

# Online training for high-performance analogue readout layers in photonic reservoir computers

Piotr Antonik · Marc Haelterman · Serge Massar

**Abstract** *Introduction.* Reservoir Computing is a bio-inspired computing paradigm for processing time-dependent signals. The performance of its hardware implementation is comparable to state-of-the-art digital algorithms on a series of benchmark tasks. The major bottleneck of these implementation is the readout layer, based on slow offline post-processing. Few analogue solutions have been proposed, but all suffered from noticeable decrease in performance due to added complexity of the setup.

*Methods.* Here we propose the use of online training to solve these issues. We study the applicability of this method using numerical simulations of an experimentally feasible reservoir computer with an analogue readout layer. We also consider a nonlinear output layer, which would be very difficult to train with traditional methods.

*Results.* We show numerically that online learning allows to circumvent the added complexity of the analogue layer and obtain the same level of performance as with a digital layer.

*Conclusion.* This work paves the way to high-performance fully-analogue reservoir computers through the use of online training of the output layers.

**Keywords** Reservoir computing · Opto-electronics · Analogue readout · FPGA · Online training

---

P. Antonik, S. Massar  
Laboratoire d'Information Quantique  
Université libre de Bruxelles  
Avenue F. D. Roosevelt 50, CP 224  
Brussels, Belgium  
E-mail: pantonik@ulb.ac.be

M. Haelterman  
Service OPERA-Photonique  
Université libre de Bruxelles,  
Avenue F. D. Roosevelt 50, CP 194/5  
Brussels, Belgium

## 1 Introduction

Reservoir computing is a set of machine learning methods for designing and training artificial neural networks, introduced independently in [1] and [2]. The idea is that one can exploit the dynamics of a recurrent nonlinear network to process time series without training the network itself, but simply adding a general linear readout layer and only training the latter. This results in a system that is significantly easier to train (the learning is reduced to solving a system of linear equations, see [3]), yet powerful enough to match other algorithms on a series of benchmark tasks. Reservoir computing has been successfully applied to wireless channel equalisation and chaotic time series forecasting [1], phoneme recognition [4], image processing [5], handwriting recognition [6], audio classification [7] and won an international competition on prediction of future evolution of financial time series [8].

Reservoir computing allows efficient implementation of simplified recurrent neural networks in hardware, such as e.g. optical components. Optical computing has been investigated for decades as photons propagate faster than electrons, without generating heat or magnetic interference, and thus promise higher bandwidth than conventional computers [9]. Reservoir computing would thus allow to build high-speed and energy efficient photonic computational devices. Several important steps have been taken towards this goal with electronic [10], opto-electronic [11–13], all-optical [14–16] and integrated [17] experimental implementation reported since 2011.

The major drawback in these experiments is the absence of efficient readout mechanisms: the states of the neurons are collected and post-processed on a computer, severely reducing the processing speeds and thus limiting the applicability. An analog readout would resolve this issue, as suggested in [18]. This research di-

rection has already been investigated experimentally in [19–21], but all these implementations suffered from significant performance degradation due to the complex structure of the readout layer. Indeed the approach used in these works was to characterise with high accuracy the linear output layer, whereupon it was possible to compute offline the output weights. However it is virtually impossible to characterise each hardware component of the setup with sufficient level of accuracy. Furthermore the components in the output layer may have a slight nonlinear behaviour. It follows that this approach does not work satisfactorily, as is apparent from the performance degradation reported in [20].

In this work we address the above issues with the online learning approach. Online training has attracted much attention in the machine learning community because it allows to train the system gradually, as the input data becomes available. It can also easily cope with non-stationary input signal, whose characteristics change with time, as the online approach can keep the model updated according to variations in the input. Finally, in the case of hardware systems, online training can easily cope with drifts in the hardware, as the system will adapt to gradual changes in the hardware components [22, 23].

In the context of reservoir computing, the online training implements a gradient descent: it gradually changes the output layer to adapt to the task. More precisely the output layer is characterised by a series of parameters (the readout weights), and in online training these weights are adjusted in small increments, so that the output of the system gets closer to the target signal. We have previously applied this method to a hardware reservoir computer with a digital output layer in [24], where we illustrated how online learning could cope with non-stationary input signals, i.e. tasks that change with time.

The important point in the present context is that, compared to previously used offline methods, in online training based on gradient descent no assumption is necessary about how these weights contribute to the output signal. That is, it is not necessary to model the output layer. Furthermore, the transfer function of the readout layer could in principle be nonlinear. Here we show, using realistic numerical simulations, how these features could be highly advantageous for training hardware reservoir computers.

For concreteness, we will consider in simulations an opto-electronic reservoir computing setup based on a ring topology already extensively studied experimentally in [11, 12]. We add to this setup an analogue layer that is now trained online by an FPGA chip processing the simple gradient descent algorithm in real time, as

in [24]. The readout layer consists of a simple Resistor-Capacitor (RC) circuit (as in [19]), instead of a more complicated RLC circuit (consisting of a resistor  $R$ , an inductor  $L$  and a capacitor  $C$ ) that was used to increase the amplitude of the output signal in [20].

We investigate the performance of this setup through numerical simulations on two benchmark tasks and show that previously encountered difficulties are almost entirely alleviated by the online training approach. In other words, with a relatively simple analogue readout layer, trained online, and without any modelling of the underlying processes, we obtain results similar to those produced by a digital layer, trained offline. We also explore a special case with a nonlinear readout function and show that this complication doesn't decrease much the performance of the system. This work thus brings an interesting solution to an important problem in the hardware reservoir computing field.

The paper is structured as follows. In the Methods section, we introduce the basic principles of reservoir computing, online learning and the benchmark tasks used here, and then present the experimental opto-electronic reservoir computer, the analogue readout layer, and specify the major aspects of our numerical simulations. We then focus on the results of our investigations and conclude the paper with future perspectives.

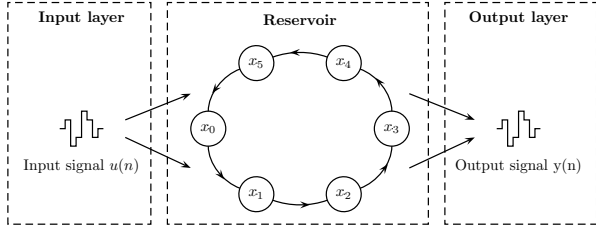
## 2 Methods

### 2.1 Reservoir Computing

A reservoir computer, schematised in figure 1, consists of a recurrent network of internal variables, usually called “nodes” or “neurons”, from its biological origins [3]. These  $N$  variables, denoted by  $x_i(n)$ , with  $i = 0, \dots, N - 1$ , evolve in discrete time  $n \in \mathbb{Z}$ , as follows

$$\begin{aligned} x_0(n+1) &= f(\alpha x_{N-1}(n) + \beta M_0 u(n)), \\ x_i(n+1) &= f(\alpha x_{i-1}(n) + \beta M_i u(n)), \end{aligned} \quad (1)$$

where  $f$  is a nonlinear function,  $u(n)$  is the input signal that is injected into the system,  $\alpha$  and  $\beta$  are feedback and input gains, respectively, used to adjust the dynamics of the system, and  $M_i$  is the input mask, drawn from a uniform distribution over the interval  $[-1, +1]$ , as in e.g. [25, 11, 14]. In our implementation, we use a sine function  $f = \sin(x)$  as nonlinearity and a ring topology [10, 25] to simplify the interconnection matrix of the network, so that only the first neighbour nodes are connected. Both choices are dictated by the proposed hardware setup of the opto-electronic reservoir. As will be discussed in Sec. 2.4, we use a light intensity modulator with a sine transfer function as the nonlinear node,



**Fig. 1** Schematic representation of a reservoir computer with  $N = 6$  nodes. In terms of artificial neural networks, its architecture is composed of a single input neuron (which receives the input signal  $u(n)$ ), one layer of hidden neurons and a single output neuron (which produces the output signal  $y(n)$ ). The configuration of neurons in the hidden layer can be arbitrary, but for ease of hardware implementation we use a ring-like topology.

and a delay system with time-multiplexing of the reservoir states. A detailed discussion of these experimental aspects can be found in the Supplementary Material of [11].

The reservoir computer produces an output signal  $y(n)$ , given by a linear combination of the states of its internal variables

$$y(n) = \sum_{i=0}^{N-1} w_i(n) x_i(n), \quad (2)$$

where  $w_i(n)$  are the readout weights, trained either offline (using standard linear regression methods), in which case they are time independent, or online, as described in the next section, in order to minimise a task-dependent error function, which will be introduced alongside the benchmark tasks further in this paper. As discussed below in detail, in hardware implementations of reservoir computing the readout layer can be more complex than Eq. (2).

## 2.2 Simple gradient descent algorithm

Contrary to offline, or batch learning, where the entire training dataset is used at once to compute the best readout weights  $w_i$ , online training approach handles the data sequentially in order to optimise the performance step by step. As discussed in Sec. 1, this allows the Reservoir Computer to be optimised without accurate knowledge of the underlying hardware, which is exactly what is required for an analogue readout layer. This approach can be realised with various algorithms, and in this work we chose to work with the simple gradient descent algorithm, for ease of implementation on the FPGA board.

The gradient, or steepest, descent method is an algorithm for finding a local minimum of a function using its gradient [26]. For the task considered here (see the

next section) we update the readout weights using the procedure given in [27]

$$w_i(n+1) = w_i(n) + \lambda (d(n) - y(n)) x_i(n), \quad (3)$$

where  $\lambda$  is the step size, used to control the learning rate, and  $d(n)$  is the task-dependent target signal (see sections 2.3.1 and 2.3.2). The origin of this procedure is that if the error at time  $n$  is given by  $(d(n) - y(n))^2$ , then the derivative of the error with respect to  $w_i$  gives  $(d(n) - y(n))x_i(n)$ , i.e. the right-hand side of Eq. (3). At high values of  $\lambda$ , the weights get close to the optimal values very quickly (in a few steps), but keep oscillating around these values. At low values, the weights converge slowly to the optimal values. In practice, we start with a high value  $\lambda = \lambda_0$ , and then gradually decrease it during the training phase until a minimum value  $\lambda_{min}$  is reached, according to the equation

$$\lambda(m+1) = \lambda_{min} + \gamma (\lambda(m) - \lambda_{min}), \quad (4)$$

with  $\lambda(0) = \lambda_0$  and  $m = \lfloor n/k \rfloor$ , where  $\gamma < 1$  is the decay rate and  $k$  is the update rate for the parameter  $\lambda$ . Previous work has shown that setting  $\lambda_0 = 0.4$ ,  $\lambda_{min} = 0$  and  $\gamma = 0.999$  is a reasonable choice for good performance [24]. The update rate  $k$  defines the convergence speed and the resulting error rate : higher  $k$  requires a longer training dataset but offers better results. More complex optimisation methods could be used here, such as simulated annealing [28] or stochastic gradient descent [29]. However, the above technique is very simple and provides sufficiently good results for this application.

## 2.3 Benchmark tasks

We tested our system on two benchmark tasks commonly used by the reservoir computing community: wireless channel equalisation and emulation of a 10-th order Nonlinear Auto Regressive Moving Average system (NARMA10). Note that, to the best of our knowledge, the latter has never been tested on an online-trained reservoir computer.

### 2.3.1 Wireless channel equalisation

The operating principle of this practically relevant task is the following. A sequence of symbols  $d(n)$  is transmitted through a wireless channel. The receiver records a sequence  $u(n)$ , which is a corrupted version of  $d(n)$ . The main sources of corruption are noise (thermal or electronic), multipath propagation, which leads to inter-symbol interference, and nonlinear distortion induced

by power amplifiers. The goal is to recover  $d(n)$  from  $u(n)$  [1].

The channel input signal  $d(n)$  contains 2-bit symbols with values picked randomly from  $\{-3, -1, 1, 3\}$ . The channel is modelled by a linear system with memory of length 10 [30]

$$\begin{aligned} q(n) = & 0.08d(n+2) - 0.12d(n+1) + d(n) \\ & + 0.18d(n-1) - 0.1d(n-2) + 0.091d(n-3) \\ & - 0.05d(n-4) + 0.04d(n-5) + 0.03d(n-6) \\ & + 0.01d(n-7), \end{aligned} \quad (5)$$

followed by an instantaneous memoryless nonlinearity

$$u(n) = q(n) + 0.036q^2(n) - 0.011q^3(n), \quad (6)$$

where  $u(n)$  is the channel output signal. The reservoir computer has to restore the clean signal  $d(n)$  from the distorted noisy signal  $u(n)$ . The performance is measured in terms of wrongly reconstructed symbols, called the Symbol Error Rate (SER).

### 2.3.2 NARMA10

The goal of this rather academic task is to emulate a 10-th order Nonlinear Auto Regressive Moving Average system. The input signal  $u(n)$  is drawn randomly from a uniform distribution over the interval  $[0, 0.5]$ . The target output  $d(n)$  is defined by the following equation

$$\begin{aligned} d(n+1) = & 0.3d(n) + 0.05d(n) \left( \sum_{i=0}^9 d(n-i) \right) \\ & + 1.5u(n-9)u(n) + 0.1. \end{aligned} \quad (7)$$

Since the reservoir doesn't produce  $d(n)$  exactly, its performance is measured in terms of an error metric. We use the Normalised Mean Square Error (NMSE), given by

$$\text{NMSE} = \frac{\langle (y(n) - d(n))^2 \rangle}{\langle (d(n) - \langle d(n) \rangle)^2 \rangle}. \quad (8)$$

A perfect match yields  $\text{NMSE} = 0$ , while a completely off-target output gives a NMSE of 1.

## 2.4 Proposed Experimental Setup

Fig. 2 depicts the proposed experimental setup that we have investigated using numerical simulations. This section overviews the three main components: the optoelectronic reservoir, the analogue readout layer and the FPGA board.

### 2.4.1 Opto-electronic reservoir

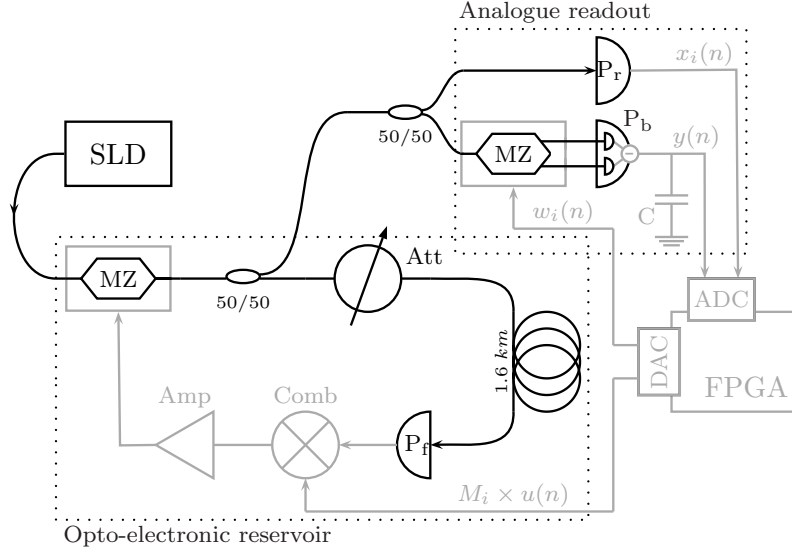
The optoelectronic reservoir is based on the same scheme as in [11, 12]. The reservoir states are encoded into the intensity of incoherent light signal, produced by a superluminescent diode (Thorlabs SLD1550P-A40). The Mach-Zehnder (MZ) intensity modulator (Photline MXAN-LN-10) implements the nonlinear function, its operating point is adjusted by applying a bias voltage, produced by a Hameg HMP4040 power supply. Half of the signal is extracted from the loop and sent to the readout layer. The optical attenuator (Agilent 81571A) is used to set the feedback gain  $\alpha$  of the system (see Eq. (1)). The fibre spool consists of approximately 1.6 km of single mode fibre, giving a round trip time of  $T = 7.94 \mu\text{s}$ . The resistive combiner sums the electrical feedback signal, produced by the feedback photodiode (TTI TIA-525I), with the input signal from the FPGA to drive the MZ modulator, with an additional amplification stage of +27 dB (coaxial amplifier ZHL-32A+) to span the entire  $V_\pi$  interval of the MZ modulator.

### 2.4.2 Analogue readout layer

The analogue readout layer uses the same scheme as proposed in [19]. The optical power it receives from the reservoir is split in two. Half is sent to the readout photodiode (TTI TIA-525I), and the resulting voltage signal, containing the reservoir states  $x_i(n)$ , is recorded by the FPGA for the training process (see Eq. (3)). The other half is modulated by a dual-output Mach-Zehnder modulator (EOSPACE AX-2X2-0MSS-12) which applies the readout weights  $w_i$ , generated by the DAC of the FPGA. The outputs of the modulator are connected to a balanced photodiode (TTI TIA-527), which produces a voltage level proportional to difference of the light intensities received at its two inputs. This allows to multiply the reservoir states by both positive and negative weights [19]. The summation of the weighted states is performed by a low-pass RC filter. The resistance  $R$  of the filter, not shown on the scheme, is the  $50 \Omega$  output impedance of the balanced photodiode. The resulting output signal, proportional to  $y(n)$ , is also recorded by the FPGA, for training and performance evaluation.

Let us compute explicitly the output of the analogue readout layer. The capacitor integrates the output of the balanced photodiode with an exponential kernel and a time constant  $\tau$ . The impulse response of the RC filter is given in [31]

$$h(t) = \frac{1}{RC} e^{\frac{-t}{RC}} = \frac{1}{\tau} e^{\frac{-t}{\tau}}, \quad (9)$$



**Fig. 2** Scheme of the proposed experimental setup. The optical and electronic components are shown in black and grey, respectively. The reservoir layer consists of an incoherent light source (SLD), a Mach-Zehnder intensity modulator (MZ), a 50/50 beam splitter, an optical attenuator (Att), an approximately 1.6 km fibre spool, a feedback photodiode ( $P_f$ ), a resistive combiner (Comb) and an amplifier (Amp). The analogue readout layer contains another 50/50 beam splitter, a readout photodiode ( $P_r$ ), a dual-output intensity modulator (MZ), a balanced photodiode ( $P_b$ ) and a capacitor (C). The FPGA board generates the inputs and the readout weights, samples the reservoir states and the output signal, and trains the system.

the voltage  $Q(t)$  on the capacitor is then given by

$$Q(t) = \int_{-\infty}^t X(s)W(s)h(t-s)ds, \quad (10)$$

where  $X(t)$  is the continuous signal, containing the reservoir states, and  $W(t)$  are the readout weights, applied to the dual-output intensity modulator. The output  $y(n)$  is given by the charge on the capacitor at the discrete times  $t = nT$ :

$$y(n) = Q(nT). \quad (11)$$

Since  $X(t)$  and  $W(t)$  are stepwise functions  $X(t) = x_i(n)$  and  $W(t) = w_i$  for  $t \in [\theta(i-1), \theta i]$ , where  $\theta = T/N$  is the duration of one neuron, we can approximate the integration by a discrete summation to obtain

$$\begin{aligned} y(n) &= \theta \sum_{i=1}^N w_i \left( \sum_{k=0}^{\infty} x_i(n-k)h(N-i-Nk) \right) \\ &= \frac{\theta}{\tau} \sum_{i=1}^N w_i \left( \sum_{k=0}^{\infty} x_i(n-k)e^{-\rho(N-i-Nk)} \right) \end{aligned} \quad (12)$$

where we have introduced the RC integrator ratio  $\rho = \theta/\tau$ .

The readout layer output  $y(t) = Q(t)$  is thus a linear combination of the reservoir states  $x_i$ , weighted by  $w_i$  and by the exponential kernel of the RC filter. Note that contrary to usual reservoir computer outputs (see e.g. Eq. (2), in Eq. (12) the output at time  $n$  depends not only on the current states  $x_i(n)$ , but also on the states at previous times  $x_i(n-k)$ .

In the previous experimental investigation of the analogue readout [20], the readout weights  $w_i$  were computed using ridge regression [32], assuming an output signal given by Eq. (2). But since the experiment produced an output similar to Eq. (12) instead, the readout weights needed to be corrected appropriately. For more details, we refer to the original paper [20]. In the present work, the weights  $w_i$  are adjusted gradually to match the reservoir output signal  $y(n)$  with the target output  $d(n)$  (see Sec. 2.2), without any assumptions about how these weights actually contribute to the output signal  $y(n)$ . This is a much easier task, which allows to obtain better experimental results, as will be shown in Sec. 3.

#### 2.4.3 FPGA board

The reservoir computer is operated by a Xilinx Virtex 6 FPGA chip, placed on a Xilinx ML605 evaluation board. It is paired with a 4DSP FMC151 daughter card, containing one two-channel ADC (Analog-to-Digital converter) and one two-channel DAC (Digital-to-Analog converter). The ADC's maximum sampling frequency is 250 MHz with 14-bit resolution, while the DAC can sample at up to 800 MHz with 16-bit precision.

The FPGA generates the input signal  $M_i \times u(n)$  and sends it into the opto-electronic reservoir. After recording the resulting reservoir states  $x_i(n)$  from one delay loop, it executes the simple gradient descent algorithm in order to update the readout weights  $w_i(n+1)$ . These

are sent to the readout layer and used to generate the output signal  $y(n)$ , also recorded by the FPGA.

## 2.5 Numerical simulations

All numerical experiments were performed in Matlab. We used a custom model of our reservoir computer, based on previous investigations [11, 24], that has been shown to emulate very well the dynamics of the real system. The simulations were performed in discrete time, and took into account the internal structure of the Reservoir Computer described above, such as the ring-like topology, sine nonlinearity and the analogue readout layer with an RC filter. The simulations allow to try out different configurations and to scan various experimental parameters, including values that are impossible to achieve experimentally or imposed by the hardware. All simulations were performed on a dedicated high-performance workstation with 12-core CPU and 64 Gb RAM. Since the convergence of the gradient descent algorithm is quite slow, we limited our investigations to a fast update rate  $k = 10$  (see Eq. (4)), so that each simulation lasted about 24 hours.

The principal goal of the simulations was to check how the online learning approach would cope with experimental difficulties encountered in previous works [19, 20]. To that end, we gathered a list of possible issues and scanned the corresponding experimental parameters in order to check the system performance. In particular, we investigated the following parameters:

- The RC integrator ratio  $\rho$ . This is the most important parameter of the analogue readout layer. While its accurate measure is not required in our setup – since we do not correct the readout weights  $w_i$  – it defines the integration span of the filter, and thus the reservoir states that contribute to the output signal. It can thus significantly influence the results. Another question of importance is how dependent the system performance is on the exact value of  $\rho$ .
- The MZ modulator bias. Mach-Zehnder modulators need to be applied a constant voltage to maintain their transfer function symmetric around zero. The devices we were using up to now are subject to slight drifts over time, often resulting in a non-perfectly symmetric response. We thus checked in simulations whether such an offset would impact the results.
- The DAC resolution. The precision of the DACs on the FMC151 daughtercard is limited to 16 bits. Numerical investigations have shown that the precision of readout weights has a significant impact on the performance, see e.g. [33–35]. We thus checked

whether the resolution available is enough for this experiment.

Besides these potentially problematic parameters, we also scanned the input and feedback gain parameters (denoted by  $\beta$  and  $\alpha$  in Eq. (1)) in order to find the optimal dynamics of the reservoir for each task.

In a separate set of simulations, we investigated the applicability of the proposed method to nonlinear readout layers. That is, we checked whether the simple gradient descent method would still work with a nonlinear response of the analogue readout layer with respect to the reservoir states  $x_i(n)$  (see Eq. (12)). We picked two “saturation” functions of sigmoid shape. This choice arises from the transfer function of common light detectors that are linear at lower intensities and become nonlinear at higher intensities. We used the following functions: a logistic function, given by

$$g_{\text{lg}}(x) = \frac{2}{1 + e^{-2x}} - 1, \quad (13)$$

and the hyperbolic tangent function, given by

$$g_{\text{ht}}(x) = 0.6 \tanh(1.8x). \quad (14)$$

These functions,  $g_{\text{lg}}$  and  $g_{\text{ht}}$ , do not model any particular photodiode, but are two simple examples that allow us to address the above question. Both functions are plotted in figure 5(b), together with a linear response, for comparison.

We investigated two possible nonlinearities in the output layer. In the first case, the readout photodiode ( $P_r$  in figure 2) produces a nonlinear response, while the balanced photodiode ( $P_b$  in figure 2) remains linear. This scenario, that we shall refer to as “nonlinear readout”, allows one to investigate what happens when the reservoir states  $x_i$  used to compute the output signal  $y(n)$  (see Eq. (2)) differ from those employed to update the readout weights (see Eq. (3)). Thus in this case the update rule (Eq. (3)) for the output weights becomes

$$w_i(n+1) = w_i(n) + \lambda(d(n) - y(n))g(x_i(n)), \quad (15)$$

where  $g$  is given by either Eq. (13) or Eq. (14), while the output layer is given by Eq. (12).

In the second case, called “nonlinear output”, the readout photodiode is linear, but the balanced photodiode exhibits a saturable behaviour. In this case the update rule Eq. (3) for the output weights is unchanged, but the output layer Eq. (12) becomes

$$y(n) = \frac{\theta}{\tau} \sum_{i=1}^N w_i \left( \sum_{k=0}^{\infty} g(x_i(n-k)) e^{-\rho(N-i-Nk)} \right). \quad (16)$$

Note that we have only considered cases with just one nonlinear photodiode, so as to check whether the difference between the reservoir states used for training and those to compute the readout (see Eqs. (3) and (2), respectively) would degrade the performance of the system. The scenario with both nonlinear photodiodes is hence more simple, as the reservoir states are the same in both equations. One could consider the case with two photodiodes exhibiting different nonlinear behaviours. In that situation, similar to the results we will show in Sec. 3, we expect the algorithm to cope with the difference up to a certain point, before running into troubles. For this reason, we leave that scenario for future investigations.

### 3 Results

#### 3.1 Linear readout: RC circuit

In this section we present our numerical results and answer the questions raised in the previous section.

For each of the two tasks considered here, we performed three kinds of simulations: we scanned the RC integrator ratio  $\rho = \theta/\tau$  in the first simulation, the MZ bias in the second, and the resolution of the DAC in the third. Furthermore, since different values of these parameters may work better with different dynamics of the reservoir, we also scanned the input gain  $\beta$  and the feedback gain  $\alpha$  in all three simulations independently, and applied the optimal values in each case.

For both tasks, we used a network with  $N = 50$  neurons, as in most previous experimental works [11, 14, 16, 20]. The reservoir was trained on 83000 inputs, with an update rate  $k = 10$ , and then tested over  $10^5$  symbols for the channel equalisation task and  $10^4$  inputs for NARMA10 task. For statistical purposes, we ran each simulation 10 times, with different random input masks. In the following figures, averaged results over the masks are plotted, while the error bars give the standard deviation over the different input masks. Results related to the channel equalisation task are plotted with solid lines, while dashed lines correspond to those for NARMA10.

For the channel equalisation task, our system yields SERs between  $10^{-4}$  and  $10^{-3}$  depending on the input mask, as summarised in Tab. 1 (first line). This is comparable to previous experiments with the same opto-electronic reservoir: error rates of order of  $10^{-4}$  were reported in [11] using a digital readout and in [20] with an analogue readout, using an RLC filter. The first experimental analogue system, using a simple RC circuit, as we did in this work, performed significantly worse,

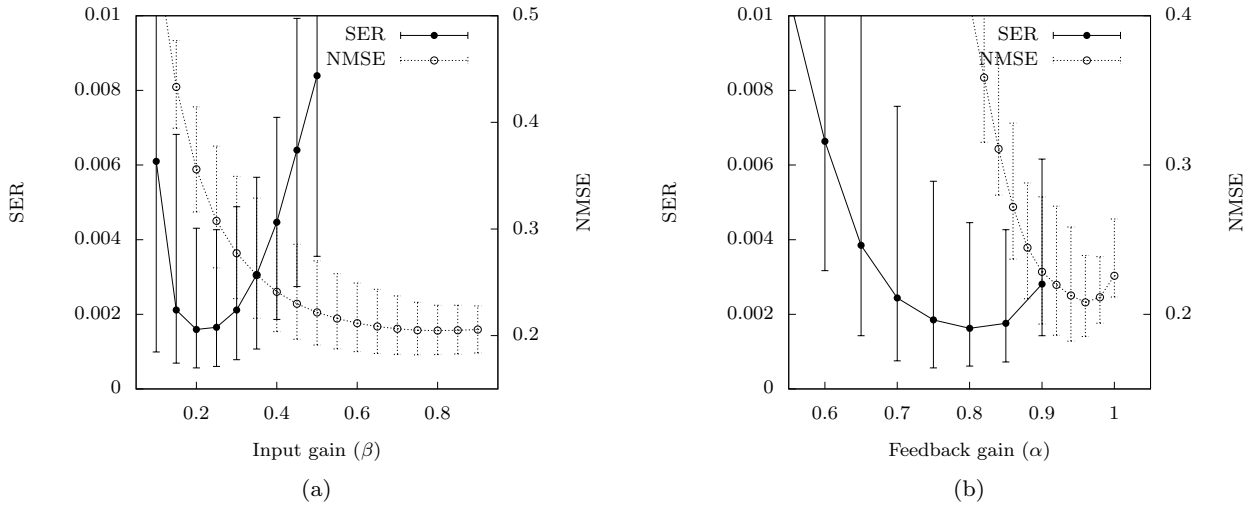
with SER of order of  $10^{-2}$  [19]. That is, online learning does not outperform other methods, but allows to obtain significantly better results with a simpler setup.

As for the NARMA10 task, we obtain a NMSE of  $0.20 \pm 0.02$ . Previous experiments with a digital readout layer produced  $0.168 \pm 0.015$  [11] and  $0.107 \pm 0.012$  [16]. With an analogue readout layer, the best NMSE reported was  $0.230 \pm 0.023$  [20]. Our system thus slightly outperforms the analogue approach, and gets close to the digital one, except for the very good result obtained with a different reservoir, based on a passive cavity [16]. Again, our results were obtained with a simple setup and no modelling of the readout, contrary to [20].

Furthermore, the error rates obtained here can be significantly lowered with more training, as has been demonstrated numerically and experimentally in [24]. To keep reasonable simulation times (about 24 hours per simulation), we limited the training to 83000 input values, with an update rate  $k = 10$ . Higher update rates can be used experimentally, since running the opto-electronic setup is much faster than simulating it. We thus expect to obtain lower error rates experimentally with longer training sets and update rates up to  $k = 200$ . To illustrate this point with results reported in [24], short training sets with  $k = 10$  yielded SERs of order of  $10^{-4}$  for the channel equalisation task. Increasing  $k$  up to 200 allowed to decrease the error rate down to  $5.7 \times 10^{-6}$ .

Figures 3(a) and 3(b) show the influence of input and feedback gain parameters on the performance of the system. All curves present a pronounced minimum, except for the input gain  $\beta$  for the NARMA10 task, where values above 0.6 seem to produce comparable results. Note that the channel equalisation task requires a low input signal with  $\beta = 0.2$ , while NARMA10 works best with stronger input and  $\beta = 0.8$ . As for the feedback gain, NARMA10 shifts the system close to the chaotic regime with  $\alpha = 0.95$ , while channel equalisation works better with  $\alpha = 0.8$ .

Figure 4(a) shows the results of the scan of the RC integrator ratio  $\rho$ . Both tasks work well on a relatively wide range of values, with NARMA10 much less sensitive to  $\rho$  than channel equalisation. In particular, the channel is equalised best with  $\rho = 0.03$ . With  $N = 50$ , this corresponds to  $\tau = T/0.03N = 5.29 \mu\text{s}$ , which is shorter than the roundtrip time  $T = 7.94 \mu\text{s}$ . On the other hand, NARMA10 output is best reproduced with  $\rho = 0.003$ , which yields  $\tau = T/0.003N = 52.93 \mu\text{s}$ . This is significantly longer than the roundtrip time  $T$ , meaning that reservoir states from previous time steps are also taken into account for computation of an output value. This is not surprising, since NARMA10 function has a long memory (see Eq. (7)). However, this memory



**Fig. 3** Reservoir computer performances for different input ( $\beta$ ) and feedback ( $\alpha$ ) gains (solid lines: channel equalisation, dashed lines: NARMA10). **(a)** While channel equalisation is relatively sensitive to  $\beta$ , NARMA10 works well in a wide range of values. Note that although it seems that higher input gain would give better results, the dashed curve actually rises slightly for large  $\beta$ , and the optimum input gain is around 0.8. **(b)** Both tasks require a system with significant memory (feedback gain at least  $\alpha = 0.8$ ), and even a near-chaotic regime for NARMA10 ( $\alpha = 0.95$ ).

effect in the readout layer is not crucial, as the system performs equally well with higher  $\rho$  and thus lower  $\tau$ . All in all, these results are very encouraging for upcoming experiments, as they show that an accurate choice of capacitor is not crucial for the performance of the system.

Figure 4(b) illustrates the impact of the bias of the readout Mach-Zehnder modulator on the reservoir computer performance. NARMA10 task is clearly more affected by this offset, as the NMSE grows quickly from a bias of roughly 0.06. The SER, on the other hand, stays low until 0.1. For a MZ modulator with  $V_\pi = 4.5$  V this corresponds to a tolerance of roughly 0.1 V, which is superior to expected experimental deviations. The Hameg power supply that we use to bias the modulator (see Sec. 2.4) has a resolution of 0.001 V.

Figure 5(a) shows that the 16-bit DAC resolution is not an issue for this experiment, as the minimal precision required for good performance is 8 bits, for both tasks.

### 3.2 Nonlinear readout

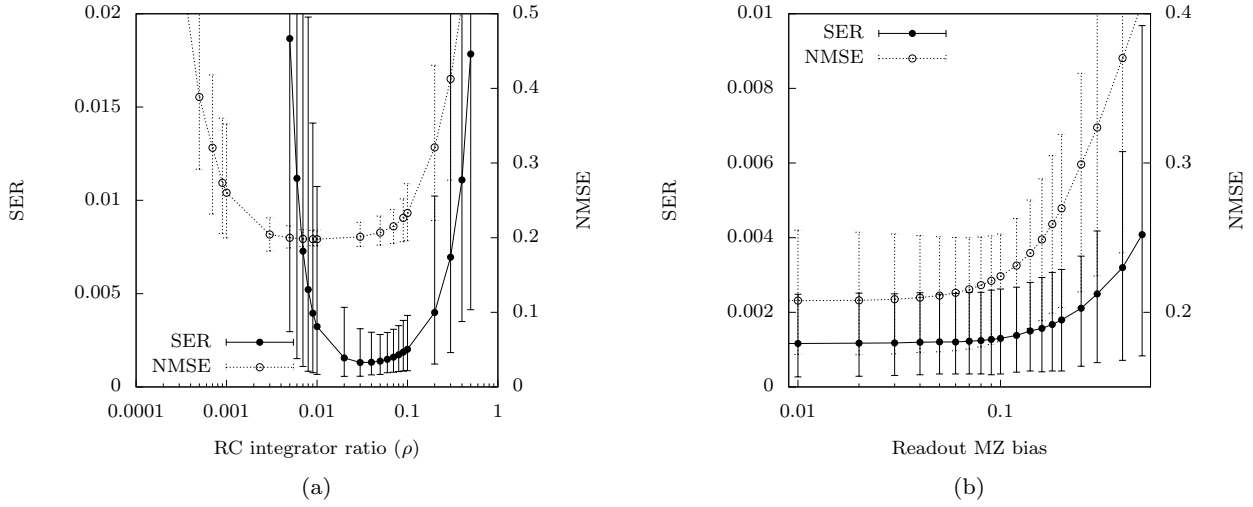
Table 1 sums up the results obtained with a nonlinear readout layer. We used optimal experimental parameters, as described above, and generated new sets of data for the training and test phases. We investigated two scenarios and used two functions of sigmoid shape,  $x \rightarrow g_{lg}(x)$  and  $x \rightarrow g_{ht}(x)$ , as described in section 2.5. The system was trained over 83000 inputs, with an update rate  $k = 10$ , and tested over  $10^5$  symbols

for the channel equalisation task and  $10^4$  inputs for NARMA10. We report error values averaged over 10 trials with different random input masks, as well as the standard deviations. The figures show that the performance deterioration is more manifest with the hyperbolic tangent function  $g_{ht}$ , as it is much more nonlinear than the logistic function  $g_{lg}$ . Overall, the added nonlinearity does not have a significant influence on the results in both cases. The SER roughly doubles, at most, for the channel equalisation task. The impact on NARMA10 is barely noticeable, as the error increase of 5% is smaller than the standard deviation. Using offline training on the same system (i.e. with nonlinear output) we observed an increase of the SER by one order of magnitude for the channel equalisation task, and a 30% increase of the NMSE with the NARMA task. These results show that online training is very well suited for experimental analogue layers, as it can cope with realistic components that do not have a perfectly linear response.

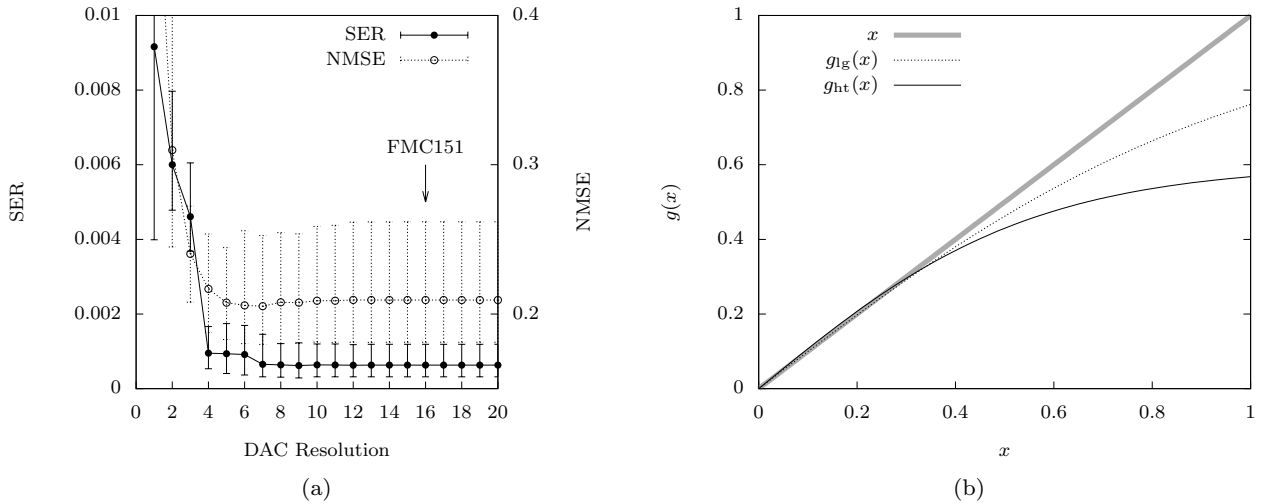
## 4 Conclusion

In this work we propose the online learning technique to improve the performance of analogue readout layers for photonic reservoir computers. We demonstrate an optoelectronic setup with an output layer based on a simple RC filter, and test it, using numerical simulations, on two benchmark tasks. Training the setup online, with a simple gradient descent algorithm, allows to obtain





**Fig. 4** Impact of the RC integrator ratio ( $\rho$ ) and the readout MZ modulator bias on the reservoir computer performance (solid lines: channel equalisation, dashed lines: NARMA10). **(a)** Ratios within  $\rho \in [0.03, 0.08]$  are suitable for channel equalisation and  $\rho \in [0.002, 0.07]$  for NARMA10. Remarkably, inaccurate choice of  $\rho$ , and thus  $\tau$ , will not result in significant performance loss, as long as the value lies approximately in the optimal interval. **(b)** Although the NARMA10 task is more sensitive to this bias, both tasks work reasonably well with a bias up to 0.06, which is superior to expected experimental deviations.



**Fig. 5** **(a)** Impact of the DAC resolution on the reservoir computer performance (solid lines: channel equalisation, dashed lines: NARMA10). The results show that the 16-bit resolution of the FMC151 daughtercard is sufficient for this application. **(b)** Nonlinear response curves of the photodiodes: hyperbolic tangent function  $g_{ht}$  (solid line) and logistic function  $g_{lg}$  (dotted line). The linear response is plotted with a thick grey line.

the same level of performance as with a digital readout layer. Furthermore, our approach doesn't require any modelling of the underlying hardware, and is robust against possible experimental imperfections, such as inaccurate choice of parameters or components. It is also capable of dealing with a nonlinearity in the readout layer, such as saturable response of the photodiodes. Finally, we expect the conclusions of the present investigation, namely the advantage of online training, to be applicable to all hardware reservoir computers, and

not restricted to the delay dynamical opto-electronic systems used for the sake of illustration in the present work.

Note that the proposed setup is rather slow for practical applications. With a roundtrip time of  $T = 7.94 \mu\text{s}$ , its bandwidth is limited to 126 kHz. This is significantly lower than e.g. a standard Wi-Fi 802.11g channel with a bandwidth of 20 MHz. The speed limit of the system is set by the bandwidth of the different components, and in particular of the ADC and DAC. However, the speed

Readout	Transfer function	Chan. Equal. (SER $\times 10^{-3}$ )	NARMA10 (NMSE)
Linear	$x$	$1.1 \pm 0.7$	$0.20 \pm 0.02$
Nonlinear readout	$g_{lg}(x)$	$1.3 \pm 0.9$	$0.21 \pm 0.03$
	$g_{ht}(x)$	$1.2 \pm 0.8$	$0.21 \pm 0.02$
Nonlinear output	$g_{lg}(x)$	$2.0 \pm 1.6$	$0.21 \pm 0.02$
	$g_{ht}(x)$	$2.5 \pm 2.1$	$0.21 \pm 0.01$

**Table 1** Summary of reservoir computer performances with nonlinear readout layers, measured with error metrics related to the tasks considered here. All values are averaged over 10 random input masks and presented with their standard deviations. We used two functions with sigmoid shape to model the response of the photodiodes. We investigated two scenarios: in the “nonlinear readout” configuration, the readout photodiode  $P_r$  is nonlinear, while the balanced photodiode  $P_b$  is linear, and vice versa in the “nonlinear output” scheme. The linear case  $x \rightarrow x$  is shown for comparison. For both tasks, the added nonlinearity does not significantly deteriorate the system performance.

of the setup can be easily increased. For instance, with  $T = 50$  ns (and thus, a bandwidth of 20 MHz), and keeping  $N = 50$ , the reservoir states should have a duration of 1 ns, and hence the ADC and DAC should have bandwidths significantly above 1 GHz. Such components are readily available commercially. As an illustration of how a fast system would operate, we refer to the optical experiment [15] in which information was injected into a reservoir at rates beyond 1 GHz.

The results reported in this work will serve as a basis for future investigations involving experimental validation of the proposed method. Experimental realisation of an efficient analogue readout layer would allow building fully-analogue high-performance RCs, abandon the slow digital post-processing and take full advantage of the fast optical components. Such setups could be applied to emerging communication channels [36]. Furthermore, fully-analogue setups would open the possibility of feeding the output signal back into the reservoir, thus significantly enriching its dynamics and making it capable of solving signal generation tasks. Recent investigations reported a reservoir computer with digital output feedback capable of periodic and chaotic signal generation [35,37]. Replacing the digital layer in these implementations with an analogue solution would significantly increase the speed of such generators. Our work thus brings an efficient solution to an important problem in the reservoir computing field, potentially leading to a significant speed gain and a broader range of applications.

## Compliance with Ethical Standards

**Funding.** This study was funded by the Interuniversity Attraction Poles program of the Belgian Science Policy Office (grant IAP P7-35 “photonics@be”), by the Fonds de la Recherche Scientifique FRS-FNRS and by the Action de Recherche Concertée of the Académie Universitaire Wallonie-Bruxelles (grant AUWB-2012-12/17-ULB9).

**Conflict of Interest.** All authors declare that they have no conflict of interest.

**Ethical approval.** This article does not contain any studies with human participants or animals performed by any of the authors.

## References

1. H. Jaeger, H. Haas, *Science* **304**, 78 (2004)
2. W. Maass, T. Natschläger, H. Markram, *Neural comput.* **14**, 2531 (2002)
3. M. Lukoševičius, H. Jaeger, *Comp. Sci. Rev.* **3**, 127 (2009)
4. F. Triefenbach, A. Jalalvand, B. Schrauwen, J.P. Martens, *Adv. Neural Inf. Process. Syst.* **23**, 2307 (2010)
5. B. Meftah, O. Lézoray, A. Benyettou, *Cognitive Computation* **8**(2), 237 (2016)
6. Z.K. Malik, A. Hussain, J. Wu, *Cognitive Computation* **6**(3), 595 (2014)
7. S. Scardapane, A. Uncini, *Cognitive Computation* pp. 1–11 (2016)
8. The 2006/07 forecasting competition for neural networks & computational intelligence. <http://www.neural-forecasting-competition.com/NN3/> (2006)
9. H. Arsenault, *Optical processing and computing* (Elsevier, 2012)
10. L. Appeltant, M.C. Soriano, G. Van der Sande, J. Danckaert, S. Massar, J. Dambre, B. Schrauwen, C.R. Mirasso, I. Fischer, *Nat. Commun.* **2**, 468 (2011)
11. Y. Paquot, F. Duport, A. Smerieri, J. Dambre, B. Schrauwen, M. Haelterman, S. Massar, *Sci. Rep.* **2**, 287 (2012)
12. L. Larger, M. Soriano, D. Brunner, L. Appeltant, J.M. Gutiérrez, L. Pesquera, C.R. Mirasso, I. Fischer, *Opt. Express* **20**, 3241 (2012)
13. R. Martinenghi, S. Rybalko, M. Jacquot, Y.K. Chembo, L. Larger, *Phys. Rev. Lett.* **108**, 244101 (2012)
14. F. Duport, B. Schneider, A. Smerieri, M. Haelterman, S. Massar, *Opt. Express* **20**, 22783 (2012)
15. D. Brunner, M.C. Soriano, C.R. Mirasso, I. Fischer, *Nature communications* **4**, 1364 (2013)
16. Q. Vinckier, F. Duport, A. Smerieri, K. Vandoorne, P. Bienstman, M. Haelterman, S. Massar, *Optica* **2**(5), 438 (2015)
17. K. Vandoorne, P. Mechet, T. Van Vaerenbergh, M. Fiers, G. Morthier, D. Verstraeten, B. Schrauwen, J. Dambre, P. Bienstman, *Nat. Commun.* **5**, 3541 (2014)
18. D. Woods, T.J. Naughton, *Nature Physics* **8**(4), 257 (2012)
19. A. Smerieri, F. Duport, Y. Paquot, B. Schrauwen, M. Haelterman, S. Massar, in *Advances in Neural Information Processing Systems* (2012), pp. 944–952
20. F. Duport, A. Smerieri, A. Akrou, M. Haelterman, S. Massar, *Sci. Rep.* **6**, 22381 (2016)

21. Q. Vinckier, A. Bouwens, M. Haelterman, S. Massar, in *Conference on Lasers and Electro-Optics* (Optical Society of America, 2016), p. SF1F.1
22. L. Bottou, in *Online Learning and Neural Networks* (Cambridge University Press, 1998)
23. S. Shalev-Shwartz, *Foundations and Trends in Machine Learning* **4**(2), 107 (2012)
24. P. Antonik, F. Duport, M. Hermans, A. Smerieri, M. Haelterman, S. Massar, *IEEE Transactions on Neural Networks and Learning Systems* **28**(11), 2686 (2017)
25. A. Rodan, P. Tino, *IEEE Trans. Neural Netw.* **22**, 131 (2011)
26. G.B. Arfken, *Mathematical methods for physicists* (Orlando FL: Academic Press, 1985)
27. C.M. Bishop, *Pattern recognition and machine learning* (Springer, 2006)
28. W.H. Press, B.P. Flannery, S.A. Teukolsky, W.T. Vetterling, *Numerical Recipes: The Art of Scientific Computing* (1986)
29. L. Bottou, in *Advanced Lectures on Machine Learning* (Springer Verlag, 2004), pp. 146–168
30. V.J. Mathews, J. Lee, in *SPIE's 1994 International Symposium on Optics, Imaging, and Instrumentation* (International Society for Optics and Photonics, 1994), pp. 317–327
31. P. Horowitz, W. Hill, *The art of electronics* (Cambridge University Press, 1980)
32. A.N. Tikhonov, A. Goncharsky, V. Stepanov, A.G. Yagola, *Numerical methods for the solution of ill-posed problems*, vol. 328 (Springer Netherlands, 1995)
33. M.C. Soriano, S. Ortín, D. Brunner, L. Larger, C. Mirasso, I. Fischer, L. Pesquera, *Optics express* **21**(1), 12 (2013)
34. M.C. Soriano, S. Ortín, L. Keuninckx, L. Appeltant, J. Danckaert, L. Pesquera, G. Van der Sande, *IEEE transactions on neural networks and learning systems* **26**(2), 388 (2015)
35. P. Antonik, M. Hermans, F. Duport, M. Haelterman, S. Massar, in *SPIE's 2016 Laser Technology and Industrial Laser Conference*, vol. 9732 (2016), vol. 9732, p. 97320B
36. M. Bauduin, Q. Vinckier, S. Massar, F. Horlin, in *Signal Processing Advances in Wireless Communications (SPAWC), 2016 IEEE 17th International Workshop on* (IEEE, 2016), pp. 1–5
37. P. Antonik, M. Hermans, M. Haelterman, S. Massar, in *APNNS's 23th International Conference on Neural Information Processing, LNCS*, vol. 9948 (2016), *LNCS*, vol. 9948, pp. 318–325