchmark.

3.4 Quantum computing

Obst et al. [2013] describe preliminary steps towards nano-scale reservoir computing using optical properties of quantum dots. The research has focused on the development of an accumulator-based sensing system that reacts to changes in the environment, as well as the development of a software simulation. Dale et al. [2016] showed the first steps to applying the reservoir model as a simple computational layer to extract exploitable information from physical substrates consisting of single-walled carbon nanotubes and polymer mixtures.

4 A choice of challenges

In this section, we point out a choice of technological and information processing challenges that one finds in many or most non-classical substrates. These challenges have been proven very hard to overcome so far. By contrast, classical CMOS-based Turing-type computing and, to a lesser extent, classical analog signal processing can cope with most of these problems and thus are still the only real contenders for fielded applications.

With regards to the currently ongoing research efforts aiming at "neuromorphic" devices, we perceive three major types of challenges: 1. how to make best of neural spikes, 2. how to ensure computational robustness on microchips plagued by various sorts of device mismatch and parameter drift, and 3. how to cope with (very) low bit precision of static and dynamical variables.

4.1 Computing with spikes

A common objective for the design of spiking neuromorphic computing systems is to trade energy for quality of the computed result.

Tsai and et al [2017] describe a TrueNorth implementation of speech recognition which is fed by the raw (spikified) microphone signal - the time-to-frequency domain transformation (spectral analysis) is done with spiking neurons. It documents various precision-energy trade-offs and describes numerous "tricks" to realize signal processing functionality in spikes.

Emphasizing demands for low energy in Brain Computer Interfacing (BCI), Lungu et al. [2017], describe how the "Spikey" chip designed in Heidelberg is used to emulate an insect-brain inspired motion intention detector from 12 monkey-recorded neurons. Architecture combines hand-designed preprocessing network with elementary perceptron-like trained decoder.

In search for "foundry-friendly" (compatible with CMOS process) design of memristive synapes and low energy consumption, Yao and et al [2017] describe a memristive neurochip which is demonstrated on a (simple) face recognition task.

In addition to this energy-efficiency trade-off objective, in order to design spike based processing systems, more challenges remained unresolved:

- How to reconcile the discontinuous nature of spiking with the fact that most error functions are based on real-valued and continuous-time quantities?
- By definition computation in spiking neural networks is closely related to the challenge of encoding and decoding with spikes. It is important to note that much of the discussion on rate versus spike coding in neuroscience does not apply to spiking neural networks.
- Due to the differences in neural models, network topology (single layer, multiple layer, recurrent or not etc.) and encoding that different learning algorithms require, it is difficult to compare the performance in different spiking neural networks fairly. Hence tasks that can in principle be learnt by one algorithm are an unfair challenge for another and vice versa.
- To date no fully satisfactory general-purpose learning algorithm exists for networks of spiking neurons. Most of these approaches stand isolated and do not discuss how they compare or integrate with other similar approaches. This is on the one hand due to different network topologies, neuron models, neural codes and error functions as discussed. However comparison and analysis are also difficult because there is no common structured language to talk about these schemes. Formulation of equations for such algorithms is difficult as the inherent temporal dimension of spike trains introduces a notational load. There is therefore a need for a unified framework for formulating learning algorithms in spiking neural networks in continuous time.

4.2 The challenge of robustness

Advances in materials science and nanotechnology are making available a variety of unconventional computing substrates that can potentially replace top-down-designed silicon-based computing devices. However, inherent stochasticity in the fabrication process and nanometer scale of these substrates inevitably lead to design variations, defects, faults, and noise in the resulting devices. A key challenge is how to harness such devices to perform robust computation.

Silverman and Ikegami [2011] develop a firmer definition of and relation between two senses of robustness: robustness in systems, and robustness in explanation. When discussing systems, robustness is often described as a property which gives the system a certain resilience against perturbation. A robust system is thus able to retain functionality despite variation. In contrast, a robust explanation is a scientific explanation which can identify causal factors that underlie a phenomenon in a variety of circumstances. Thus, rather than an explanation which pertains to only once instance of behaviour in a system, a robust explanation will identify those elements which drive a systems behaviour as a whole.

4.2.1 Bio-inspired robustness

Highly optimized tolerance (HOT) was introduced as a conceptual framework to study the fundamental aspects of complexity and robustness [Carlson and Doyle, 2002]. HOT is motivated primarily by systems from biology and engineering and emphasizes, (i) highly structured, nongeneric, self-dissimilar internal configurations, and (ii) robust yet fragile external behavior. Based on this conceptual framework, a natural or technological system must posses some characteristic features to be robust. Modularity, robustness, having feedback loops and following the power law are some of these features [Carlson and Doyle, 2000; Stelling et al., 2004b]. In this regard, Kitano [2007, 2004] reviews numerous papers on how robustness is involved in various biological processes and on mechanisms that give rise to robustness in living systems. He also suggests a mathematical formulation for robustness in biological systems. Theoretical and experimental studies in the area of computational systems biology collect evidences, ideas, theories and computational models which address the issue of robustness in complex biological systems [Csete and Doyle, 2002; Kitano, 2004, 2002a; Stelling, 2004; Kitano, 2002b]. In recent years, also in immune system computing, the idea of degeneracy has been proposed as a design principle for achieving robustness and evolvability [Edelman and Gally, 2001; Whitacre, 2010; Whitacre and Bender, 2010] (compare Section 2.4.4).

4.2.2 Redundancy

Providing alternative ways to carry out the function that the component performs seems to be the simplest strategy to protect against failure of a specific component [Randles et al., 2011; Stelling et al., 2004b]. An important point here is that in biological systems two exactly similar components can not survive in evolution, but structurally different entities which perform overlapping functions are more appropriate components. This idea relates to previously mentioned idea of degeneracy. As an example of signal processing approaches to add redundancy, Kovačević and Chebira [2008] give an introduction to redundant signal representations called frames. Starck et al. [2004]; Stockwell [2007]; Elad and Aharon [2006]; Fuchs [2004] are examples of technical papers which provide strategies to add or

to use of redundancy in signal and image processing algorithms.

In a reservoir computing context, Bürger and Teuscher [2013] have reported that variation tolerance is improved by increasing the size of the reservoir. They claim that the inter-dependency of the memristors within a reservoir, with respect to their voltage-drop and hence their non-linearity is strongly related to the topology and the number of memristors. Increasing the number of memristors can lead to reduced individual voltage drop and non-linearity which in turn can degrade the error rate.

4.2.3 Closed loop feedback

By using the output of a function to be controlled in order to determine appropriate input signals, feedback enables a system to regulate the output by monitoring it. The degree to which a control circuit contributes to robustness strongly depends on the circuit design, control objective, and the type of perturbations affecting the system. Well-balanced positive and negative feedback can lead to a blend of sensitivity and stability [Stelling et al., 2004b]. Another possibility for achieving higher robustness consists in combining multiple levels of regulation.

To improve network robustness against update-rule perturbations, by using a random Boolean network model, Le and Kwon [2013] demonstrated that coherenat feed-forward loops (FFLs) can improve network robustness against update-rule perturbations. The role of delay feedback control on the recovery of dynamics and function in spiking neural networks, integral feedback control on robust perfect adaptation in bacterial chemotaxis, fast and slow positive feedbacks in cell decisions, multiple levels of regulation on circadian clock architectures have been investigated in systems biology studies [Vlachos et al., 2016; Yi et al., 2000; Brandman et al., 2005; Stelling et al., 2004a].

In this regard, different branches of robust control theory such as model predictive control [Song et al., 2007], sliding-mode control [Utkin, 1993; Vaidyanathan, 2015; Park et al., 2009], fractional-order control [Chen et al., 2009], H-infinity control, H-infinity loop-shaping, intelligent control (neural network controller or Bayesian controller) [Stengel, 1991; Ku and Lee, 1995], process control, robust decision making, state space control, active disturbance rejection control [Han, 2009], quantitative feedback theory give technical robust approaches [Stengel and Ryan, 1991; Bhattacharyya et al., 1995; Polyak and Tempo, 2001]. Chaos control strategies have been applied on recurrent neural networks [Laje and Buonomano, 2013; Vincent-Lamarre et al., 2016] (compare Section 2.6).

4.2.4 Modularity

Modularity, the encapsulation of functions, can contribute to both robustness of the entire system (by confining damage to separable parts) and to evolvability (by rewiring of modules or by modifications in modules that are unnoticeable from the outside) [Stelling et al., 2004b]. In biological studies, topological analysis sug-

gested that cellular regulation uses recurrent, simple building blocks ("network motifs") to perform functions such as feedback control [Yeger-Lotem et al., 2004]. Many cellular networks are claimed to have a "scale-free" organization with few central hubs and many nodes bearing few links [Barabasi and Oltvai, 2004]. Such structures were suggested to resist random perturbations because of a low probability that a perturbation affects the sensitive hubs [Albert et al., 2000; Jeong et al., 2000; Zhao et al., 2004].

Neftci et al. [2013] develop a strategy for realizing reliable intelligent computation on unreliable, low-precision neuromorphic hardware by combining redundancy with modularity. They specify the task in the format of a discrete state transition system, translate this into an abstract model of competing winner-take-all circuits, and finally map this abstract model on a neuromorphic microchip (with population redundancy) by an automated calibration transformation which adjusts the parameters of the abstract model to the parameters that are physically realized.

4.2.5 Hierarchy

An efficient means for coordination in complex systems is to organize a system hierarchically, namely to establish different layers of integration to reduce the costs of information transmission [Stelling et al., 2004b]. In Stelling et al. [2004b] there are several examples of biological systems in which the system's possible behavior on a lower level is constrained by regulation at higher levels. Wang and Zhou [2012] have investigated the role of hierarchical modular networks structure in improving the robustness of self-organized criticality in neural networks. Intuitively, the modular structure limits the size of avalanches in neural networks due to sparser connections between modules. Using computer simulations, they have shown that the critical region can be further extended significantly by the modular structure. In another study, Bagrow et al. [2015] have used an analytically tractable network model to study the robustness of modular networks to the random failures of elements. By analyzing a second network detailing the connectivity between modules, they have shown that the overlapping modular structure of the network is more susceptible to random failures than expected. Robust hierarchical networks are mainly investigated in "complex networks", "scale-free networks", "Internet network", "power grid networks" and "social networks" areas of research [Callaway et al., 2000; Motter, 2004].

4.2.6 Local homeostatic mechanisms

The use of homeostatic neurons as the basic building blocks for evolved neurocontrollers proposed in Di Paolo [2000] provides a novel approach to modelling adaptivity in artificial systems in a way that resembles the adaptive dynamics of living organisms [Iizuka and Di Paolo, 2008]. The idea behind homeostatic adaptation is based on that of the ultrastable system proposed by Ashby. This is a system – open to interaction with the environment – which will tend to change

its own configuration plastically whenever stability is lost and until it finds a new internal dynamics which will make the system stable under the current conditions. Such systems are capable of remarkable adaptation and learning. They have been applied to legged robot locomotion [Hoinville and Henaff, 2004], extended to different types of plastic functions [Williams, 2004], applied to the study of the minimal dynamics of behavioral preference [Iizuka and Di Paolo, 2007]. In Iizuka and Di Paolo [2008], an extended homeostatic neural controller has been proposed where neurons are biased to have a strong resting membrane potential and an additional fitness condition rewarding not only a positive link between homeostasis and a desired behavior but also a negative one between the breakdown of homeostasis and undesired behavior This extended model has been used in Iizuka et al. [2013] to construct a model of mental imagery.

Homeostatic plasticity refers to the capacity of neurons and synapses to regulate their own excitability relative to the network activity, usually in response to an imbalance or external disturbances. The term homeostatic plasticity derives from two opposing concepts: homeostasis (Greek for "same state") and plasticity ("change"), thus homeostatic plasticity means "staying in the same state through change". Synaptic scaling is a homeostatic mechanism that increases or decreases the strength of all of a neurons synaptic inputs as a function of activity [Turrigiano, 2008]. In neural network models, a common implementation for synaptic scaling is synaptic normalization in which the incoming connections to a neuron are proportionally adjusted so that they sum up to a constant value [Lazar et al., 2009]. Chandler and Grossberg [2012] show how mechanisms of synaptic scaling can be incorporated within recurrent on-center off-surround networks in such a way that their pattern processing capabilities, including the ability to make winner-take-all decisions, is preserved. Intrinsic plasticity (IP) is the persistent modification of a neurons excitability. It is mediated by the properties of ion channels in the neurons membrane. For the same synaptic input, changes in neuronal excitability will produce a different output. IP tends to reduce the intrinsic excitability of a neuron during long periods of stimulation and increase excitability during activity deprivation [Lazar et al., 2009]. A review paper on local and global homeostatic mechanisms shows that on a functional level, neuronal networks likely use a complex set of regulatory mechanisms to achieve homeostasis over a wide range of temporal and spatial scales [Turrigiano, 2012]. Another reviews on experimental and modeling finding were also presented in Turrigiano [1999]; Marder and Prinz [2002]; Turrigiano and Nelson [2004]; Marder and Goaillard [2006].

In recurrent neural network application of homeostatic plasticity, Remme and Wadman [2012] studied the functioning of activity-dependent homeostatic scaling of intrinsic excitability (HSE) in a recurrent neural network. Using both simulations of a recurrent network consisting of excitatory and inhibitory neurons that implement HSE, and a mean-field description of adapting excitatory and inhibitory populations, they show that the stability of such adapting networks critically depends on the relationship between the adaptation time scales of both

neuron populations. In another study, Williams and Noble [2007] showed that the effects of homeostatic plasticity on continuous-time recurrent neural networks are increased sensitivity and improved signal propagation. The reason for this is that the homeostatic plasticity offers a directed mechanism which avoids node saturation and maintains each node in the network in its most responsive region of parameter space. At the network level this offers improved signal propagation. Sensitive nodes make sensitive networks. Lazar et al. [2007] have investigated how different forms of plasticity shape the dynamics and computational properties of simple recurrent spiking neural networks. In particular, they studied the effect of combining two forms of neuronal plasticity: spike timing dependent plasticity (STDP), which changes the synaptic strength, and intrinsic plasticity (IP), which changes the excitability of individual neurons to maintain homeostasis of their activity. Schrauwen et al. [2008b] provide an example of unsupervised reservoir adaptation using IP mechanisms.

In physical neuromorphic microchips, an important problem is to preserve a constant envelope of synaptic gains over long timespans in the face of physical parameter drift and short-timescale gain changes due to learning. In a series of designs in the Zurich INI group (for instance Bartolozzi and Indiveri [2009]; Qiao et al. [2016]) local feedback control circuits are developed whose latest versions are realizable in standard CMOS technology and achieve this objective over timespans of many hours, with extremely small energy demands.

4.2.7 Taking advantage of noise

As opposed to the idea of reducing the perturbations to improve robustness, there are studies which propose to take advantages of noise in computation system. Wiesenfeld et al. [1995] have introduced stochastic resonance to use noise to enhance weak signal detection in non-linear electrical circuits. Later, Reinker et al. [2004] investigated the effect on membrane resonance and stochastic resonance on firing rate of a Hodgkin-Huxley model of an excitable cell. A similar study but on the phasic neuron model was described in Gai et al. [2010]. Katada and Nishimura [2009] investigated how stochastic resonance behavior can be observed in practical autoassociative neural networks with the Hopfield-type memory under stochastic dynamics. There are several papers which investigate this phenomena in chaotic systems, bistable systems and neuronal models.

Gai et al. [2010] have argued that the physics approach that characterizes much of the previous work in stochastic resonance has not exploited all of the possibilities for advancing our understanding of the beneficial role of noise in neural systems.

Clusella and Politi [2017] illustrates a counter-intuitive effect of an additive stochastic force, which acts independently on each element of an ensemble of globally coupled oscillators. They show numerically and semi-analytically that a very small white noise is able to stabilize an otherwise linearly unstable collective periodic regime.

A source of randomness which comes for free on neuromorphic microchips

is fabrication mismatch between computational elements that ideally should be identical. Sheik et al. [2012] use this heterogeneity of silicon neurons to design a neuromorphic multichip architecture where the span of spike response delays of neurons in a population of interneurons is used to configure axonal delays between neurons in another population; such configurable delays are needed for certain biologically motivated algorithms.

Physically fluctuating synaptic efficiency is exploited by Neftci et al. [2016] for neural sampling in a variation of the restricted Boltzmann machine (RBM) which they call Synaptic Sampling Machines (S2Ms). The S2M compares favorably to the standard RBM in a MNIST character recognition benchmark and was demonstrated to be robust (after short re-training) against connection pruning and weight precision reduction.

4.3 The challenge of low bit precision

Most literature on low-bit precision comes from the deep learning community where the objective is to achieve power-efficient digital implementations of deep neural networks.

The major advance has been the discovery of a backprop-based learning method that quantizes weights during propagations. This training method was introduced in Courbariaux et al. [2015b], where they trained state-of-the-art neural networks on benchmark datasets with three distinct formats: floating point, fixed point and dynamic fixed point. They assessed the impact of the precision of the multiplications on the final error after training, since multipliers are the most space and power-hungry arithmetic operators in hardware. As a result, they found that 10-bit precision is sufficient not just for running trained networks but also for training them.

Remarkably, a slew of recent work has shown that using just two (binary) or three (ternary) values for weights, DNNs can approach state of the art performance on popular benchmarks [Rastegari et al., 2016; Esser et al., 2016a; Courbariaux et al., 2015a; Esser et al., 2015]. The basic approach is to apply gradient descent using high precision weights during training, and project the weights via a quantization function (such as the sign function) in forward/backward passes. In this way, the high precision weights are able to accumulate small gradient updates computed with respect to the projected ones, allowing the network to explore discrete configurations in a continuous setting. It has been argued that this training procedure is essential to learning low precision representations [Courbariaux et al., 2015a]. This method was further developed in the context of spiking neuromorphic hardware in Esser et al. [2016a], where they employ a binary representation scheme for data throughout the network in order to match the use of spikes in hardware. Stromatias et al. [2015b] proposed a similar rule for contrastive divergence.

Merolla et al. [2016] from IBM shift the focus from the idea that quantizing weights during training leads to networks robust to that quantization, to a more

general phenomenon, where projecting weights via certain functions (not necessarily quantization) lead to networks robust not only to that function (or distortion), but to an entire class of distortions. This is as if a patient given the flu vaccine, finds themselves inoculated against measles, mumps, and malaria as well. They showed that deep neural networks are robust to weight binarization and other non-linear distortions beyond quantization, including additive and multiplicative noise, and a class of non-linear projections where binarization is just a special case.

Several other methods attempt to binarize the weights and the activations in neural networks. In the following long paragraph, we heavily re-use the survey given in Courbariaux et al. [2016], citing passages almost verbatim, with a few additions.

The performance of highly quantized networks (e.g., binarized) were believed to be very poor due to the destructive property of binary quantization [Courbariaux et al., 2015b]. Cheng et al. [2015] and Soudry et al. [2014] proved the contrary by showing that good performance could be achieved even if all neurons and weights are binarized to ± 1. Expectation backpropagation (EBP) in Soudry et al. [2014] showed that high performance can be achieved by a network with binary weights and binary activations. This is done by a variational Bayesian approach with mean-field and central limit approximation to calculate the posterior distribution of the weights (the probability of each weight to be +1 or -1). During the inference stage (test phase), their method samples from this distribution on a binary network and uses it to predict the targets of the test set (more than one binary network can also be used). A fully binary network at run time presented in Esser et al. [2015] using a similar approach to EBP showed significant improvement in energy efficiency. In EBP the binarized parameters were only used during inference. BinaryConnect [Courbariaux et al., 2015a] extended the probabilistic idea behind the EBP, and used the real-valued version of the weights as a key reference for the binarization process. The real-valued weight was updated using the back propagated error by simply ignoring the binarization in the update. BinaryConnect achieved state-of-the-art results on small datasets (e.g., CIFAR-10, SVHN). Rastegari et al. [2016] mentioned that their experiments indicate that BinaryConnect method is not very successful on large-scale datasets (e.g., ImageNet). The extension of BinaryConnect was proposed by BinaryNet [Courbariaux and Bengio, 2016], where both weights and activations are binarized. Kim and Smaragdis [2015] convert a real-valued network into one where weight parameters, bias terms, input, and intermediate hidden layer output signals, are all binary-valued. Two-stage training schemes are adopted: weight compression followed by noisy backprop. Another recent approach called XNOR-net approximates weights using binary values during training, formulated as a constrained optimization problem, with promising results on even large datasets such as ImageNet [Rastegari et al., 2016]. They propose two efficient approximations to standard convolutional neural networks: Binary-Weight-Networks and XNOR-Networks. In Binary-Weight-Networks, the filters are approximated with binary

values and an additional scaling factor. In XNOR-Networks, both the filters and the input to convolutional layers are binary. XNOR-Networks approximate convolutions using primarily binary operations. This results in 58× faster convolutional operations and 32× memory savings. Very recent work in this line of research is Quantized Neural Networks [Courbariaux et al., 2016], which proposes a technique for quantizing the neurons and weights during inference and training. In such networks, all multiply-accumulate operations (MACs) can be replaced with XNOR and population count (i.e., counting the number of ones in the binary number) operations.

Another line of work come from statistical physicists Carlo Baldassi, Riccardo Zecchina and others. In the past decades, various methods borrowed from statistical physics have been quite successful in studying the basic properties of neural-like systems. A well known general result is that, as is the case for other optimization problems, training neural networks is qualitatively different if the variables (the synaptic weights) are constrained to take discrete values. Yet, the standard equilibrium analysis, which suggests that the solutions to the constrained problem would be inaccessible, is at odds with some recent heuristic algorithmic advances [Braunstein and Zecchina, 2006; Baldassi et al., 2007; Baldassi, 2009; Baldassi and Braunstein, 2015], which demonstrate that simple effective protocols may be devised at least in some simple scenarios. Their recent work [Baldassi et al., 2015] proposed fully binary training and testing in an array of committee machines with randomized input. The standard statistical analysis shows that learning random patterns with binary synapses in single layer networks is exponentially dominated by isolated solutions that are extremely hard to find algorithmically. However, they introduce a novel method that allows them to find analytical evidence for the existence of sub-dominant and extremely dense regions of solutions. Numerical experiments confirm these findings. They also show that the dense regions are surprisingly accessible by simple learning protocols based on local entropy maximization.

Subhrajit Roy, Arindam Basu and their colleagues focus on implementing neural networks with $0, 1$ binary weights in spiking neuromorphic hardwares, and they take inspirations from studies on structural plasticity [Poirazi and Mel, 2001; Spiess et al., 2016] and dendritic processing in neuroscience. For example, in Hussain et al. [2015], they proposed a hardware-amenable structural learning algorithm for spike-based pattern classification using active dendrites. The learning algorithm groups correlated synaptic inputs on the same dendritic branch. Since the learning results in modified connection patterns, it can be incorporated into current event-based neuromorphic systems with little overhead. Roy and Basu [2016] proposed a winner-take-all (WTA) architecture employing neurons with nonlinear dendrites and an online unsupervised structural plasticity rule for training it through formation and elimination of connections between inputs and synapses. They demonstrated the performance of this network on two-class, four-class, and six-class classification of random Poisson spike time inputs, and use

the inhibitory time constant to set the number of subpatterns per pattern they want to detect. Hussain and Basu [2016] extend the lumped dendritic model in Hussain et al. [2015] by introducing multiple compartments on a dendrite where each compartment performs location-dependent processing of synaptic inputs.

Sheridan et al. [2015] designed a crossbar memristor device model to perform feature extraction tasks. They first apply a winner-take-all and Oja's rule approach to learn a dictionary, and weights are adjusted by applying pulses with durations proportional to the desired change. Then they introduced a locally competitive algorithm (LCA) to represent input images. As a compensating procedure to ensure robustness against parameter variation and nonlinearity during training, the pulse durations were adjusted according to a noise function depending on the voltage and material parameters.

References

Abbott, L., DePasquale, B., and Memmesheimer, R.-M. (2016). Building functional networks of spiking model neurons. *Nature Neuroscience*, 19(3):350–355.

Adamatzky, A. (2004). Collision-based computing in Belousov–Zhabotinsky medium. *Chaos, Solitons & Fractals*, 21(5):1259–1264.

Adamatzky, A. and Chen, G. (2013). *Chaos, CNN, Memristors and Beyond: A Festschrift for Leon Chua*. World Scientific.

Adamatzky, A. and Chua, L. (2011). Memristive excitable cellular automata. *International Journal of Bifurcation and Chaos*, 21(11):3083–3102.

Adamatzky, A. and de Lacy Costello, B. (2007). Binary collisions between wave-fragments in a sub-excitable Belousov–Zhabotinsky medium. *Chaos, Solitons & Fractals*, 34(2):307–315.

Adamatzky, A. and Durand-Lose, J. (2012). *Collision-based computing*. Springer.

Adamatzky, A., Holley, J., Bull, L., and Costello, B. D. L. (2011). On computing in fine-grained compartmentalised Belousov–Zhabotinsky medium. *Chaos, Solitons & Fractals*, 44(10):779–790.

Adamatzky, A. and Komosinski, M. (2009). *Artificial life models in hardware*. Springer Science & Business Media.

Adamatzky, A. and Wuensche, A. (2006). Computing in spiral rule reaction-diffusion hexagonal cellular automaton. *Complex Systems*, 16(4):277–298.

Adami, C. (1998). *Introduction to Artificial Life*, volume 1. Springer Science & Business Media.

Aihara, K., Takabe, T., and Toyoda, M. (1990). Chaotic neural networks. *Physics Letters A*, 144(6-7):333–340.

Alaghi, A. and Hayes, J. P. (2013). Survey of stochastic computing. *ACM Transactions on Embedded Computing Systems*, 12(2s):article nr. 92.

Albert, R., Jeong, H., and Barabási, A.-L. (2000). Error and attack tolerance of complex networks. *Nature*, 406(6794):378–382.

Alsing, P. M., Gavrielides, A., and Kovanis, V. (1994). Using neural networks for controlling chaos. *Physical Review E*, 49(2):1225.

Anastassiou, C., Fleischer, J. W., Carmon, T., Segev, M., and Steiglitz, K. (2001). Information transfer via cascaded collisions of vector solitons. *Optics Letters*, 26(19):1498–1500.

Antonelo, E., Schrauwen, B., and Stroobandt, D. (2008a). Mobile robot control in the road sign problem using reservoir computing networks. In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pages 911–916. IEEE.

Antonelo, E. A., Schrauwen, B., and Stroobandt, D. (2008b). Event detection and localization for small mobile robots using reservoir computing. *Neural Networks*, 21(6):862–871.

Antonik, P., Smerieri, A., Duport, F., Haelterman, M., and Massar, S. (2015). Fpga implementation of reservoir computing with online learning. In *24th Belgian-Dutch Conference on Machine Learning*.

Appeltant, L., Soriano, M. C., Van der Sande, G., Danckaert, J., Massar, S., Dambre, J., Schrauwen, B., Mirasso, C. R., and Fischer, I. (2011a). Information processing using a single dynamical node as complex system. *Nature Communications*, 2:468.

Appeltant, L., Soriano, M. C., Van der Sande, G., Danckaert, J., Massar, S., Dambre, J., Schrauwen, B., Mirasso, C. R., and Fischer, I. (2011b). Information processing using a single dynamical node as complex system. *Nature Communications*, 2(468). DOI: 10.1038/ncomms1476.

Babadi, B. and Abbott, L. (2016). Stability and competition in multi-spike models of spike-timing dependent plasticity. *PLoS Comput Biol*, 12(3):e1004750.

Bagrow, J. P., Lehmann, S., and Ahn, Y.-Y. (2015). Robustness and modular structure in networks. *Network Science*, 3(04):509–525.

Baldassi, C. (2009). Generalization learning in a perceptron with binary synapses. *Journal of Statistical Physics*, 136(5):902–916.

Baldassi, C. and Braunstein, A. (2015). A Max-Sum algorithm for training discrete neural networks. *ArXiv e-prints*, 08008:1–23.

Baldassi, C., Braunstein, A., Brunel, N., and Zecchina, R. (2007). Efficient supervised learning in networks with binary synapses. *Proceedings of the National Academy of Sciences of the United States of America*, 104(26):11079–11084.

Baldassi, C., Ingrosso, A., Lucibello, C., Saglietti, L., and Zecchina, R. (2015). Subdominant Dense Clusters Allow for Simple Learning and High Computational Performance in Neural Networks with Discrete Synapses. *Physical Review Letters*, 115(12):1–5.

Barabasi, A.-L. and Oltvai, Z. N. (2004). Network biology: understanding the cell's functional organization. *Nature Reviews Genetics*, 5(2):101–113.

Bartolozzi, C. and Indiveri, G. (2009). Global scaling of synaptic efficacy: Homeostasis in silicon synapses. *Neurocomputing*, 72:726–731.

Beck, J., Pouget, A., and Heller, K. A. (2012). Complex inference in neural circuits with probabilistic population codes and topic models. In *Advances in neural information processing systems*, pages 3059–3067.

Bedau, M. A. (2003). Artificial life: organization, adaptation and complexity from the bottom up. *Trends in Cognitive Sciences*, 7(11):505–512.

Bedau, M. A., McCaskill, J. S., Packard, N. H., Rasmussen, S., Adami, C., Green, D. G., Ikegami, T., Kaneko, K., and Ray, T. S. (2000). Open problems in artificial life. *Artificial Life*, 6(4):363–376.

Behring, C., Bracho, M., Castro, M., and Moreno, J. (2001). An algorithm for robot path planning with cellular automata. In *Theory and Practical Issues on Cellular Automata*, pages 11–19. Springer.

Beni, G. and Wang, J. (1993). Swarm intelligence in cellular robotic systems. In *Robots and Biological Systems: Towards a New Bionics?*, pages 703–712. Springer.

Bertschinger, N. and Natschläger, T. (2004). Real-time computation at the edge of chaos in recurrent neural networks. *Neural Computation*, 16(7):1413–1436.

Bessler, W. G. (2005). A new computational approach for sofc impedance from detailed electrochemical reaction–diffusion models. *Solid State Ionics*, 176(11):997–1011.

Bhattacharyya, S., Chapellat, H., and Keel, L. (1995). Robust control: the parametric approach. *Upper Saddle River*.

Bialek, W., Rieke, F., De Ruyter Van Steveninck, R. R., and Warland, D. (1991). Reading a neural code. *Science*, 252(5014):1854–1857.

Bilchev, G. and Parmee, I. C. (1995). The ant colony metaphor for searching continuous design spaces. In *AISB workshop on evolutionary computing*, pages 25–39. Springer.

Bill, J., Buesing, L., Habenschuss, S., Nessler, B., Maass, W., and Legenstein, R. (2015). Distributed bayesian computation and self-organized learning in sheets of spiking neurons with local lateral inhibition. *PloS One*, 10(8):e0134356.

Boccato, L., Lopes, A., Attux, R., and Von Zuben, F. J. (2011). An echo state network architecture based on volterra filtering and pca with application to the channel equalization problem. In *Neural Networks (IJCNN), The 2011 International Joint Conference on*, pages 580–587. IEEE.

Boden, M. A. (1996). *The philosophy of artificial life*. Oxford University Press.

Boerlin, M. and Denève, S. (2011). Spike-based population coding and working memory. *PLoS Comput Biol*, 7(2):e1001080.

Bohte, S. M. (2012). Efficient spike-coding with multiplicative adaptation in a spike response model. In *Advances in Neural Information Processing Systems*, pages 1835–1843.

Bohte, S. M., Kok, J. N., and La Poutre, H. (2002). Error-backpropagation in temporally encoded networks of spiking neurons. *Neurocomputing*, 48(1):17–37.

Bohte, S. M. and Mozer, M. C. (2007). Reducing the variability of neural responses: a computational theory of spike-timing-dependent plasticity. *Neural Computation*, 19(2):371–403.

Bradley, D. W. and Tyrrell, A. M. (2000). Immunotronics: Hardware fault tolerance inspired by the immune system. In *International Conference on Evolvable Systems*, pages 11–20. Springer.

Brandman, O., Ferrell, J. E., Li, R., and Meyer, T. (2005). Interlinked fast and slow positive feedback loops drive reliable cell decisions. *Science*, 310(5747):496–498.

Braunstein, A. and Zecchina, R. (2006). Learning by message passing in networks of discrete synapses. *Physical Review Letters*, 96(3):1–4.

Brea, J., Senn, W., and Pfister, J.-P. (2013). Matching recall and storage in sequence learning with spiking neural networks. *The Journal of Neuroscience*, 33(23):9565–9575.

Brette, R. (2015). Philosophy of the spike: Rate-based vs. spike-based theories of the brain. *Frontiers in Systems Neuroscience*, 9.

Brette, R. and Gerstner, W. (2005). Adaptive exponential integrate-and-fire model as an effective description of neuronal activity. *Journal of Neurophysiology*, 94(5):3637–3642.

Brody, C. D. and Hopfield, J. (2003). Simple networks for spike-timing-based computation, with application to olfactory processing. *Neuron*, 37(5):843–852.

Brunel, N. (2000). Dynamics of sparsely connected networks of excitatory and inhibitory spiking neurons. *Journal of Computational Neuroscience*, 8(3):183–208.

Brunner, D., Reitzenstein, S., and Fischer, I. (2016). All-optical neuromorphic computing in optical networks of semiconductor lasers. In *Rebooting Computing (ICRC), IEEE International Conference on*, pages 1–2. IEEE.

Buesing, L., Bill, J., Nessler, B., and Maass, W. (2011). Neural dynamics as sampling: a model for stochastic computation in recurrent networks of spiking neurons. *PLoS Comput Biol*, 7(11):e1002211.

Bürger, J. and Teuscher, C. (2013). Variation-tolerant computing with memristive reservoirs. In *2013 IEEE/ACM International Symposium on Nanoscale Architectures (NANOARCH)*, pages 1–6. IEEE.

Burgin, M. and Dodig-Crnkovic, G. (2013). From the closed classical algorithmic universe to an open world of algorithmic constellations. In *Computing Nature*, pages 241–253. Springer Verlag.

Buscarino, A., Fortuna, L., Frasca, M., and Gambuzza, L. V. (2012). A chaotic circuit based on hewlett-packard memristor. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 22(2):023136.

Callaway, D. S., Newman, M. E., Strogatz, S. H., and Watts, D. J. (2000). Network robustness and fragility: Percolation on random graphs. *Physical Review Letters*, 85(25):5468.

Cao, Y., Chen, Y., and Khosla, D. (2015). Spiking deep convolutional neural networks for energy-efficient object recognition. *International Journal of Computer Vision*, 113(1):54–66.

Carlson, J. M. and Doyle, J. (2000). Highly optimized tolerance: Robustness and design in complex systems. *Physical Review Letters*, 84(11):2529.

Carlson, J. M. and Doyle, J. (2002). Complexity and robustness. *Proceedings of the National Academy of Sciences*, 99(suppl 1):2538–2545.

Chandler, B. and Grossberg, S. (2012). Joining distributed pattern processing and homeostatic plasticity in recurrent on-center off-surround shunting networks: Noise, saturation, short-term memory, synaptic scaling, and bdnf. *Neural Networks*, 25:21–29.

Chang, H.-J., Freeman, W. J., and Burke, B. C. (1998). Biologically modeled noise stabilizing neurodynamics for pattern recognition. *International Journal of Bifurcation and chaos*, 8(02):321–345.

Chechik, G. (2003). Spike-timing-dependent plasticity and relevant mutual information maximization. *Neural Computation*, 15(7):1481–1510.

Chen, L. and Aihara, K. (1995). Chaotic simulated annealing by a neural network model with transient chaos. *Neural Networks*, 8(6):915–930.

Chen, Y., Petras, I., and Xue, D. (2009). Fractional order control-a tutorial. In *2009 American control conference*, pages 1397–1411. IEEE.

Cheng, Z., Soudry, D., Mao, Z., and Lan, Z. (2015). Training Binary Multilayer Neural Networks for Image Classification using Expectation Backpropgation. *arXiv:1503.03562*.

Chialvo, D. R. (1995). Generic excitable dynamics on a two-dimensional map. *Chaos, Solitons & Fractals*, 5(3):461–479.

Chicca, E., Stefanini, F., Bartolozzi, C., and Indiveri, G. (2014). Neuromorphic electronic circuits for building autonomous cognitive systems. *Proc. of the IEEE*, 102(9):1367–1388.

Clopath, C., Büsing, L., Vasilaki, E., and Gerstner, W. (2010). Connectivity reflects coding: a model of voltage-based stdp with homeostasis. *Nature Neuroscience*, 13(3):344–352.

Clusella, P. and Politi, A. (2017). Noise-induced stabilization of collective dynamics. *arXiv preprint arXiv:1705.01038*.

Coello, C. A. C. and Cortés, N. C. (2005). Solving multiobjective optimization problems using an artificial immune system. *Genetic Programming and Evolvable Machines*, 6(2):163–190.

Corneil, D. S., Neftci, E., Indiveri, G., and Pfeiffer, M. (2014). Learning, inference, and replay of hidden state sequences in recurrent spiking neural networks. In *COSYNE 2014*. EPFL-POSTER-197325.

Coultrip, R., Granger, R., and Lynch, G. (1992). A cortical model of winner-take-all competition via lateral inhibition. *Neural Networks*, 5(1):47–54.

Courbage, M., Nekorkin, V., and Vdovin, L. (2007). Chaotic oscillations in a map-based model of neural activity. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 17(4):043109.

Courbariaux, M. and Bengio, Y. (2016). BinaryNet: Training Deep Neural Networks with Weights and Activations Constrained to +1 or -1. *arXiv e-print*.

Courbariaux, M., Bengio, Y., and David, J.-P. (2015a). BinaryConnect: Training Deep Neural Networks with binary weights during propagations. *Advances in Neural Information Processing Systems.*

Courbariaux, M., Bengio, Y., and David, J.-P. (2015b). Training deep neural networks with low precision multiplications. *ArXiv e-prints*.

Courbariaux, M., Hubara, I., Soudry, D., El-Yaniv, R., and Bengio, Y. (2016). Quantized neural networks: Training neural networks with low precision weights and activations. *ArXiv e-prints*, pages 1–29.

Csete, M. E. and Doyle, J. C. (2002). Reverse engineering of biological complexity. *Science*, 295(5560):1664–1669.

Dale, M., Miller, J. F., Stepney, S., and Trefzer, M. A. (2016). Evolving carbon nanotube reservoir computers. In *International Conference on Unconventional Computation and Natural Computation*, pages 49–61. Springer.

Darabos, C., Giacobini, M., and Tomassini, M. (2007). Performance and robustness of cellular automata computation on irregular networks. *Advances in Complex Systems*, 10(supp01):85–110.

Das, S., Biswas, A., Dasgupta, S., and Abraham, A. (2009). Bacterial foraging optimization algorithm: theoretical foundations, analysis, and applications. In *Foundations of Computational Intelligence Volume 3*, pages 23–55. Springer.

DasGupta, D. (1993). An overview of artificial immune systems and their applications. In *Artificial Immune Systems and their Applications*, pages 3–21. Springer.

Dasgupta, D. (2006). Advances in artificial immune systems. *IEEE Computational Intelligence Magazine*, 1(4):40–49.

Dasgupta, D., Ji, Z., González, F. A., et al. (2003). Artificial immune system (ais) research in the last five years. In *IEEE Congress on Evolutionary Computation (1)*, pages 123–130.

Dassow, J. and Paun, G. (1999). On the power of membrane computing. *Journal of Universal Computer Science*, 5(2):33–49.

De Castro, L. N. (2006). *Fundamentals of natural computing: basic concepts, algorithms, and applications*. CRC Press.

de Castro, L. N. (2007). Fundamentals of natural computing: an overview. *Physics of Life Reviews*, 4(1):1–36.

De Castro, L. N. and Timmis, J. (2002a). An artificial immune network for multimodal function optimization. In *Evolutionary Computation, 2002. CEC'02. Proceedings of the 2002 Congress on*, volume 1, pages 699–704. IEEE.

De Castro, L. N. and Timmis, J. (2002b). *Artificial immune systems: a new computational intelligence approach*. Springer Science & Business Media.

de Castro, L. N. and Timmis, J. (2003). Artificial immune systems as a novel soft computing paradigm. *Soft Computing*, 7(8):526–544.

De Castro, L. N. and Von Zuben, F. J. (2002). Learning and optimization using the clonal selection principle. *IEEE transactions on evolutionary computation*, 6(3):239–251.

De Wit, A. (1999). Spatial patterns and spatiotemporal dynamics in chemical systems. *Advances in Chemical Physics*, 109:435–514.

Delbruck, T. (2008). Frame-free dynamic digital vision. In *Proceedings of Intl. Symp. on Secure-Life Electronics, Advanced Electronics for Quality Life and Society*, pages 21–26.

Demis, E. C., Aguilera, R., Scharnhorst, K., Aono, M., Stieg, A. Z., and Gimzewski, J. K. (2016). Nanoarchitectonic atomic switch networks for unconventional computing. *Japanese Journal of Applied Physics*, 55(11):1102B2.

DePasquale, B., Churchland, M. M., and Abbott, L. (2016). Using firing-rate dynamics to train recurrent networks of spiking model neurons. *arXiv preprint arXiv:1601.07620*.

Di Paolo, E. (2000). Homeostatic adaptation to inversion of the visual field and other sensorimotor disruptions. In *From Animals to Animats 6. Proceedings of the Sixth International Conference on the Simulation of Adaptive Behavior*.

Diehl, P. U., Neil, D., Binas, J., Cook, M., Liu, S.-C., and Pfeiffer, M. (2015). Fast-classifying, high-accuracy spiking deep networks through weight and threshold balancing. In *2015 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE.

Ditto, W. L., Murali, K., and Sinha, S. (2008). Chaos computing: ideas and implementations. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 366(1865):653–664.

Dittrich, P. and Di Fenizio, P. S. (2007). Chemical organisation theory. *Bulletin of Mathematical Biology*, 69(4):1199–1231.

Dittrich, P., Liljeros, F., Soulier, A., and Banzhaf, W. (2000). Spontaneous group formation in the seceder model. *Physical Review Letters*, 84(14):3205.

Dittrich, P., Ziegler, J., and Banzhaf, W. (2001). Artificial chemistriesa review. *Artificial Life*, 7(3):225–275.

Dorigo, M., Birattari, M., and Stutzle, T. (2006). Ant colony optimization. *IEEE Computational Intelligence Magazine*, 1(4):28–39.

Dorigo, M. and Blum, C. (2005). Ant colony optimization theory: A survey. *Theoretical Computer Science*, 344(2):243–278.

Dorigo, M. and Stützle, T. (2009). Ant colony optimization: overview and recent advances. *Techreport, IRIDIA, Universite Libre de Bruxelles*.

Doya, K. (2000). Reinforcement learning in continuous time and space. *Neural Computation*, 12(1):219–245.

Duport, F., Smerieri, A., Akrout, A., Haelterman, M., and Massar, S. (2016). Fully analogue photonic reservoir computer. *Scientific Reports*, 6.

Edelman, G. M. and Gally, J. A. (2001). Degeneracy and complexity in biological systems. *Proceedings of the National Academy of Sciences*, 98(24):13763–13768.

Elad, M. and Aharon, M. (2006). Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image Processing*, 15(12):3736–3745.

Ermentrout, G. B. and Edelstein-Keshet, L. (1993). Cellular automata approaches to biological modeling. *Journal of Theoretical Biology*, 160(1):97–133.

Esser, S. K., Appuswamy, R., Merolla, P., Arthur, J. V., and Modha, D. S. (2015). Backpropagation for energy-efficient neuromorphic computing. *Advances in neural information processing systems*, pages 1117–1125.

Esser, S. K., Merolla, P. A., Arthur, J. V., Cassidy, A. S., Appuswamy, R., Andreopoulos, A., Berg, D. J., McKinstry, J. L., Melano, T., Barch, D. R., di Nolfo, C., Datta, P., Amir, A., Taba, B., Flickner, M. D., and Modha, D. S. (2016a). Convolutional Networks for Fast, Energy-Efficient Neuromorphic Computing. *Proc. of the National Academy of Sciences*.

Esser, S. K., Merolla, P. A., Arthur, J. V., Cassidy, A. S., Appuswamy, R., Andreopoulos, A., Berg, D. J., McKinstry, J. L., Melano, T., Barch, D. R., et al. (2016b). Convolutional networks for fast, energy-efficient neuromorphic computing. *arXiv preprint arXiv:1603.08270*.

Faisal, A. A., Selen, L. P., and Wolpert, D. M. (2008). Noise in the nervous system. *Nature Reviews Neuroscience*, 9(4):292–303.

Farries, M. A. and Fairhall, A. L. (2007). Reinforcement learning with modulated spike timing–dependent synaptic plasticity. *Journal of Neurophysiology*, 98(6):3648–3665.

Fates, N. A. and Morvan, M. (2004). An experimental study of robustness to asynchronism for elementary cellular automata. *arXiv preprint nlin/0402016*.

Fernando, C. and Sojakka, S. (2003). Pattern recognition in a bucket. In *European Conference on Artificial Life*, pages 588–597. Springer.

Fiers, M. A. A., Van Vaerenbergh, T., Wyffels, F., Verstraeten, D., Schrauwen, B., Dambre, J., and Bienstman, P. (2014). Nanophotonic reservoir computing with photonic crystal cavities to generate periodic patterns. *IEEE Transactions on Neural Networks and Learning systems*, 25(2):344–355.

Florian, R. V. (2007). Reinforcement learning through modulation of spike-timing-dependent synaptic plasticity. *Neural Computation*, 19(6):1468–1502.

Florian, R. V. (2012). The chronotron: a neuron that learns to fire temporally precise spike patterns. *Plos One*, 7(8):e40233.

Fontana, W. (1991). Algorithmic chemistry: A model for functional self-organization. *Artificial Life II*, pages 159–202.

Fradkov, A. L. and Evans, R. J. (2005). Control of chaos: methods and applications in engineering. *Annual Reviews in Control*, 29(1):33–56.

Freeman, W. J. (1987). Simulation of chaotic EEG patterns with a dynamic model of the olfactory system. *Biological Cybernetics*, 56:139–150.

Frémaux, N., Sprekeler, H., and Gerstner, W. (2013). Reinforcement learning using a continuous time actor-critic framework with spiking neurons. *PLoS Comput Biol*, 9(4):e1003024.

Fuchs, J.-J. (2004). On sparse representations in arbitrary redundant bases. *IEEE Transactions on Information Theory*, 50(6):1341–1344.

Gai, Y., Doiron, B., and Rinzel, J. (2010). Slope-based stochastic resonance: how noise enables phasic neurons to encode slow signals. *PLoS Comput Biol*, 6(6):e1000825.

Gaines, B. R. et al. (1969). Stochastic computing systems. *Advances in Information Systems Science*, 2(2):37–172.

Gale, E. (2017). Memristors in unconventional computing: How a biomimetic circuit element can be used to do bioinspired computation. In *Advances in Unconventional Computing*, pages 497–542. Springer.

Garrett, S. M. (2005). How do we evaluate artificial immune systems? *Evolutionary Computation*, 13(2):145–177.

Gerstner, W. (2001a). Coding properties of spiking neurons: reverse and cross-correlations. *Neural Networks*, 14(6):599–610.

Gerstner, W. (2001b). A framework for spiking neuron models: The spike response model. *Handbook of Biological Physics*, 4:469–516.

Gerstner, W., Kempter, R., van Hemmen, J. L., and Wagner, H. (1996). A neuronal learning rule for sub-millisecond temporal coding. *Nature*, 383(LCN-ARTICLE-1996-002):76–78.

Gerstner, W., Kistler, W. M., Naud, R., and Paninski, L. (2014). *Neuronal dynamics: From single neurons to networks and models of cognition*. Cambridge University Press.

Gerstner, W., Ritz, R., and Van Hemmen, J. L. (1993). Why spikes? hebbian learning and retrieval of time-resolved excitation patterns. *Biological Cybernetics*, 69(5-6):503–515.

Gibson, M. J., Keedwell, E. C., and Savić, D. A. (2015). An investigation of the efficient implementation of cellular automata on multi-core cpu and gpu hardware. *Journal of Parallel and Distributed Computing*, 77:11–25.

Glackin, C., Maguire, L., McDaid, L., and Sayers, H. (2011). Receptive field optimisation and supervision of a fuzzy spiking neural network. *Neural Networks*, 24(3):247–256.

Goldbeter, A. (1997). *Biochemical oscillations and cellular rhythms: the molecular bases of periodic and chaotic behaviour*. Cambridge university press.

Goudarzi, A., Lakin, M. R., and Stefanovic, D. (2013). DNA reservoir computing: A novel molecular computing approach. In *International Workshop on DNA-Based Computers*, pages 76–89. Springer.

Greensmith, J. and Aickelin, U. (2009). Artificial dendritic cells: multi-faceted perspectives. In *Human-Centric Information Processing Through Granular Modelling*, pages 375–395. Springer.

Grüning, A. and Bohte, S. M. (2014). Spiking neural networks: Principles and challenges. In *ESANN*.

Grüning, A. and Sporea, I. (2012). Supervised learning of logical operations in layered spiking neural networks with spike train encoding. *Neural Processing Letters*, 36(2):117–134.

Gütig, R. and Sompolinsky, H. (2006). The tempotron: a neuron that learns spike timing-based decisions. *Nature Neuroscience*, 9(3):420–428.

Habenschuss, S., Jonke, Z., and Maass, W. (2013). Stochastic computations in cortical microcircuit models. *PLoS Computational Biology*, 9(11):e1003311.

Han, J. (2009). From pid to active disturbance rejection control. *IEEE Transactions on Industrial Electronics*, 56(3):900–906.

Han, J. and Orshansky, M. (2013). Approximate computing: An emerging paradigm for energy-efficient design. In *Proceedings of 18th IEEE European Test Symposium (ETS)*, pages 1–6.

Harmer, P. K., Williams, P. D., Gunsch, G. H., and Lamont, G. B. (2002). An artificial immune system architecture for computer security applications. *IEEE transactions on evolutionary computation*, 6(3):252–280.

Harth, E., Csermely, T., Beek, B., and Lindsay, R. (1970). Brain functions and neural dynamics. *Journal of Theoretical Biology*, 26(1):93–120.

Hicke, K., Escalona-Morán, M. A., Brunner, D., Soriano, M. C., Fischer, I., and Mirasso, C. R. (2013). Information processing using transient dynamics of semiconductor lasers subject to delayed feedback. *IEEE Journal of Selected Topics in Quantum Electronics*, 19(4):1501610.

Hindmarsh, J. and Rose, R. (1984). A model of neuronal bursting using three coupled first order differential equations. *Proceedings of the Royal Society of London B: Biological Sciences*, 221(1222):87–102.

Hodgkin, A. L. and Huxley, A. F. (1952). A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of Physiology*, 117(4):500.

Hofmeyr, S. A. and Forrest, S. (2000). Architecture for an artificial immune system. *Evolutionary Computation*, 8(4):443–473.

Hoinville, T. and Henaff, P. (2004). Evolving plastic neural controllers stabilized by homeostatic mechanisms for adaptation to a perturbation. In *Proceedings of the 9th International Conference on the Simulation and Synthesis of Living Systems*, pages 81–87.

Holden, A. V., Markus, M., and Othmer, H. (2013). *Nonlinear wave processes in excitable media*, volume 244. Springer.

Hosoya, T., Baccus, S. A., and Meister, M. (2005). Dynamic predictive coding by the retina. *Nature*, 436(7047):71–77.

Hunt, J. E. and Cooke, D. E. (1996). Learning using an artificial immune system. *Journal of Network and Computer Applications*, 19(2):189–212.

Hussain, S. and Basu, A. (2016). Morphological learning in multicompartment neuron model with binary synapses. *Proceedings - IEEE International Symposium on Circuits and Systems*, 2016-July(3):2527–2530.

Hussain, S., Liu, S.-C., and Basu, A. (2014). Improved margin multi-class classification using dendritic neurons with morphological learning. In *2014 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 2640–2643. IEEE.

Hussain, S., Liu, S.-C., and Basu, A. (2015). Hardware-Amenable Structural Learning for Spike-Based Pattern Classification Using a Simple Model of Active Dendrites. *Neural Computation*, 27(4):845–897.

Hutton, T. J. (2002). Evolvable self-replicating molecules in an artificial chemistry. *Artificial Life*, 8(4):341–356.

Hutton, T. J. (2007). Evolvable self-reproducing cells in a two-dimensional artificial chemistry. *Artificial Life*, 13(1):11–30.

Ibarz, B., Casado, J. M., and Sanjuán, M. A. (2011). Map-based models in neuronal dynamics. *Physics Reports*, 501(1):1–74.

Iizuka, H., Ando, H., and Maeda, T. (2013). Extended homeostatic adaptation model with metabolic causation in plasticity mechanismtoward constructing a dynamic neural network model for mental imagery. *Adaptive Behavior*, page 1059712313488426.

Iizuka, H. and Di Paolo, E. A. (2007). Toward spinozist robotics: Exploring the minimal dynamics of behavioral preference. *Adaptive Behavior*, 15(4):359–376.

Iizuka, H. and Di Paolo, E. A. (2008). Extended homeostatic adaptation: Improving the link between internal and behavioural stability. In *International Conference on Simulation of Adaptive Behavior*, pages 1–11. Springer.

Indiveri, G. and Liu, S.-C. (2015). Memory and information processing in neuromorphic systems. *Proceedings of the IEEE*, 103(8):1379–1397.

Itoh, M. and Chua, L. O. (2009). Memristor cellular automata and memristor discrete-time cellular neural networks. *International Journal of Bifurcation and Chaos*, 19(11):3605–3656.

Iu, H. H.-C., Yu, D., Fitch, A. L., Sreeram, V., and Chen, H. (2011). Controlling chaos in a memristor based circuit using a twin-t notch filter. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 58(6):1337–1344.

Izhikevich, E. M. (2000). Neural excitability, spiking and bursting. *International Journal of Bifurcation and Chaos*, 10(06):1171–1266.

Izhikevich, E. M. (2004). Which model to use for cortical spiking neurons? *IEEE Transactions on Neural Networks*, 15(5):1063–1070.

Izhikevich, E. M. (2006). Polychronization: computation with spikes. *Neural Computation*, 18(2):245–282.

Izhikevich, E. M. et al. (2003). Simple model of spiking neurons. *IEEE Transactions on Neural etworks*, 14(6):1569–1572.

Jaeger, H. (2001). The echo state approach to analysing and training recurrent neural networks-with an erratum note. *Bonn, Germany: German National Research Center for Information Technology GMD Technical Report*, 148(34):13.

Jaeger, H. and Haas, H. (2004). Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication. *Science*, 304(5667):78–80.

Jakubowski, M. H., Steiglitz, K., and Squier, R. (2002). Computing with solitons: a review and prospectus. In *Collision-Based Computing*, pages 277–297. Springer.

Jakubowski, M. H., Steiglitz, K., and Squier, R. K. (1996). When can solitons compute? *Complex Systems*, 10(1):1–22.

Jeong, H., Tombor, B., Albert, R., Oltvai, Z. N., and Barabási, A.-L. (2000). The large-scale organization of metabolic networks. *Nature*, 407(6804):651–654.

Jin, D. Z. and Seung, H. S. (2002). Fast computation with spikes in a recurrent neural network. *Physical Review E*, 65(5):051922.

Jordehi, A. R. (2015). A chaotic artificial immune system optimisation algorithm for solving global continuous optimisation problems. *Neural Computing and Applications*, 26(4):827–833.

Kampa, B. M., Letzkus, J. J., and Stuart, G. J. (2007). Dendritic mechanisms controlling spike-timing-dependent synaptic plasticity. *Trends in Neurosciences*, 30(9):456–463.

Kappel, D., Habenschuss, S., Legenstein, R., and Maass, W. (2015). Network plasticity as bayesian inference. *PLoS Comput Biol*, 11(11):e1004485.

Kappel, D., Nessler, B., and Maass, W. (2014). Stdp installs in winner-take-all circuits an online approximation to hidden markov model learning. *PLoS Comput Biol*, 10(3):e1003511.

Kari, L. and Rozenberg, G. (2008). The many facets of natural computing. *Communications of the ACM*, 51(10):72–83.

Kasabov, N., Dhoble, K., Nuntalid, N., and Indiveri, G. (2013). Dynamic evolving spiking neural networks for on-line spatio-and spectro-temporal pattern recognition. *Neural Networks*, 41:188–201.

Katada, N. and Nishimura, H. (2009). Stochastic resonance in recurrent neural network with hopfield-type memory. *Neural Processing Letters*, 30(2):145–154.

Kayser, C., Montemurro, M. A., Logothetis, N. K., and Panzeri, S. (2009). Spike-phase coding boosts and stabilizes information carried by spatial and temporal spike patterns. *Neuron*, 61(4):597–608.

Kazem, A., Sharifi, E., Hussain, F. K., Saberi, M., and Hussain, O. K. (2013). Support vector regression with chaos-based firefly algorithm for stock market price forecasting. *Applied Soft Computing*, 13(2):947–958.

Kepecs, A. and Lisman, J. (2003). Information encoding and computation with spikes and bursts. *Network: Computation in neural systems*, 14(1):103–118.

Kephart, J. O. et al. (1994). A biologically inspired immune system for computers. In *Artificial Life IV: proceedings of the fourth international workshop on the synthesis and simulation of living systems*, pages 130–139.

Kim, M. and Smaragdis, P. (2015). Bitwise Neural Networks. *International Conference on Machine Learning (ICML) Workshop on Resource-Efficient Machine Learning*, 37.

Kistler, W. M. and Van Hemmen, J. L. (2000). Modeling synaptic plasticity in conjunction with the timing of pre-and postsynaptic action potentials. *Neural Computation*, 12(2):385–405.

Kitano, H. (2002a). Computational systems biology. *Nature*, 420(6912):206–210.

Kitano, H. (2002b). Systems biology: a brief overview. *Science*, 295(5560):1662–1664.

Kitano, H. (2004). Biological robustness. *Nature Reviews Genetics*, 5(11):826–837.

Kitano, H. (2007). Towards a theory of biological robustness. *Molecular Systems Biology*, 3(1):137.

Kohonen, T. (1988). An introduction to neural computing. *Neural Networks*, 1(1):3–16.

Kovačević, J. and Chebira, A. (2008). An introduction to frames. *Foundations and Trends in Signal Processing*, 2(1):1–94.

Ku, C.-C. and Lee, K. Y. (1995). Diagonal recurrent neural networks for dynamic systems control. *IEEE Transactions on Neural Networks*, 6(1):144–156.

Laje, R. and Buonomano, D. V. (2013). Robust timing and motor patterns by taming chaos in recurrent neural networks. *Nature Neuroscience*, 16(7):925–933.

Langton, C. G. (1986). Studying artificial life with cellular automata. *Physica D: Nonlinear Phenomena*, 22(1-3):120–149.

Langton, C. G. (1990). Computation at the edge of chaos: phase transitions and emergent computation. *Physica D: Nonlinear Phenomena*, 42(1):12–37.

Langton, C. G. (1997). *Artificial life: An overview*. Mit Press.

Langton, C. G. et al. (1989). *Artificial life*. Addison-Wesley Publishing Company Redwood City, CA.

Larger, L., Soriano, M. C., Brunner, D., Appeltant, L., Gutiérrez, J. M., Pesquera, L., Mirasso, C. R., and Fischer, I. (2012). Photonic information processing beyond turing: an optoelectronic implementation of reservoir computing. *Optics Express*, 20(3):3241–3249.

Layeghi, H., Arjmand, M. T., Salarieh, H., and Alasty, A. (2008). Stabilizing periodic orbits of chaotic systems using fuzzy adaptive sliding mode control. *Chaos, Solitons & Fractals*, 37(4):1125–1135.

Lazar, A., Pipa, G., and Triesch, J. (2007). Fading memory and time series prediction in recurrent networks with different forms of plasticity. *Neural Networks*, 20(3):312–322.

Lazar, A., Pipa, G., and Triesch, J. (2009). SORN: a self-organizing recurrent neural network. *Frontiers in Computational Neuroscience*, 3:article 23.

Lazar, A. A. and Tóth, L. T. (2003). Time encoding and perfect recovery of bandlimited signals. In *ICASSP (6)*, pages 709–712.

Le, D.-H. and Kwon, Y.-K. (2013). A coherent feedforward loop design principle to sustain robustness of biological networks. *Bioinformatics*, 29(5):630–637.

Lee, J. H., Delbruck, T., and Pfeiffer, M. (2016). Training deep spiking neural networks using backpropagation. *Frontiers in Neuroscience*, 10.

Legenstein, R., Naeger, C., and Maass, W. (2005). What can a neuron learn with spike-timing-dependent plasticity? *Neural Computation*, 17(11):2337–2382.

Legenstein, R., Pecevski, D., and Maass, W. (2008). A learning theory for reward-modulated spike-timing-dependent plasticity with application to biofeedback. *PLoS Comput Biol*, 4(10):e1000180.

Letzkus, J. J., Kampa, B. M., and Stuart, G. J. (2006). Learning rules for spike timing-dependent plasticity depend on dendritic synapse location. *The Journal of Neuroscience*, 26(41):10420–10429.

Li, D., Han, M., and Wang, J. (2012). Chaotic time series prediction based on a novel robust echo state network. *IEEE Transactions on Neural Networks and Learning Systems*, 23(5):787–799.

Liu, Y. and Passino, K. (2002). Biomimicry of social foraging bacteria for distributed optimization: models, principles, and emergent behaviors. *Journal of Optimization Theory and Applications*, 115(3):603–628.

Lukoševičius, M. and Jaeger, H. (2009). Reservoir computing approaches to recurrent neural network training. *Computer Science Review*, 3(3):127–149.

Lungu, I.-A., Riehle, A., Nawrot, M. P., and Schmuker, M. (2017). Predicting voluntary movements from motor cortical activity with neuromorphic hardware. *IBM J. Res. & Dev.*, 61(2-3):5:1 – 5:12.

Ma, W. J., Beck, J. M., Latham, P. E., and Pouget, A. (2006). Bayesian inference with probabilistic population codes. *Nature Neuroscience*, 9(11):1432–1438.

Maass, W. (1997). Networks of spiking neurons: the third generation of neural network models. *Neural Networks*, 10(9):1659–1671.

Maass, W. (2003). Computation with spiking neurons. *The handbook of brain theory and neural networks*, pages 1080–1083.

Maass, W., Natschläger, T., and Markram, H. (2002). Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural Computation*, 14(11):2531–2560.

MacLennan, B. J. (2004). Natural computation and non-turing models of computation. *Theoretical Computer Science*, 317(1-3):115–145.

Major, G., Larkum, M. E., and Schiller, J. (2013). Active properties of neocortical pyramidal neuron dendrites. *Annual Review of Neuroscience*, 36:1–24.

Marchese, F. M. (2002). A directional diffusion algorithm on cellular automata for robot path-planning. *Future Generation Computer Systems*, 18(7):983–994.

Marder, E. and Goaillard, J.-M. (2006). Variability, compensation and homeostasis in neuron and network function. *Nature Reviews Neuroscience*, 7(7):563–574.

Marder, E. and Prinz, A. A. (2002). Modeling stability in neuron and network function: the role of activity in homeostasis. *Bioessays*, 24(12):1145–1154.

Markram, H., Lübke, J., Frotscher, M., and Sakmann, B. (1997). Regulation of synaptic efficacy by coincidence of postsynaptic aps and epsps. *Science*, 275(5297):213–215.

Martinenghi, R., Rybalko, S., Jacquot, M., Chembo, Y. K., and Larger, L. (2012). Photonic nonlinear transient computing with multiple-delay wavelength dynamics. *Physical Review Letters*, 108(24):244101.

Masquelier, T., Guyonneau, R., and Thorpe, S. J. (2009). Competitive stdp-based spike pattern learning. *Neural Computation*, 21(5):1259–1276.

Matsugu, M., Mori, K., Ishii, M., and Mitarai, Y. (2002). Convolutional spiking neural network model for robust face detection. In *Neural Information Processing, 2002. ICONIP'02. Proceedings of the 9th International Conference on*, volume 2, pages 660–664. IEEE.

Medvedev, G. S. (2005). Reduction of a model of an excitable cell to a one-dimensional map. *Physica D: Nonlinear Phenomena*, 202(1):37–59.

Mehta, M., Lee, A., and Wilson, M. (2002). Role of experience and oscillations in transforming a rate code into a temporal code. *Nature*, 417(6890):741–746.

Merolla, P., Appuswamy, R., Arthur, J., Esser, S. K., and Modha, D. (2016). Deep neural networks are robust to weight binarization and other non-linear distortions. *ArXiv e-prints*.

Miner, D. and Triesch, J. (2016). Plasticity-driven self-organization under topological constraints accounts for non-random features of cortical synaptic wiring. *PLoS Comput Biol*, 12(2):e1004759.

Mitchell, M., Crutchfield, J. P., and Hraber, P. T. (1994). Evolving cellular automata to perform computations: Mechanisms and impediments. *Physica D: Nonlinear Phenomena*, 75(1):361–391.

Mitchell, M. et al. (1996). Computation in cellular automata: A selected review. *Nonstandard Computation*, pages 95–140.

Mitra, S., Fusi, S., and Indiveri, G. (2009). Real-time classification of complex patterns using spike-based learning in neuromorphic VLSI. *IEEE Trans. on Biomedical Circuits and Systems*, 3(1):32–42.

Morrison, A., Aertsen, A., and Diesmann, M. (2007). Spike-timing-dependent plasticity in balanced random networks. *Neural Computation*, 19(6):1437–1467.

Mostafa, H., Müller, L. K., and Indiveri, G. (2015). An event-based architecture for solving constraint satisfaction problems. *Nature Communications*, 6:8941.

Motter, A. E. (2004). Cascade control and defense in complex networks. *Physical Review Letters*, 93(9):098701.

Muller, S. D., Marchetto, J., Airaghi, S., and Kournoutsakos, P. (2002). Optimization based on bacterial chemotaxis. *IEEE transactions on Evolutionary Computation*, 6(1):16–29.

Munakata, T., Sinha, S., and Ditto, W. L. (2002). Chaos computing: implementation of fundamental logical gates by chaotic elements. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 49(11):1629–1633.

Muthuswamy, B. (2010). Implementing memristor based chaotic circuits. *International Journal of Bifurcation and Chaos*, 20(05):1335–1350.

Nagumo, J. and Sato, S. (1972). On a response characteristic of a mathematical neuron model. *Kybernetik*, 10(3):155–164.

Natschläger, T. and Ruf, B. (2009). Spatial and temporal pattern analysis via spiking neurons. *Network: Computation in Neural Systems*.

Neftci, E., Binas, J., Rutishauser, U., Chicca, E., Indiveri, G., and Douglas, R. J. (2013). Synthesizing cognition in neuromorphic electronic systems. *PNAS*, pages E3468–E3476.

Neftci, E. O., Pedroni, B. U., Joshi, S., Al-Shedivat, M., and Cauwenberghs, G. (2016). Stochastic synapses enable efficient brain-inspired learning machines. *Frontiers in Neuroscience*, 10:Article 241.

Nessler, B., Pfeiffer, M., Buesing, L., and Maass, W. (2013). Bayesian computation emerges in generic cortical microcircuits through spike-timing-dependent plasticity. *PLoS Comput Biol*, 9(4):e1003037.

Nicolis, G., Prigogine, I., et al. (1977). *Self-organization in nonequilibrium systems*, volume 191977. Wiley, New York.

Obst, O., Trinchi, A., Hardin, S. G., Chadwick, M., Cole, I., Muster, T. H., Hoschke, N., Ostry, D., Price, D., Pham, K. N., et al. (2013). Nano-scale reservoir computing. *Nano Communication Networks*, 4(4):189–196.

Ono, N. and Ikegami, T. (2001). Artificial chemistry: Computational studies on the emergence of self-reproducing units. In *European Conference on Artificial Life*, pages 186–195. Springer.

Oster, M., Douglas, R., and Liu, S.-C. (2009). Computation with spikes in a winner-take-all network. *Neural Computation*, 21(9):2437–2465.

Ott, E., Grebogi, C., and Yorke, J. A. (1990). Controlling chaos. *Physical Review Letters*, 64(11):1196.

Oya, T., Asai, T., and Amemiya, Y. (2007). A single-electron reaction-diffusion device for computation of a voronoi diagram. In Teuscher, C. and Adamatzky, A., editors, *2005 Workshop on Unconventional Computing: From Cellular Automata to Wetware*, pages 13–26. Luniver press.

OConnor, P., Neil, D., Liu, S.-C., Delbruck, T., and Pfeiffer, M. (2015). Real-time classification and sensor fusion with a spiking deep belief network. *Neuromorphic Engineering Systems and Applications*, page 61.

Panzeri, S., Petersen, R. S., Schultz, S. R., Lebedev, M., and Diamond, M. E. (2001). The role of spike timing in the coding of stimulus location in rat somatosensory cortex. *Neuron*, 29(3):769–777.

Paquot, Y., Duport, F., Smerieri, A., Dambre, J., Schrauwen, B., Haelterman, M., and Massar, S. (2012). Optoelectronic reservoir computing. *Scientific Reports*, 2.

Park, B. S., Yoo, S. J., Park, J. B., and Choi, Y. H. (2009). Adaptive neural sliding mode control of nonholonomic wheeled mobile robots with model uncertainty. *IEEE Transactions on Control Systems Technology*, 17(1):207–214.

Păun, G. (2000). Computing with membranes. *Journal of Computer and System Sciences*, 61(1):108–143.

Paun, G. (2006). Introduction to membrane computing.

Păun, G. (2012). Membrane computing. *Handbook of Natural Computing*, pages 1355–1377.

Paun, G. (2012). *Membrane computing: an introduction*. Springer Science & Business Media.

Păun, G. and Rozenberg, G. (2002). A guide to membrane computing. *Theoretical Computer Science*, 287(1):73–100.

Pecevski, D., Buesing, L., and Maass, W. (2011). Probabilistic inference in general graphical models through sampling in stochastic networks of spiking neurons. *PLoS Comput Biol*, 7(12):e1002294.

Pecevski, D. and Maass, W. (2016). Learning probabilistic inference through stdp. *eneuro*, pages ENEURO–0048.

Pei, Y. (2014). Chaotic evolution: fusion of chaotic ergodicity and evolutionary iteration for optimization. *Natural Computing*, 13(1):79–96.

Perrinet, L., Delorme, A., Samuelides, M., and Thorpe, S. J. (2001). Networks of integrate-and-fire neuron using rank order coding a: How to implement spike time dependent hebbian plasticity. *Neurocomputing*, 38:817–822.

Piccinini, G. (2015). *Physical computation: A mechanistic account*. OUP Oxford.

Poirazi, P. and Mel, B. W. (2001). Impact of active dendrites and structural plasticity on the memory capacity of neural tissue. *Neuron*, 29(3):779–796.

Polyak, B. and Tempo, R. (2001). Probabilistic robust design with linear quadratic regulators. *Systems & Control Letters*, 43(5):343–353.

Polyak, B. T. (2005). Stabilizing chaos with predictive control. *Automation and Remote Control*, 66(11):1791–1804.

Poppelbaum, W., Afuso, C., and Esch, J. (1967). Stochastic computing elements and systems. In *Proceedings of the November 14-16, 1967, fall joint computer conference*, pages 635–644. ACM.

Pyragas, K. (1992). Continuous control of chaos by self-controlling feedback. *Physics Letters A*, 170(6):421–428.

Qiao, N., Indiveri, G., and Bartolozzi, C. (2016). Automatic gain control of ultra-low leakage synaptic scaling homeostatic plasticity circuits. In *Proc. Biomedical Circuits and Systems Conference (BioCAS)*, pages 156–159. IEEE.

Ramakrishnan, S., Wunderlich, R., Hasler, J., and George, S. (2013). Neuron array with plastic synapses and programmable dendrites. *IEEE Transactions on Biomedical Circuits and Systems*, 7(5):631–642.

Rand, D., Steiglitz, K., and Prucnal, P. R. (2004). Signal standardization in collision-based soliton computing. *IJUC*, 1(1):31–45.

Randles, M., Lamb, D., Odat, E., and Taleb-Bendiab, A. (2011). Distributed redundancy and robustness in complex systems. *Journal of Computer and System Sciences*, 77(2):293–304.

Rao, R. P. and Sejnowski, T. J. (2001). Spike-timing-dependent hebbian plasticity as temporal difference learning. *Neural Computation*, 13(10):2221–2237.

Rastegari, M., Ordonez, V., Redmon, J., and Farhadi, A. (2016). Xnor-net: Imagenet classification using binary convolutional neural networks. *arXiv preprint*, pages 1–17.

Reinker, S., Puil, E., and Miura, R. M. (2004). Membrane resonance and stochastic resonance modulate firing patterns of thalamocortical neurons. *Journal of Computational Neuroscience*, 16(1):15–25.

Remme, M. W. and Wadman, W. J. (2012). Homeostatic scaling of excitability in recurrent neural networks. *PLoS Comput Biol*, 8(5):e1002494.

Rombouts, J. and Bohte, S. M. (2010). Fractionally predictive spiking neurons. In *Advances in Neural Information Processing Systems*, pages 253–261.

Roy, S., Banerjee, A., and Basu, A. (2014). Liquid state machine with dendritically enhanced readout for low-power, neuromorphic vlsi implementations. *IEEE Transactions on Biomedical Circuits and Systems*, 8(5):681–695.

Roy, S. and Basu, A. (2016). An Online Unsupervised Structural Plasticity Algorithm for Spiking Neural Networks. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–11.

Roy, S., San, P. P., Hussain, S., Wei, L. W., and Basu, A. (2015). Learning spike time codes through morphological learning with binary synapses. *IEEE Transactions on Neural Networks and Learning Systems*, 27(7):1572–1577.

Rulkov, N. F. (2001). Regularization of synchronized chaotic bursts. *Physical Review Letters*, 86(1):183.

Rulkov, N. F. (2002). Modeling of spiking-bursting neural behavior using two-dimensional map. *Physical Review E*, 65(4):041922.

Rutishauser, U., Douglas, R. J., and Slotine, J.-J. (2011). Collective stability of networks of winner-take-all circuits. *Neural Computation*, 23(3):735–773.

Rutishauser, U., Slotine, J.-J., and Douglas, R. J. (2012). Competition through selective inhibitory synchrony. *Neural Computation*, 24(8):2033–2052.

Sayama, H. (2009). Swarm chemistry. *Artificial Life*, 15(1):105–114.

Schiller, U. D. and Steil, J. J. (2005). Analyzing the weight dynamics of recurrent learning algorithms. *Neurocomputing*, 63:5–23.

Schmidhuber, J., Wierstra, D., Gagliolo, M., and Gomez, F. (2007). Training recurrent networks by evolino. *Neural Computation*, 19(3):757–779.

Schrauwen, B., D'Haene, M., Verstraeten, D., and Van Campenhout, J. (2007). Compact hardware for real-time speech recognition using a liquid state machine. In *Neural Networks, 2007. IJCNN 2007. International Joint Conference on*, pages 1097–1102. IEEE.

Schrauwen, B., DHaene, M., Verstraeten, D., and Van Campenhout, J. (2008a). Compact hardware liquid state machines on fpga for real-time speech recognition. *Neural Networks*, 21(2):511–523.

Schrauwen, B., Wardermann, M., Verstraeten, D., Steil, J. J., and Stroobandt, D. (2008b). Improving reservoirs using intrinsic plasticity. *Neurocomputing*, 71(7):1159–1171.

Schürmann, F., Meier, K., and Schemmel, J. (2004). Edge of chaos computation in mixed-mode vlsi-a hard liquid. In *Advances in neural information processing systems*, pages 1201–1208.

Senn, W., Markram, H., and Tsodyks, M. (2001). An algorithm for modifying neurotransmitter release probability based on pre-and postsynaptic spike timing. *Neural Computation*, 13(1):35–67.

Seung, H. S. (2003). Learning in spiking neural networks by reinforcement of stochastic synaptic transmission. *Neuron*, 40(6):1063–1073.

Sheik, S., Chicca, E., and Indiveri, G. (2012). Exploiting device mismatch in neuromorphic VLSI systems to implement axonal delays. In *Proc. WCCI 2012 IEEE World Congress on Computational Intelligence*, pages 1940–1945. IEEE.

Sheik, S., Pfeiffer, M., Stefanini, F., and Indiveri, G. (2013). Spatio-temporal spike pattern classification in neuromorphic systems. In Lepora et al., N. F., editor, *iving Machines 2013,*, volume 8064 of *LNAI*, pages 262–273. Springer.

Sheridan, P. M., Du, C., and Lu, W. D. (2015). Feature Extraction Using Memristor Networks. *IEEE Transactions on Neural Networks and Learning Systems*, 27(11):2327–2336.

Shilnikov, A. L. and Rulkov, N. F. (2004). Subthreshold oscillations in a map-based neuron model. *Physics Letters A*, 328(2):177–184.

Shinbrot, T., Grebogi, C., Ott, E., and Yorke, J. A. (1993). Using small perturbations to control chaos. *Nature*, 363(6428):411–417.

Siettos, C. and Starke, J. (2016). Multiscale modeling of brain dynamics: from single neurons and networks to mathematical tools. *Wiley Interdisciplinary Reviews: Systems Biology and Medicine*, 8(5):438–458.

Silverman, E. and Ikegami, T. (2011). Robustness in artificial life. *International Journal of Bio-Inspired Computation 9*, 3(3):179–186.

Sinha, S. and Ditto, W. L. (1999). Computing with distributed chaos. *Physical Review E*, 60(1):363.

Sjöström, P. J., Turrigiano, G. G., and Nelson, S. B. (2001). Rate, timing, and cooperativity jointly determine cortical synaptic plasticity. *Neuron*, 32(6):1149–1164.

Snyder, D., Goudarzi, A., and Teuscher, C. (2013). Computational capabilities of random automata networks for reservoir computing. *Physical Review E*, 87(4):042808.

Song, Y., Chen, Z., and Yuan, Z. (2007). New chaotic pso-based neural network predictive control for nonlinear process. *IEEE Transactions on Neural Networks*, 18(2):595–601.

Soudry, D., Hubara, I., and Meir, R. (2014). Expectation Backpropagation: parameter-free training of multilayer neural networks with real and discrete weights. *Neural Information Processing Systems 2014*, 2(1):1–9.

Spiess, R., Diehl, P. U., George, R., and Cook, M. (2016). Structural Plasticity Denoises Responses and Improves Learning Speed. *Frontiers in Computational Neuroscience*, 10(September):93.

Sporea, I. and Grüning, A. (2013). Supervised learning in multilayer spiking neural networks. *Neural Computation*, 25(2):473–509.

Starck, J.-L., Elad, M., and Donoho, D. (2004). Redundant multiscale transforms and their application for morphological component separation. *Advances in Imaging and Electron Physics*, 132:287–348.

Steels, L. (1993). The artificial life roots of artificial intelligence. *Artificial Life*, 1(1_2):75–110.

Steimer, A. and Douglas, R. (2013). Spike-based probabilistic inference in analog graphical models using interspike-interval coding. *Neural Computation*, 25(9):2303–2354.

Stelling, J. (2004). Mathematical models in microbial systems biology. *Current Opinion in Microbiology*, 7(5):513–518.

Stelling, J., Gilles, E. D., and Doyle, F. J. (2004a). Robustness properties of circadian clock architectures. *Proceedings of the National Academy of Sciences of the United States of America*, 101(36):13210–13215.

Stelling, J., Sauer, U., Szallasi, Z., Doyle, F. J., and Doyle, J. (2004b). Robustness of cellular functions. *Cell*, 118(6):675–685.

Stengel, R. F. (1991). Intelligent failure-tolerant control. *IEEE Control Systems*, 11(4):14–23.

Stengel, R. F. and Ryan, L. (1991). Stochastic robustness of linear time-invariant control systems. *IEEE Transactions on Automatic Control*, 36(1):82–87.

Stockwell, R. G. (2007). A basis for efficient representation of the s-transform. *Digital Signal Processing*, 17(1):371–393.

Stoy, K. (2006). Using cellular automata and gradients to control self-reconfiguration. *Robotics and Autonomous Systems*, 54(2):135–141.

Stromatias, E. and Marsland, J. S. (2015). Supervised learning in spiking neural networks with limited precision: Snn/lp. In *2015 International Joint Conference on Neural Networks (IJCNN)*, pages 1–7. IEEE.

Stromatias, E., Neil, D., Pfeiffer, M., Galluppi, F., Furber, S. B., and Liu, S.-C. (2015a). Robustness of spiking deep belief networks to noise and reduced bit precision of neuro-inspired hardware platforms. *Frontiers in Neuroscience*, 9.

Stromatias, E., Neil, D., Pfeiffer, M., Galluppi, F., Furber, S. B., and Liu, S. C. (2015b). Robustness of spiking Deep Belief Networks to noise and reduced bit precision of neuro-inspired hardware platforms. *Frontiers in Neuroscience*, 9(JUN):1–14.

Subrata, R. and Zomaya, A. Y. (2003). A comparison of three artificial life techniques for reporting cell planning in mobile computing. *IEEE Transactions on Parallel and Distributed Systems*, 14(2):142–153.

Subrata, R., Zomaya, A. Y., and Landfeldt, B. (2007). Artificial life techniques for load balancing in computational grids. *Journal of Computer and System Sciences*, 73(8):1176–1190.

Suzuki, Y., Takabayashi, J., and Tanaka, H. (2002). Investigation of tritrophic interactions in an ecosystem using abstract chemistry. *Artificial Life and Robotics*, 6(3):129–132.

Talbi, E.-G. (2009). *Metaheuristics: from design to implementation*, volume 74. John Wiley & Sons.

Tan, K. C., Goh, C. K., Mamun, A., and Ei, E. (2008). An evolutionary artificial immune system for multi-objective optimization. *European Journal of Operational Research*, 187(2):371–392.

Teuscher, C. (2014). Unconventional computing catechism. *Frontiers in Robotics and AI*, 1:10.

Thalmeier, D., Uhlmann, M., Kappen, H. J., and Memmesheimer, R.-M. (2015). Learning universal computations with spikes. *arXiv preprint arXiv:1505.07866*.

Thorpe, S., Delorme, A., and Van Rullen, R. (2001). Spike-based strategies for rapid processing. *Neural Networks*, 14(6):715–725.

Thorpe, S. and Gautrais, J. (1998). Rank order coding. In *Computational Neuroscience*, pages 113–118. Springer.

Timmis, J., Andrews, P., and Hart, E. (2010). On artificial immune systems and swarm intelligence. *Swarm Intelligence*, 4(4):247–273.

Timmis, J., Hone, A., Stibor, T., and Clark, E. (2008). Theoretical advances in artificial immune systems. *Theoretical Computer Science*, 403(1):11–32.

Timmis, J., Neal, M., and Hunt, J. (2000). An artificial immune system for data analysis. *Biosystems*, 55(1):143–150.

Tiňo, P. and Mills, A. J. (2006). Learning beyond finite memory in recurrent networks of spiking neurons. *Neural Computation*, 18(3):591–613.

Touboul, J. (2008). Bifurcation analysis of a general class of nonlinear integrate-and-fire neurons. *SIAM Journal on Applied Mathematics*, 68(4):1045–1079.

Touboul, J. and Brette, R. (2009). Spiking dynamics of bidimensional integrate-and-fire neurons. *SIAM Journal on Applied Dynamical Systems*, 8(4):1462–1506.

Toyoizumi, T., Pfister, J.-P., Aihara, K., and Gerstner, W. (2004). Spike-timing dependent plasticity and mutual information maximization for a spiking neuron model. In *Advances in neural information processing systems*, pages 1409–1416.

Triefenbach, F., Jalalvand, A., Schrauwen, B., and Martens, J.-P. (2010). Phoneme recognition with large hierarchical reservoirs. In *Advances in neural information processing systems*, pages 2307–2315.

Tsai, W.-Y. and et al (2017). Always-on speech recognition using TrueNorth, a reconfigurable, neurosynaptic processor. *IEEE Trans. on Computers*, 66(6):996–1007.

Turrigiano, G. (2012). Homeostatic synaptic plasticity: local and global mechanisms for stabilizing neuronal function. *Cold Spring Harbor perspectives in biology*, 4(1):a005736.

Turrigiano, G. G. (1999). Homeostatic plasticity in neuronal networks: the more things change, the more they stay the same. *Trends in Neurosciences*, 22(5):221–227.

Turrigiano, G. G. (2008). The self-tuning neuron: synaptic scaling of excitatory synapses. *Cell*, 135(3):422–435.

Turrigiano, G. G. and Nelson, S. B. (2004). Homeostatic plasticity in the developing nervous system. *Nature Reviews Neuroscience*, 5(2):97–107.

Tzionas, P. G., Thanailakis, A., and Tsalides, P. G. (1997). Collision-free path planning for a diamond-shaped robot using two-dimensional cellular automata. *IEEE Transactions on Robotics and Automation*, 13(2):237–250.

Urbanczik, R. and Senn, W. (2009a). A gradient learning rule for the tempotron. *Neural Computation*, 21(2):340–352.

Urbanczik, R. and Senn, W. (2009b). Reinforcement learning in populations of spiking neurons. *Nat Neurosci*, 12(3):250–252.

Utkin, V. I. (1993). Sliding mode control design principles and applications to electric drives. *IEEE Transactions on Industrial Electronics*, 40(1):23–36.

Vaidyanathan, S. (2015). Global chaos synchronization of chemical chaotic reactors via novel sliding mode control method. *Parameters*, 1:4.

Van Rossum, M. C., Bi, G. Q., and Turrigiano, G. G. (2000). Stable hebbian learning from spike timing-dependent plasticity. *The Journal of Neuroscience*, 20(23):8812–8821.

Van Rullen, R. and Thorpe, S. J. (2001). Rate coding versus temporal order coding: what the retinal ganglion cells tell the visual cortex. *Neural Computation*, 13(6):1255–1283.

Vandoorne, K., Dambre, J., Verstraeten, D., Schrauwen, B., and Bienstman, P. (2011). Parallel reservoir computing using optical amplifiers. *IEEE Transactions on Neural Networks*, 22(9):1469–1481.

Vandoorne, K., Dierckx, W., Schrauwen, B., Verstraeten, D., Baets, R., Bienstman, P., and Van Campenhout, J. (2008). Toward optical signal processing using photonic reservoir computing. *Optics Express*, 16(15):11182–11192.

Vandoorne, K., Fiers, M., Verstraeten, D., Schrauwen, B., Dambre, J., and Bienstman, P. (2010). Photonic reservoir computing: a new approach to optical information processing. In *Photonics North 2010*, pages 775022–775022. International Society for Optics and Photonics.

Vandoorne, K., Mechet, P., Van Vaerenbergh, T., Fiers, M., Morthier, G., Verstraeten, D., Schrauwen, B., Dambre, J., and Bienstman, P. (2014). Experimental demonstration of reservoir computing on a silicon photonics chip. *Nature Communications*, 5.

Vasilaki, E., Frémaux, N., Urbanczik, R., Senn, W., and Gerstner, W. (2009). Spike-based reinforcement learning in continuous state and action space: when policy gradient methods fail. *PLoS Comput Biol*, 5(12):e1000586.

Vidal, C. (2008). The future of scientific simulations: from artificial life to artificial cosmogenesis. *arXiv preprint arXiv:0803.1087*.

Vincent-Lamarre, P., Lajoie, G., and Thivierge, J.-P. (2016). Driving reservoir models with oscillations: a solution to the extreme structural sensitivity of chaotic networks. *Journal of Computational Neuroscience*, pages 1–18.

Vlachos, I., Deniz, T., Aertsen, A., and Kumar, A. (2016). Recovery of dynamics and function in spiking neural networks with closed-loop control. *PLoS Comput Biol*, 12(2):e1004720.

Vogels, T. P. and Abbott, L. (2009). Gating multiple signals through detailed balance of excitation and inhibition in spiking networks. *Nature Neuroscience*, 12(4):483–491.

Von Neumann, J. (1956). Probabilistic logics and the synthesis of reliable organisms from unreliable components. *Automata Studies*, 34:43–98.

von Neumann, J. (1966). *Theory of self-replicating automata (edited and completed by A. W. Burks)*. University of Illinois Press.

Wade, J. J., McDaid, L. J., Santos, J. A., and Sayers, H. M. (2010). Swat: a spiking neural network training algorithm for classification problems. *IEEE Transactions on Neural Networks*, 21(11):1817–1830.

Wang, S.-J. and Zhou, C. (2012). Hierarchical modular structure enhances the robustness of self-organized criticality in neural networks. *New Journal of Physics*, 14(2):023005.

Watkins, A., Timmis, J., and Boggess, L. (2004). Artificial immune recognition system (airs): An immune-inspired supervised learning algorithm. *Genetic Programming and Evolvable Machines*, 5(3):291–317.

Whitacre, J. and Bender, A. (2010). Degeneracy: a design principle for achieving robustness and evolvability. *Journal of Theoretical Biology*, 263(1):143–153.

Whitacre, J. M. (2010). Degeneracy: a link between evolvability, robustness and complexity in biological systems. *Theoretical Biology and Medical Modelling*, 7(1):1.

Wiesenfeld, K., Moss, F., et al. (1995). Stochastic resonance and the benefits of noise: from ice ages to crayfish and squids. *Nature*, 373(6509):33–36.

Williams, H. (2004). Homeostatic plasticity in recurrent neural networks. In *From Animals to Animats 8: Proceedings of the 8th International Conference on the Simulation of Adaptive Behavior*, pages 344–353.

Williams, H. and Noble, J. (2007). Homeostatic plasticity improves signal propagation in continuous-time recurrent neural networks. *Biosystems*, 87(2):252–259.

Wolfram, S. (1983). Statistical mechanics of cellular automata. *Reviews of Modern Physics*, 55(3):601.

Wolfram, S. (1984). Cellular automata as models of complexity. *Nature*, 311(5985):419–424.

Wolfram, S. et al. (1986). *Theory and applications of cellular automata*. World scientific Singapore.

Xie, X. and Seung, H. S. (2004). Learning in neural networks by reinforcement of irregular spiking. *Physical Review E*, 69(4):041909.

Yamamoto, L., Collet, P., and Banzhaf, W. (2013). Artificial chemistries on gpu. In *Massively Parallel Evolutionary Computation on GPGPUs*, pages 389–419. Springer.

Yao, P. and et al (2017). Face classification using electronic synapses. *Nature Communications*, page 8:15199. DOI: 10.1038/ncomms15199.

Yeger-Lotem, E., Sattath, S., Kashtan, N., Itzkovitz, S., Milo, R., Pinter, R. Y., Alon, U., and Margalit, H. (2004). Network motifs in integrated cellular networks of transcription–regulation and protein–protein interaction. *Proceedings of the National Academy of Sciences of the United States of America*, 101(16):5934–5939.

Yi, T.-M., Huang, Y., Simon, M. I., and Doyle, J. (2000). Robust perfect adaptation in bacterial chemotaxis through integral feedback control. *Proceedings of the National Academy of Sciences*, 97(9):4649–4653.

Zenke, F., Agnes, E. J., and Gerstner, W. (2015). Diverse synaptic plasticity mechanisms orchestrated to form and retrieve memories in spiking neural networks. *Nature Communications*, 6.

Zenke, F., Hennequin, G., and Gerstner, W. (2013). Synaptic plasticity in neural networks needs homeostasis with a fast rate detector. *PLoS Comput Biol*, 9(11):e1003330.

Zhang, G., Rong, H., Neri, F., and Pérez-Jiménez, M. J. (2014). An optimization spiking neural p system for approximately solving combinatorial optimization problems. *International Journal of Neural Systems*, 24(05):1440006.

Zhang, G. and Shen, Y. (2013). New algebraic criteria for synchronization stability of chaotic memristive neural networks with time-varying delays. *IEEE Transactions on Neural Networks and Learning Systems*, 24(10):1701–1707.

Zhao, L., Park, K., and Lai, Y.-C. (2004). Attack vulnerability of scale-free networks due to cascading breakdown. *Physical Review E*, 70(3):035101.

Zhou, Q. and Liao, X. (2012). Collision-based flexible image encryption algorithm. *Journal of Systems and Software*, 85(2):400–407.

Ziegler, J. and Banzhaf, W. (2001). Evolving control metabolisms for a robot. *Artificial Life*, 7(2):171–190.