Natural Computing Series

Series Editors

Thomas Bäck, Natural Computing Group–LIACS, Leiden University, Leiden, The Netherlands

Lila Kari, School of Computer Science, University of Waterloo, Waterloo, ON, Canada

More information about this series at http://www.springer.com/series/4190

Kohei Nakajima · Ingo Fischer
Editors

Reservoir Computing

Theory, Physical Implementations, and Applications

Springer

Editors
Kohei Nakajima
University of Tokyo
Tokyo, Japan

Ingo Fischer
Consejo Superior de Investigaciones
Científicas, IFISC (UIB-CSIC)
Palma, Spain

ISSN 1619-7127
Natural Computing Series
ISBN 978-981-13-1686-9 ISBN 978-981-13-1687-6 (eBook)
https://doi.org/10.1007/978-981-13-1687-6

This Springer imprint is published by the registered company Springer Nature Singapore Pte Ltd.
The registered company address is: 152 Beach Road, #21-01/04 Gateway East, Singapore 189721, Singapore

Foreword

Reservoir computing seems simple but is difficult, feels new but is old, opens horizons, and is brutally limiting. I will do my best in this foreword to leave the reader with many questions—to be answered, or maybe not, in the many chapters of this richly filled book.

The basic principle of reservoir computing (RC) is simple. Given: a *training* input signal $\mathbf{u}^{train}(t)$ paired with a desired target output signal $\mathbf{y}^{train}(t)$. Wanted: a filter (transducer) $\mathcal{F}$ which, when fed with input $\mathbf{u}^{train}(t)$, generates an output signal $\hat{\mathbf{y}}^{train}(t)$ which comes close to the target $\mathbf{y}^{train}(t)$. Approach: *Step 1.* Prepare a high-dimensional dynamical system $X(t)$, the *reservoir*, which can be driven by input $\mathbf{u}^{train}(t)$ and in which many state variables $x_i(t)$ (where $i = 1, \ldots, N$) can be observed and recorded. *Step 2.* Drive this system with input $\mathbf{u}^{train}(t)$ and record the corresponding reservoir-internal response signals $x_i^{train}(t)$. *Step 3.* Find (train, learn, and estimate) a *readout functionF* which maps every recorded state vector $(x_1^{train}(t), \ldots, x_N^{train}(t))$ to an output $\hat{\mathbf{y}}^{train}(t)$ which approximates the training targets $\mathbf{y}^{train}(t)$. Finding such a readout F often boils down to a simple linear regression. Exploitation: feed new input signals $\mathbf{u}(t)$ to the reservoir, observe the reservoir-internal state vectors $\mathbf{x}(t) = (x_1(t), \ldots, x_N(t))$, and compute the output signal $\hat{\mathbf{y}}(t) = F(\mathbf{x}(t))$.

This basic scheme is very versatile. One can solve temporal input-output tasks for time series prediction, dynamical pattern generation, classification and segmentation, control, de-noising and channel equalization, rare event monitoring, and many more. One can apply RC to obtain practical engineering solutions in signal processing and control, robotics, communication technologies, machine learning, and AI; one can call upon RC models as an explanatory principle in theoretical neuroscience; and in mathematics, one can use RC as an entry point to identify and analyze a number of interesting phenomena in high-dimensional dynamical systems. But most importantly, one can in principle use *any* kind of nonlinear, high-dimensional dynamical system for the reservoir $X(t)$, regardless of whether it is an experimental probe of a freshly engineered nanomaterial, a quantum dot preparation, a replica of an octopus arm made from soft plastic and suspended in water, or a digital simulation of a neural network all to be found in the scintillating collection of reservoirs that the reader will find in this book.

But... the closer one becomes involved with RC, the more difficult it gets, and if the one to embrace it in full contact, it gets almost impossibly difficult. Reservoirs are high-dimensional, input-driven, nonlinear, and often stochastic dynamical systems. A full theory of reservoir dynamics would be a full theory of everything that evolves in time. Only fragmentary insights into the unbounded phenomenal richness in general dynamical systems are currently available in mathematics, theoretical physics and biology, or the general complex systems sciences. Compared to what we *could* know about, observe in, and utilize from reservoir dynamics, we currently *do* know, see, and use almost nothing. It is easy to program a recurrent neural network with 100 neurons on a digital computer, declare it a reservoir, apply the basic RC scheme on a simple modeling task, exclaim "it works!" and call it good. This is how students worldwide get hooked on RC. However, when one gets pushed out of the comfort zone of the dozen or so ever-repeated "benchmark" tasks that pervade the RC literature, then reservoirs turn into feral beasts that take an enormous amount of patience and experience to tame. This applies, e.g., when the data are noisy or incomplete, have outliers or are nonstationary, have a wandering baseline or variable amplitude, are high-dimensional, or have multiple spatial or temporal scales, when stability conditions have to be guaranteed, the task demands continual online learning, or when the input data consist of rare events spiking out of a zero baseline. Moreover, problems arise, when there are many possible options for input and output signal re-coding (there always are), when one's computer allows only fast experimentation with small reservoirs, but one wants to extrapolate to large ones, or when one wants to automate the readout training. The promise of RC, one *need not* train the reservoir, turns into a problem: one *cannot* train the reservoir. There is an unlimited variability in task specifics, and there is an infinity of dynamical behaviors in candidate reservoir systems. In a haystack of possibilities one must find a reservoir whose native dynamics matches the demands of the task at hand. After two decades of RC research, we only have the faintest inklings of how to match reservoir dynamics with task dynamics. Many of my students choose a reservoir computing theme for their graduating thesis. I dare say that, when after much trial and error, they ultimately arrive at the point where "it works!" they don't understand *why* it works—and neither do I.

Many contemporary RC papers that I read or review introduce their subject still with "Reservoir computing is a new approach to train neural networks ...". Well... RC may be called "new" compared to Newton's and Leibniz's calculus, but by the standards of the fast-paced innovation cycles in machine learning it is rather old. The basic RC principle has been discovered and re-discovered many times, and I continue to become aware of earlier and earlier "first" sightings. This is how it goes with most ideas that are elementary and useful.

It is not customary to cite references in a foreword, but I take this as an opportunity to give due credit to RC pathfinders. The earliest perception of the RC principle that I am aware is Kirby and Day (1990), a 1-page conference abstract that was subsequently worked out by Kevin Kirby in a paper where he gives what I consider the first concise and comprehensive account of the RC principle (Kirby 1991), with the readout from the reservoir (which he called *context reverberation subsystem*) trained by the perceptron learning algorithm. The problem of finding a "good" reservoir is clearly identified, and a sentence in the Conclusion section reads like prophesy: "This may encourage molecular electronic hardware implementations." Both papers remained entirely unnoticed (the single Google Scholar cite that I saw when I queried this in 2017 was a self-citation). In the same year 1991, Lambert Schomaker, in Chapter 7 of his Ph.D. thesis (Schomaker 1991) (separately published in Schomaker (1992)), described how a target output signal can be obtained by learning a linear readout from a random ensemble of spiking neural oscillators. I got to know about this work by an unlikely chance: after I was appointed at the University of Groningen in 2019, Lambert became my direct senior manager and he told me about his Ph.D. thesis in a casual conversation. I wonder how many other casual conversations with other senior colleagues worldwide would bring up similar surprises. Both Schomaker and Kirby refer back to earlier precursor ideas in their texts—clues for further studies in scientific archeology. The next independent discovery of RC that I know about occurred in cognitive neuroscience. Peter F. Dominey described a multi-module (human) brain circuit for sequence generation which included a simplified model of prefrontal cortex as a reservoir from which trainable readouts send information to the caudate nucleus (Dominey 1995). Up to the present day, and in close collaboration with other RC researchers, Dominey has been continuing to work out elaborate neuro-cognitive architectures with an RC core both for neuroscience modeling and for robotic/human-maching interaction applications. His chapter in this book gives a summary of a 25-year-long personal research mission.

The current RC literature mostly localizes the origin of RC in the propositions of *liquid state machines* by Wolfgang Maass and my *echo state networks* (Maass et al. 2002; Jaeger 2001). Wolfgang and I got to know of each other at the 2001 EU Advanced Course in Computational Neuroscience at the International Center for Theoretical Physics in Trieste, Italy, August 2001, where to our mutual surprise we found our own ideas almost identically reflected in the respective other's. We started to collaborate, soon joined by Benjamin Schrauwen who coined the term "reservoir computing" (or was it his brilliant Ph.D. student David Verstraeten? the first published paper where this term was used seems to be Verstraeten et al. (2005)). Benjamin rapidly built up an enormously productive RC research group at the University of Gent even before he was awarded his Ph.D. degree. I think several factors came together why reservoir computing took off only then. First, the three of us teamed up instead of defending proprietary RC islands. Second, for the first time the mathematical preconditions that make reservoirs functional were clearly spelled out through the *fading memory* and *separation property* in Wolfgang's models and the *echo state property* and an analysis of a reservoir's *memory capacity* in my work. Mathematical formulae gave authority to a wild-looking idea. Third, "it worked" really well in many demo tasks that met the taste and demands of the time—while training recurrent neural networks with other then-existing learning algorithms was difficult, unstable, and slow. The deep learning revolution superseded RC only toward the end of the 2010s when the intricacies of gradient-descent training of recurrent neural networks finally became mastered. Reservoir computing research receded into a niche for a few years.

But RC research re-awakened and sprouted out again from this niche when RC principles were adopted first in the field of optical computing (see, e.g., chapters by Kanno et al. and Dambre et al.) and swiftly also in other domains of *physical reservoir computing* (surveyed in the chapter by Dale et al.). Most chapters in this book are a testimonial to the refreshing new thrust that RC has given to the wider fields of *unconventional/in-materio/natural/...* computing (I have a private list of about 15 different namings that have been branded in the last four decades or so). Materials scientists and non-digital device engineers from the most diverse makings continue to discover RC for themselves. As long as RC continues to be freshly discovered by colleagues in widening circles, there is still truth in when they say, in the introductory passages of articles, that RC is "... a new approach ...".

Ah, before I forget: there is one little technical thing that I want to point out to everyone who uses RC for the first time. So many neural-network-based RC papers state in their methods section that the spectral radius (largest absolute eigenvalue) of the network weight matrix should be less than unity to ensure the echo state property, a necessary condition to make RC work. This is a myth. A spectral radius $SR < 1$ is neither sufficient nor necessary for the echo state property (Yildiz et al. 2012), and a value much larger than 1 often gives the best performance. Please don't perpetuate this myth in your work! And while I am at it: another myth is that reservoirs work best when they are tuned to operate "at the edge of chaos", or "close to criticality". First, it's a misnomer, because the edge in question here is the edge of the echo state property, not the edge of chaos. If a reservoir slides across this edge, it doesn't necessarily (even not typically) enter a chaotic regime. Second, reservoirs are input-driven systems, and mathematicians still haven't entirely agreed on how to define chaos in input-driven systems. Finally, reservoirs "close to criticality" work well only for a certain class of learning tasks—the sort of tasks which are invariably re-iterated in articles on this subject—but reservoirs far on the stable side of that edge work much better for many other tasks. Cramer et al. (2020) point a spotlight on this affair. I really can't understand why this myth remains recited so often, given the massive counter-evidence from so many practical applications where carefully optimized reservoirs come out sitting safe and far away from this edge.

RC has the elegance of simplicity, which may be explanation enough why it inspires researchers in many fields. Of course, there are more substantial reasons why RC keeps blossoming, for instance, because it connects the neuro- with the computing sciences in stronger than purely metaphorical ways; or that it opens new doors for theoretical analyses of high-dimensional dynamical systems; or that materials scientists today really don't have many alternatives to make their unconventional substrates "compute".

But one should be aware that the powers of RC as a stand-alone carrier of "learning" or "computing" are decisively limited. Biological brains may be using RC in some places and some ways—it is unlikely that they don't because evolution will find and keep any trick that works—but brains use many other dynamical mechanisms and structuring principles and information encoding procedures as well, and I don't think we have an idea yet even of *how* many. From my perspective of machine learning, AI and theory of computing, the strongest weakness (what a nice oxymoron) of pure RC is its inherent blindness to hierarchical multi-scale compositionality of data structures, processes. and architectures. Here, I understand compositionality in a strong sense which includes *bidirectional* interaction between higher modules or layers and their sub-modules or lower layers. An example are planning architectures for autonomous agents where higher planning modules generate longer-term plans that "call" lower sub-plan modules in a *top-down* direction, and stay informed about execution progress in a *bottom-up* direction of communication from the sub-modules. Another example are the Boltzmann machine or Friston's free-energy models of neural processing where higher layers send statistical biases to lower layers, and are informed about conditional feature distributions from below. In computer science, the object-oriented programming paradigm is the very manifestation of bidirectionally effective compositionality. The top-down actions can be interpreted in a variety of ways, for instance, as attention control, predictive context settings, or read/write signals in working memory systems. Such top-down modulations are essential for full-fledged cognitive information processing, but such bidirectionally effective hierarchical cognitive architectures cannot be realized by RC alone. Additional structures and algorithms are needed to coordinate intermodule communication (as in my attempt in Jaeger (2007) to design a hierarchical RC learning architecture that can discover temporal features on several timescales), or additional teacher signals for the individual modules must be created (as in Pascanu and Jaeger (2011) where we trained a kind of parser for visual text input that had a nested grammatical structure), or additional control mechanisms must be installed to modulate the reservoir dynamics "from above" (like *conceptors* (Jaeger 2017)). It is one of the strongest strengths of today's deep learning networks that such multi-directionally organized architectures, for instance, *neural Turing machines*, can be trained by "end-to-end" gradient descent, where the requisite local training signals are automatically generated. This said, I emphasize that RC can positively be applied with much benefit in certain multi-scale learning tasks using uni-directionally coupled stacks of reservoirs where lower, typically faster reservoirs connect upwards to higher, typically slower reservoirs. Gallicchio and Micheli (in this volume) survey such architectures, which they call *deep RC* systems.

RC research keeps advancing in many directions. I want to conclude with my personal favorite challenges for the next evolutionary steps in RC research:

- Coordinate RC modules in complex learning and information processing systems with the aid of additional mechanisms—this theme is also highlighted in the Conclusion of Dambre et al.'s chapter in this book.
- In physical RC, find ways to realize the readout and its training directly in the non-digital material substrate, instead of delegating it to a digital host computer.
- Leverage the infinite dimensionality of spatially extended nonlinear excitable media, extending the readout combination of a finite number of reservoir signals to an infinite-dimensional integration, convolution, or field transformation. In physical RC, one might envision spatially continuous two-layer substrates where

the bottom layer acts as a reservoir and the top layer would function as a continuous version of what today are the readout weight matrices.

- Find effective ways to cope with the unpleasant properties of physical reservoirs, such as device mismatch, parameter drift, temperature sensitivity, low numerical precision and stochasticity, and partial observability. Physical reservoirs may only become practically useful when appropriate auto-calibration or homeostatic regulatory mechanisms are realized in combination with numerically robust and swiftly self-adapting readout processes.
- Rigorously analyze which abstract dynamical characteristics of input and output data and task specifications should be reflected in which characteristics of reservoir dynamics. Currently available insights are mostly distilled from experimental studies of timescale profiles or frequency spectra in input data and provide no comprehensive guides for optimizing reservoir designs.

Many chapters in this collection include historical summaries of major RC research strands, and all tell enticing stories about what today's achievements are and are not. This book lets us see where we stand and invites us to imagine where we can go further. Being a veteran of the field, I feel enormously grateful for the massive labor of editors and authors to plant this landmark after 30 years of a voyage that will continue to feel fresh and young.

Herbert Jaeger
Bernoulli Institute for Mathematics
Computer Science and Artificial Intelligence
Cognitive Systems and Materials Center
(CogniGron)
University of Groningen
Groningen, The Netherlands

References

B. Cramer, D. Stöckel, M. Kreft, M. Wibral, J. Schemmel, K. Meier, V. Priesemann. Control of criticality and computation in spiking neuromorphic networks with plasticity. Nat. Commun. **11**, 2853, 2020

D. Verstraeten, B. Schrauwen, D. Stroobandt. Reservoir computing with stochastic bitstream neurons. In *Proceedings of the 16th Annual Prorisc Workshop*, pp. 454–459, 2005

H. Jaeger, The "echo state" approach to analysing and training recurrent neural networks. GMD Report 148, GMD - German Nat. Res. Instit. Comp. Sci. 2001. https://www.ai.rug.nl/minds/uploads/EchoStatesTechRepErratum.pdf.

H. Jaeger. Discovering multiscale dynamical features with hierarchical echo state networks. Technical report 10, School of Engineering and Science, Jacobs University, 2007. URL https://www.ai.rug.nl/minds/uploads/hierarchicalesn_techrep10.pdf.

H. Jaeger, Using conceptors to manage neural long-term memories for temporal patterns. *JMRL*, **18**, 1–43, 2017

I. B. Yildiz, H. Jaeger, S. J. Kiebel, Re-visiting the echo state property. Neural Netw. **35**, 1–20, 2012

K. Kirby, Context dynamics in neural sequential learning. In *Proceedings Florida AI Research Symposium (FLAIRS)*, pp. 66–70, 1991

K. G. Kirby, N. Day, The neurodynamics of context reverberation learning. In *Proceedings Annual International Conference of the IEEE Engineering in Medicine and Biology Society, Vol. 12 4*, pp. 1781–1782, 1990

L. R. B. Schomaker, *Simulation and Recognition of Handwriting Movements: A vertical approach to modeling human motor behavior*. Phd thesis, Nijmeegs Instituut voor Cognitie-onderzoek en Informatietechnologie, Nijmegen, 1991. URL https://repository.ubn.ru.nl/handle/2066/113914

L. R. B. Schomaker, A neural oscillator-network model of temporal pattern generation. Hum. Mov. Sci. **11**, 181–192, 1992

P. F. Dominey, Complex sensory-motor sequence learning based on recurrent state representation and reinforcement learning. Biol. Cybern. **73**, 265–274, 1995

R. Pascanu, H. Jaeger. A neurodynamical model for working memory. Neural Netw. **240**(2), 199–207, 2011

W. Maass, T. Natschläger, H. Markram. Real-time computing without stable states: A new framework for neural computation based on perturbations. Neural Comput.**140**(11), 2531–2560, 2002

Preface

Reservoir Computing: Theory, Physical Implementations, and Applications is the first comprehensive book about reservoir computing (RC). RC was introduced in the early 2000s as a unified framework for recurrent neural network (RNN) training; it included a number of seminal models, such as echo-state networks (Jaeger 2001) and liquid state machines (Maass et al. 2002). Although RC originated in computational neuroscience and machine learning, in recent years, the use of RC has spread, and it has been introduced into a wide variety of fields, including nonlinear dynamical systems theory, physics, material science, biological science, and robotics (Fig. 1). One of the major reasons for this increase in relevance of RC is its conceptual simplicity. RC capitalizes on the nonlinear responses of a high-dimensional dynamical system, referred to as the reservoir. By restricting the learning process to the readout layer, RC resolved the difficulties and instabilities of RNN training, which had conventionally been implemented using the method of gradient descent. Using

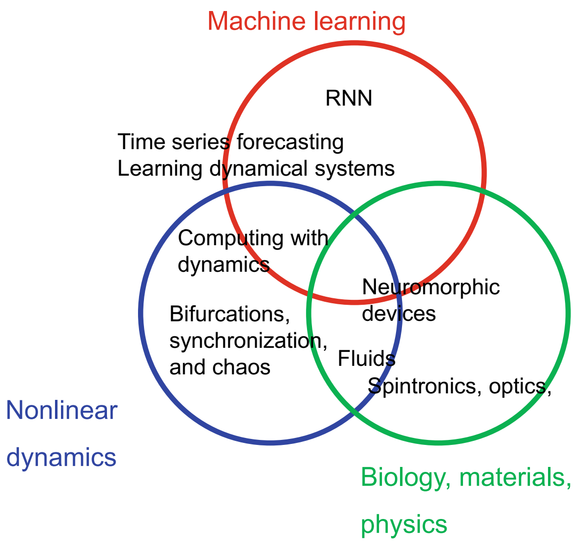

Fig. 1 Diagram summarizing how each field is connected through the concept of RC

gradient descent, all the network weights were trained, according to performance optimization needs and available training time. The particular RC concept allows us to exploit many "well-behaving" dynamical systems as reservoirs. This finding has resulted in important opportunities exploiting not only standard RNN but also many nonlinear dynamical systems and various physical dynamics found in nature as a computational resource.

Accordingly, recent RC trends and technologies exhibit two important directions. The first direction is to extend the framework of RC from the conventional RNN to a more abstract setup using the terms of nonlinear dynamical systems. With this broadened perspective, RC is not specific to the field of machine learning anymore but can be connected to a much wider class of systems. In particular, the connection and relationship between many technical terms developed in different fields and originating from different contexts have been revealed and bridged, which makes the RC technique accessible to various disciplines. For instance, the echo state property, which was originally proposed in the context of the echo-state networks, can also be related to generalized synchronization between the input stream and the reservoir dynamics. These bridges prove also effective and vital to the following second direction.

The second direction is the exploitation of physical dynamical systems as reservoirs; the framework for doing so is called physical reservoir computing (PRC). Because of the rapid development of computational technologies and sensing systems worldwide, novel schemes and devices are required to process massive amounts of data quickly in real time. In conventional computational architectures, due to the separation of the processing system and the memory system, there is a limit to the information processing speed, which is called a von Neumann bottleneck. This limit could be overcome using an approach inspired by biology or by using a dynamical-system-based implementation that realizes information processing and carries a memory of past input streams simultaneously; this is a typical non-von Neumann architecture. PRC is one of the main candidates for such architectures that researchers are currently focusing on. Many physical systems and materials have been already suggested and implemented as reservoir computing substrates. These systems include a wide range of physical systems exhibiting different spatiotemporal scales ranging from mechanical systems to optics, nanomaterials, spintronics, and quantum many-body systems. They are expected to be the substrates for next-generation neuromorphic devices that can process information natively at the edge according to the spatiotemporal scale, which is often termed edge computing. The variety of physical substrates provides a large diversity in the type of information processing that can be implemented. It is noticeable that this inspiration of PRC is, in fact, not a recent invention but has been around for a while since the genesis of RC approaches. Original attempts to implement PRC can be found in the ideas of the liquid computer (Natschläger et al. 2002) and the liquid brain (Fernando et al. 2003).

This book presents recent developments in the area of RC and is sub-structured into two major parts: theory and physical implementations. The book is a compilation of chapters contributed by different authors, who are leading experts in their respective fields. In detail, the book is structured as follows:

The first part (Part I) is devoted to theoretical aspects of RC. It starts with a wide perspective of aspects on how the real human brain processes information. In W. Singer's chapter, by comparing the recent system architecture of artificial intelligence and the real brain comprehensively, the important role of nonlinear dynamics in the cerebral cortex is discussed. In P. F. Dominey's chapter, it is argued that these dynamics generated in the cerebral cortex with structures of recurrency actually act as a reservoir. Subsequently, based on these properties of the real brain, A. Subramoney, F. Scherr, and W. Maass propose a novel architecture that can include meta-learning, called learning-to-learn, into the reservoir using plastic connections of weights. Deep architectures have also been introduced in the RC framework, and C. Gallicchio and A. Micheli provide a comprehensive overview of recent developments of deep reservoir computing. In the chapter by M. Inubushi, K. Yoshimura, Y. Ikeda, and Y. Nagasawa, the role of common-signal-induced synchronization on the information processing capability of the reservoir is discussed as a key to guaranteeing reproducible input-output relations. Finally, in the chapter by J. S. Pathak and E. Ott, recent progress on time series forecasting of large-scale spatiotemporal chaos introducing parallel spatial coupling in RC is presented, and the performance improvement is discussed in detail.

The second part (from Part II to Part VII) focuses on the physical implementations of RC, namely, PRC. M. Dale, J. F. Miller, S. Stepney, and M. Trefzer initiate the discussion on how to classify the appropriate physical substrates for computation in generic settings and propose a scheme and measures to systematically evaluate them. PRC is then introduced in the context of mechanical systems (Part III). H. Hauser has reviewed several case studies of PRC applications in robotics and discusses the importance of embodiment and the effectiveness of the approach for soft robotics. Guillaume Dion, Anouar Idrissi-El Oudrhiri, Bruno Barazani, Albert Tessier-Poirier, and Julien Sylvestre present PRC using MEMS.

Part IV begins by focusing on neuromorphic devices. F. Hadaeghi provides a systematic survey of neuromorphic electronic systems and their applications to RC and summarizes future challenges. In the chapter by S. Apostel, N. D. Haynes, E. Schöll, O. D'Huys, and D. J. Gauthier, field-programmable gate array implementations of an autonomous Boolean network for RC are demonstrated and analyzed in detail. R. Aguilera, H. O. Sillin, A. Z. Stieg, and J. K. Gimzewski present an atomic switch network as a substrate of RC implementations. The subsequent three chapters focus on spintronics approaches (Part V). M. Riou, J. Torrejon, and F. Abreu, et. al. present the recent development of neuromorphic applications for nanoscale spin-torque oscillators. T. Taniguchi, S. Tsunegi, and S. Miwa, et al. analyze the information processing capability (e.g., memory capacity) of a spin-torque oscillator as a reservoir, and H. Nomura, H. Kubota, and Y. Suzuki demonstrate an approach that uses a simple magnetic nano-dots array and discuss the possibility of implementing a larger scale array as a reservoir.

Part VI concentrates on photonic reservoir computing, which exploits optical systems as reservoirs. K. Kanno and A. Uchida demonstrate that the computational performance of a photonic reservoir can be improved by introducing a chaotic input mask signal, and they present an implementation of a miniature size photonic integrated circuit. J. Dambre, A. Katumba, C. Ma, S. Sackesyn, F. Laporte, M. Freiberger, and P. Bienstman provide a brief history of integrated photonic reservoirs and introduce recent approaches designed to increase the computational power of the system.

Part VII is devoted to PRC developments in the field of quantum machine learning. In recent years, remarkable progress has been made in quantum computation and the development of quantum computer technology is heating up worldwide. Simultaneously, noisy intermediate-scale quantum devices, which include a number of qubits with no error correction capability and their applications are receiving attention from many physicists. Part VII begins with the chapter by K. Fujii and K. Nakajima that introduces a framework for quantum reservoir computing (QRC) from the basics. This chapter also introduces several approaches, such as quantum extreme learning machine (QELM) and quantum circuit learning, and demonstrates emulation tasks of chaotic attractors based on QRC. The chapter by M. Negoro, K. Mitarai, K. Nakajima, and K. Fujii presents the first implementation of a quantum reservoir using nuclear magnetic resonance (NMR) ensemble systems and successfully demonstrates QELM. The authors also discuss a future scenario for implementing QRC using NMR ensemble systems.

Last but not the least, we note that the year 2020, in which this book was prepared, was a difficult and challenging year for mankind in general and for the people involved in this book in particular. Many problems arose in the face of COVID-19, and the processes involved in creating this book were significantly delayed. During this difficult time, all the chapter authors, the Springer editor, the collaborators, and our families have been incredibly supportive and patient. We would like to sincerely thank them all and acknowledge their role in making this publication possible. It is our greatest pleasure to bring out this exciting book into the world.

Tokyo, Japan — Kohei Nakajima
Palma, Spain — Ingo Fischer
November 2020

References

H. Jaeger. The "echo state" approach to analysing and training recurrent neural networks-with an erratum note. Bonn, Germany: German Nat. Res. Cent. Inf. Technol. GMD Tech. Rep. **148**(34), 13 (2001)

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

T. Natschläger, W. Maass, H. Markram, The "liquid computer": A novel strategy for real-time computing on time series. Spec. issue Found. Inf. Process. TELEMATIK, 8(ARTICLE), 39–43 (2002)

C. Fernando, S. Sojakka, Pattern Recognition in a Bucket. In: Banzhaf W., Ziegler J., Christaller T., Dittrich P., Kim J.T. (eds) Advances in Artificial Life. ECAL 2003. Lecture Notes in Computer Science, vol 2801. Springer, Berlin, Heidelberg (2003)

Contents