

Functional identification of biological neural networks using reservoir adaptation for point processes

Tayfun Gürel · Stefan Rotter · Ulrich Egert

Received: 23 December 2008 / Revised: 18 May 2009 / Accepted: 2 July 2009 / Published online: 29 July 2009
© The Author(s) 2009. This article is published with open access at Springerlink.com

Abstract The complexity of biological neural networks does not allow to directly relate their biophysical properties to the dynamics of their electrical activity. We present a reservoir computing approach for functionally identifying a biological neural network, i.e. for building an artificial system that is functionally equivalent to the reference biological network. Employing feed-forward and recurrent networks with fading memory, i.e. reservoirs, we propose a point process based learning algorithm to train the internal parameters of the reservoir and the connectivity between the reservoir and the memoryless readout neurons. Specifically, the model is an Echo State Network (ESN) with leaky integrator neurons, whose individual leakage time con-

stants are also adapted. The proposed ESN algorithm learns a predictive model of stimulus-response relations in *in vitro* and simulated networks, i.e. it models their response dynamics. Receiver Operating Characteristic (ROC) curve analysis indicates that these ESNs can imitate the response signal of a reference biological network. Reservoir adaptation improved the performance of an ESN over readout-only training methods in many cases. This also held for adaptive feed-forward reservoirs, which had no recurrent dynamics. We demonstrate the predictive power of these ESNs on various tasks with cultured and simulated biological neural networks.

Keywords Cultured neural networks · Echo State Networks · Reservoir computing

Action Editor: Rob Kass

T. Gürel (✉)
Bernstein Center for Computational Neuroscience
Freiburg and Institute for Computer Science,
Albert-Ludwig University of Freiburg, Germany
e-mail: guerel@informatik.uni-freiburg.de

S. Rotter
Bernstein Center for Computational Neuroscience
Freiburg and Faculty of Biology,
Albert-Ludwig University of Freiburg, Germany

U. Egert
Department of Microsystems Engineering,
Albert-Ludwig University of Freiburg, Germany

U. Egert
Bernstein Center for Computational Neuroscience,
Albert-Ludwig University of Freiburg, Germany

U. Egert
Bernstein Focus Neurotechnology Freiburg,
Albert-Ludwig University of Freiburg, Germany

1 Introduction

One central goal in neuroscience is to understand how the brain represents, processes, and conveys information. Starting from Hebb's cell assemblies (Brown and Milner 2003; Hebb 1949), many neurobiologically founded theories and hypotheses have been developed towards this goal. It stands clear now that spikes are the elemental quanta of information processing in the mammalian cortex (Hodgkin and Huxley 1952; Kandel et al. 2000). As a result of extensive experiments of cortical recordings, it has been widely postulated and accepted that function and information of the cortex are encoded in the spatio-temporal network dynamics (Abeles et al. 1993; Lindsey et al. 1997; Mainen and Sejnowski 1995; Prut et al. 1998; Riehle et al. 1997; Villa et al. 1999). The right level of describing the dynamics,

however, is a matter of intensive discussions (Prut et al. 1998; Rieke et al. 1999). Are spike rates or spike timings more relevant? What is the right temporal precision if the latter proves significant? What should be the spatial resolution of this description? How far can the population activity of neurons be related to function or behavior? Does the correlated activity of multiple neurons indicate a functionally relevant state?

Depending on the answers to the above questions one would preferably apply different models to relate the network activity to function. Another approach is to employ a generic network model, which can be assumed to be universal for problems of neural encoding. The parameters of the model would be learned by adaptive algorithms. Obviously, such a model should be able to deal with single spikes with high temporal precision as well as population rates. It should also be able to catch, with the appropriate parameters, network synchrony and polychrony (Izhikevich 2006).

1.1 Reservoir computing

Liquid State Machines (LSM) and Echo State Networks (ESN) have been introduced as efficiently learning recurrent neural networks (Jaeger 2001; Maass et al. 2002). The common key contribution of these approaches is the proposal of a recurrent neural network with a fixed connectivity, i.e. a *reservoir*, which does not have stable states and has a fading memory of the previous inputs and network states. In response to an input stream, the reservoir generates a higher-dimensional spatio-temporal dynamics reflecting the structure in the input stream. The higher dimensional reservoir state can be mapped to a target output stream online, with a second module, namely a readout. LSMs are networks of spiking *integrate-and-fire* (IAF) neurons whereas ESNs use continuous valued sigmoid neurons and a single layer of readout neurons (see Section 2).

With the appropriate parameters, reservoir dynamics can be sensitive to different features of the input such as correlations, polychronous and synchronous spikes, different frequency bands or even temporally precise single spikes. For instance, LSMs can approximate any time invariant filter with fading memory if their traces of internal states differ at least for one time point in response to any two different input streams (*Separation Property, SP*) and if the readout modules can approximate any continuous function from $\mathbb{R}^m \rightarrow \mathbb{R}$, where $m \in \mathbb{N}$ (*Approximation Property, AP*) (Maass et al. 2002). Satisfying the separation property depends on whether the reservoir is composed of sufficient basis filters. A random reservoir needs to have a rich repertoire of basis filters in order to approximate the target

time-invariant filter. This could be achieved for several tasks with sufficiently large random reservoirs (Maass et al. 2002). Furthermore, reservoirs have been shown to simulate a large class of higher order differential equations with appropriate feedback and readout functions (Maass et al. 2007). These findings suggest that reservoir computing can be used as a generic tool for the problems of neural encoding.

Biological neural networks, *in vivo*, process a continuous stream of inputs in real time. Moreover, they prove successful to react and compute fast, independent of their instantaneous state and ongoing activity, when prompted by sudden changes in stimuli. In other words, they perform *any time computing*. Reservoir computing has also been suggested as a model of cortical information processing for their capability of online and any-time computing, for their fading memory and for their separation properties. It has been argued that specific cortical circuitry is possible to build into generic LSM framework (Maass et al. 2004). Bringing reservoir computing into the problem will not only deliver expressive models that can distinguish a rich set of input patterns, but also may provide more biological relevance to the theoretical tool.

1.2 Neuronal cell cultures as biological neural networks

While brain tissue has highly specialized architecture and developmental history, generic biological networks can be created as cell cultures of mammalian cortical neurons that have been dissociated and regrown outside an organism. They are closed system *in vitro* living networks, which are frequently used to study physiological properties of neural systems (Marom and Shahaf 2002). Although the anatomical structure of the brain is not preserved in cultures, their inherent properties as networks of real neurons, their accessibility and small size make them very attractive for investigating information processing in biological networks. Using cultured networks also eliminates the problem of interference with ongoing activity from different parts of the brain. Compared to *in vitro* brain slices, cultured networks are more accessible in terms of ratio of the recorded neurons to the whole network. In other words, they pose a less serious under-sampling problem. Studying such networks can provide insight into the generic properties biological neural networks, independent of a specific anatomy (Marom and Shahaf 2002). Another motivation to study cultures is understanding and employing them as biological computers. For instance, neuronal cell cultures have been shown

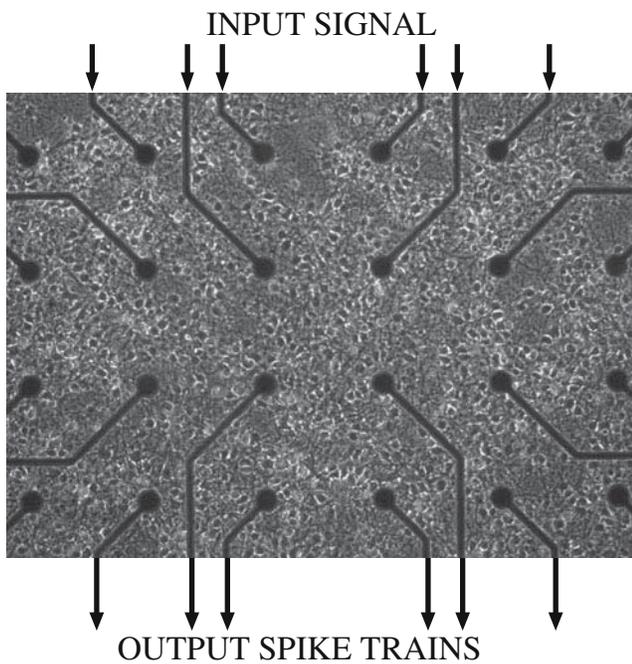


Fig. 1 A photo image of a neuronal culture with MEA electrodes. We aim at generating an equivalent network in terms of input–output relations. Photo was taken by Steffen Kandler from BCCN Freiburg

to be able to learn. Shahaf and Marom (2001) demonstrated how the response of the network to a stimulus can be ‘improved’ by systematically training the culture. Jimbo et al. (1999) showed how neuronal cultures can increase or decrease their overall response to pulse inputs stimulus by tetanic stimuli training.

Micro-Electrode Arrays (MEA) have commonly been employed to both stimulate and record from neuronal cultures and slices (Egert et al. 2002; Marom and Shahaf 2002) (Fig. 1). Standard MEAs allow for simultaneous recordings and stimulations from up to 60 electrodes from a surface area of around 2 mm². Each electrode picks up the extracellular electrical field of one or several (1 to 3) neurons. Neurons transmit

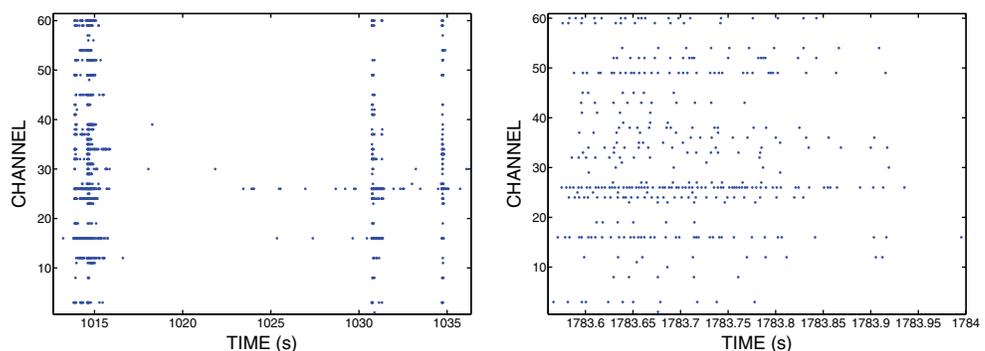
information by *action potentials*, or *spikes*, which can be extracted by adaptive high-pass filtering of the extracellular signals (Egert et al. 2002; Wagenaar et al. 2005).

The activity in neuronal cultures is composed of irregular network-wide bursts of spikes, even in absence of an external stimulation (Marom and Shahaf 2002). These networks display little or no spiking activity most of the time (that is, between bursts) and very high spiking activity during the bursts (Fig. 2). Although the inter-burst intervals are not necessarily regular, there seem to be spatio-temporal patterns that occur within bursts (Feber le et al. 2007; Rolston et al. 2007). For example, a burst might always start with the activity of the same channel and continue with the activity of another particular channel.

1.3 Problem statement

Although we cannot relate the activity dynamics to a physiological function in random *in vitro* BNNs, by studying their activity dynamics, we gain experience and information about generic network properties forming the basis of *in vivo* networks. Moreover, one can assign pseudo-functions to random BNNs by artificially mapping network states to a predefined set of actions (Chao et al. 2008). One can also regard the response spike train of a BNN to a stream of various stimuli as a very detailed characterization of its pseudo-function and aim at modeling stimulus-response relations. In the present work, we take this approach. We record the output responses of simulated and cultured BNNs to random multivariate streams of stimuli. We tackle the question *whether it is possible to train an artificial neural network that predicts the response of a reference biological neural network under the applied stimulus range*. In other words, we aim at generating an equivalent network of a BNN in terms of stimulus-response relations. Given the same stimulus, the equivalent network should predict the output of the biological neural network.

Fig. 2 Spike activity in neuronal cultures. Burst activity (left). Zoom into a burst. Each dot shows a spike detected on the corresponding electrode at the corresponding time (right)



A model or a predictor for biological neural networks can be useful for relating the physiological and physical determinants of its activity and thereby, can be a tool for analyzing information coding in these networks. It can also be helpful for interacting with BNNs by means of electrical stimulation. Here, we employ Echo State Networks (ESN) as a reservoir computing tool and propose an algorithm to find appropriate models for the relations between continuous input streams and multi-unit recordings in biological neural networks. The algorithm uses point process log-likelihood as an optimization criterion and adapts the readout and reservoir parameters of ESNs accordingly. Moreover, we investigate the performance of our approach on different feed-forward and recurrent reservoir structures and demonstrate its applicability to *stimulus-response* modeling problem in BNNs.

We shortly review ESNs in Section 2 and point process modeling of spike data in Section 3. An elaboration of ESN adaptation for point processes is presented in Section 4. In Section 5 we present our evaluation methods. A detailed experimental section and their implications can be found in Sections 6 and 7, respectively.

2 Echo State Networks

Echo State Networks (ESN) (Jaeger 2001) consist of a recurrent neural network (RNN) reservoir and a readout module (Fig. 3). Recurrent neural networks (RNN) are in general difficult to train with gradient descent methods (Bengio et al. 1994). The *echo state* approach to RNNs is motivated by the observation that a fixed (untrained) RNN can be very useful for discriminating multivariate time series if the RNN is gradually forgetting internal states and thereby also the inputs at previous time steps. This property is described as generating the *echo states* of the input. Existence of

echo states is independent of the specific structure of the reservoir, but strongly determined by the extension of the eigenvalues of its connectivity matrix. The input stream is used to stimulate a higher dimensional dynamical system, which can be mapped to a target output stream by adaptive linear or sigmoidal readout units. Even with random fixed reservoirs, ESNs make up a successful reservoir computing framework with applications in several engineering and nonlinear modeling tasks (Jaeger 2003; Jaeger and Eck 2006; Jaeger and Haas 2004).

2.1 Dynamics

The ESN dynamics is described as

$$x^{n+1} = f(W^{in} u^{n+1} + W x^n)$$

$$y^n = f^{out}(W^{out} [u^n; x^n]),$$

where,

- W $N \times N$ internal reservoir weight matrix,
- W^{in} $N \times K$ input weight matrix,
- W^{out} $L \times (N + K)$ output weight matrix,
- x^n $N \times 1$ state vector for time step n ,
- u^n $K \times 1$ input vector for time step n ,
- y^n $L \times 1$ output vector for time step n ,
- ',' vertical vector concatenation.

Note that ESNs can have feedback projections from the readout module to the reservoir (Jaeger 2001). Throughout this note, however, we employ ESN architectures without such feedback projections. We choose $f(x) = \tanh(x) = \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)}$, which allows for existence of echo states.

2.2 ESN with leaky integrators

Fading memory in the reservoir is due to recurrent connectivity if the individual neurons are memoryless. Using leaky integrator neurons results in a reservoir made from a network of low pass filters, which can be referred to as Leaky Integrator Echo State Networks (LI-ESN). LI-ESNs have been shown to work well on noisy systems with slowly changing dynamics (Jaeger et al. 2007). The leakage time constant is another aspect of fading memory. For instance, even a purely feed-forward reservoir will have a fading memory due the memory of the individual neurons. We describe the LI-ESN dynamics with

$$x^{n+1} = \left(1 - \frac{1}{\tau}\right) x^n + \frac{1}{\tau} f(W^{in} u^{n+1} + W x^n), \quad \tau \geq 1.$$

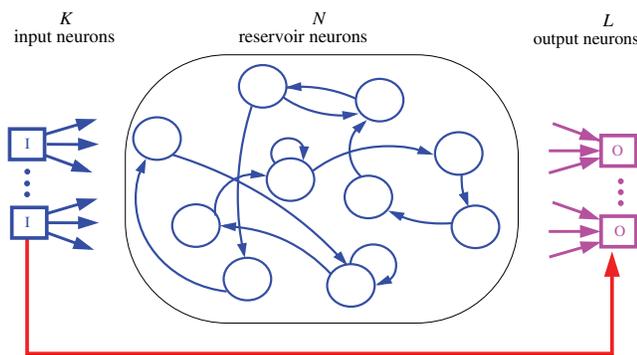


Fig. 3 Architecture of Echo State Networks

$$y^n = f^{out} (W^{out} [u^n; x^n]),$$

where τ is a global parameter for all reservoir neurons.

2.3 Feed-Forward Echo State Networks

Acyclic random neural networks with leaky integrator neurons can be used as reservoirs, as they also have fading memory. We refer an ESN with such a reservoir as a Feed-Forward Echo State Network (FF-ESN). Naturally, it can be asked whether FF-ESNs can have the same expressive power as the recurrent ESNs. Intuitively, it can be expected that a recurrent ESN with the same number of neurons discriminates between more input patterns than a FF-ESN, i.e. the recurrent ESN has a much more powerful *separation property*. In case of adaptive reservoirs, however, it is worthwhile to test whether the reservoir adaptation algorithm works especially better on feed-forward reservoirs than on recurrent ones. Here, we also propose the use of FF-ESNs for functional identification of biological neural networks. A recurrent reservoir can easily be made acyclic by inverting the connections that cause recurrence (Fig. 4).

2.4 ESN learning

Learning in Echo State Networks will typically include learning readout parameters by linear regression, e.g. using the pseudo inverse matrix

$$\tilde{W}^{out} = Y [X; U]^+,$$

where $L \times T$ matrix Y is the collection of target vectors y for T time steps, X, U are the same collection for

reservoir states and input vectors. $+$ is pseudo inverse operation. Much more efficient is the use of the Wiener-Hopf equation for ESN readout learning (Jaeger 2001). Steil (2004) proposed the *backpropagation-decorrelation* algorithm, which seeks a trade-off between error minimization and decorrelation of the reservoir activity.

Although ESN learning is restricted to readout learning in many cases, there have recently been several approaches to adapt reservoirs, among which a significant improvement over untrained reservoirs was reported by Steil (2007). Adopting the intrinsic plasticity rule from real biological systems improved the performance of the backpropagation-decorrelation algorithm.

In the current approach, we adapt the reservoir connectivity and individual neuronal time constants with a one-step propagation of the log-likelihood into the reservoir. This is basically a gradient descent approach, where the log-likelihood of a point process is used as an optimization criterion. We elaborate our approach in Section 4.

3 Point process modeling of spike data

Signals that are comprised of point-like events in time, e.g. trains of action potentials generated by neurons, can be characterized in terms of stochastic point processes (Brown et al. 2001; Chornoboy et al. 1988; Okatan et al. 2005; Rajaram et al. 2005). A spiking process, modeled as a point process, is in turn fully characterized by its conditional intensity function (Cox and Isham 1980; Daley and Vere-Jones 2003).

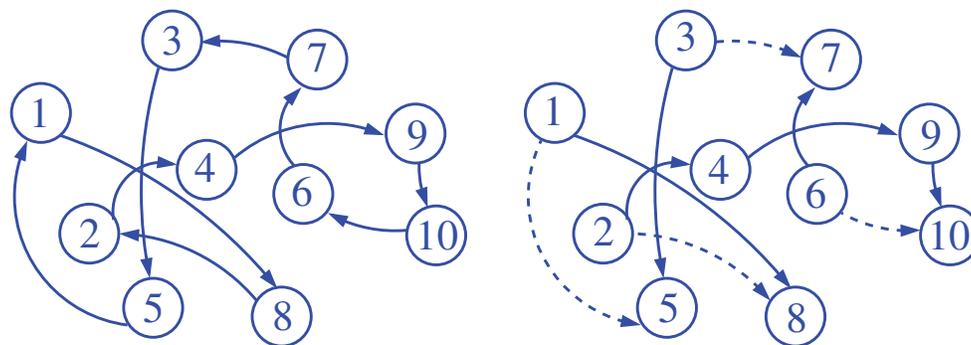


Fig. 4 A recurrent sparse random reservoir (left) and a feed-forward sparse random reservoir (right). Numbers denote the neuron indices. A reservoir is guaranteed to be acyclic if connections are allowed only from neurons with lower indices to

neurons with higher indices. A recurrent reservoir (left) can be converted into a feed-forward one by inverting the connections that are from higher indexed neurons to lower indexed ones (dashed lines, right)

The instantaneous firing rate is defined as a conditional intensity function

$$\lambda_i(t|I_t, H_t) = \lim_{\Delta \rightarrow 0} \frac{P(N_i(t + \Delta) - N_i(t) = 1|I_t, H_t)}{\Delta},$$

where $N_i(t)$ is the cumulative spike count of unit i . $\lambda_i(t|I_t, H_t)$ represents the conditional probability density that a spiking event occurs at unit i . I_t and H_t stand for input history and spike history, respectively. For small Δ , the probability of a spike can be computed by Bernoulli approximation to the point process (Daley and Vere-Jones 2003; Brown et al. 2003)

$$P(\delta_{N_i(t)} = 1|I_t, H_t) \approx \lambda_i(t|I_t, H_t) \Delta,$$

where $\delta_{N_i(t)} = N_i(t + \Delta) - N_i(t)$. If λ is a function of I_t , H_t and parameter set θ , the log-likelihood \mathcal{L} of a sample count path for a node (electrode) i is expressed as in Chornoboy et al. (1988):

$$\begin{aligned} \mathcal{L}_i(\theta|N_i) &= \log p(N_i|\theta) \\ &= \int_0^T \log \lambda_i(t|I_t, H_t, \theta) dN_i(t) \\ &\quad - \int_0^T \lambda_i(t|I_t, H_t, \theta) dt, \end{aligned}$$

where $p(N_i|\theta)$ is the conditional probability density for the count path. The instantaneous log-likelihood (Brown et al. 2001) of a point process trajectory (spike train) is defined as

$$\ell_i(t, \theta) = \log \lambda_i(t|I_t, H_t, \theta) \frac{dN_i(t)}{dt} - \lambda_i(t|I_t, H_t, \theta) \quad (1)$$

Note that

$$\mathcal{L}_i(\theta|N_i) = \int_0^T \ell_i(t, \theta) dt.$$

With a conversion to discrete time, i.e. $t \rightarrow n, \Delta \rightarrow 1$, instantaneous log-likelihood becomes,

$$\ell_i^n = \log \lambda_i^n \delta_{N_i}^n - \lambda_i^n, \quad (2)$$

where n is the discrete time step index. Any parameter θ_j of the model is then learned then by the standard gradient descent learning.

$$\theta_j^{n+1} \leftarrow \theta_j^n + \eta \left. \frac{\partial \ell_i^n}{\partial \theta_j} \right|_{\theta_j = \theta_j^n},$$

where η is the learning rate. For the sake of readability, we will leave the calculation points of partial derivatives out and simply use the notation $\theta_j^{n+1} \leftarrow \theta_j^n + \eta \frac{\partial \ell_i^n}{\partial \theta_j}$.

The quality of point process modeling strongly depends on the expressive power of function $\lambda(t|I_t, H_t, \theta)$. Linear or log-linear models might be preferable for successful adaptation by gradient descent, whereas non-linear functions allow for more expressive models. In the current work, we model this function by an Echo State Network, which inherently incorporates input and network history into the instantaneous network states.

4 Point process modeling with ESN and log likelihood propagation

We relate input and reservoir state to the conditional intensity function in discrete time as

$$\lambda(t|I_t, H_t, \theta) \approx \lambda^n = f^{out}(W^{out}[u^n; x^n]), \quad (3)$$

where λ^n is an L -dimensional vector of conditional intensity estimations for time step n . We use an exponential f^{out}

$$f^{out}(\xi) = \exp(A \xi), \quad (4)$$

where $0 \leq A \leq 1$ is a constant. Exponential functions were already applied in point process modeling and shown to have desirable properties such as avoiding local minima in Generalized Linear Models (GLM) (Paninski 2004).

ESN readout learning can be trivially adapted to point process data, where the output of the ESN estimates the conditional intensity function in batch mode

$$W_{ij}^{out} \leftarrow W_{ij}^{out} + \eta \frac{\partial \mathcal{L}_i}{\partial W_{ij}^{out}}, \quad (5)$$

and also in online mode

$$W_{ij}^{out} \leftarrow W_{ij}^{out} + \eta_{out} \frac{\partial \ell_i^n}{\partial W_{ij}^{out}}. \quad (6)$$

We are going to provide the expansion of the above learning rule and its extension to reservoir adaptation assuming an online mode. Combining Eqs. (2), (3), and (4) yields

$$\begin{aligned} \frac{\partial \ell_i^n}{\partial W_{ij}^{out}} &= A x_j^n \delta_{N_i}^n - A x_j^n f^{out}(\xi^n) \\ &= A x_j^n \delta_{N_i}^n - A x_j^n \lambda_i^n, \end{aligned} \quad (7)$$

where $\xi^n = W^{out}[u^n; x^n]$.

4.1 Reservoir adaptation

One interesting question is whether it is sufficient to learn the output weights or whether one needs to adapt the reservoir using the point process log-likelihood as the fitness criterion. In the current work, we adapt recurrent and feed-forward reservoirs with one step propagation of the point process log-likelihood into the reservoir and compare the results to non-adaptive reservoirs.

Any connectivity weight W_{kl} within the reservoir can also be adapted with the gradient descent rule,

$$W_{kl}^n \leftarrow W_{kl}^{n-1} + \eta_{res} \frac{\partial \ell^n}{\partial W_{kl}}. \tag{8}$$

As we assume the output channels are mutually independent given the parameters,

$$\frac{\partial \ell^n}{\partial W_{kl}} = \sum_{i=1}^L \frac{\partial \ell_i^n}{\partial W_{kl}}. \tag{9}$$

Obviously, ℓ_i^n has a long term dependency on W_{kl} due to reservoir memory, i.e. ℓ_i^n not only depends on W_{kl}^{n-1} but also on W_{kl}^{n-h} , where $h > 1$. However, for the sake of computational efficiency, and in view of the fact that long term partial derivatives tend to vanish exponentially with respect to time (Bengio et al. 1994), we utilize a one-step propagation of the instantaneous log-likelihood into the reservoir.

$$\frac{\partial \ell_i^n}{\partial W_{kl}} \approx \frac{\partial \ell_i^n}{\partial W_{kl}^{n-1}}. \tag{10}$$

Note that if neuron ψ projects to the readouts, any connection to ψ is updated with one-step propagation. As every reservoir neuron is projecting to the readout units, every reservoir connection is updated with the learning rule.

$$x^n = \left(1 - \frac{1}{\tau}\right) x^{n-1} + \frac{1}{\tau} f(W^{in} u^n + W^{n-1} x^{n-1}), \tag{11}$$

W_{kl}^{n-1} contributes to ℓ_i^n for all i, k and l through x_k^n . We can now utilize the chain rule,

$$\frac{\partial \ell_i^n}{\partial W_{kl}^{n-1}} = \frac{\partial \ell_i^n}{\partial x_k^n} \frac{\partial x_k^n}{\partial W_{kl}^{n-1}}. \tag{12}$$

Partial derivatives with respect to state variables can be computed by combining Eqs. (2), (3), and (4).

$$\frac{\partial \ell_i^n}{\partial x_k^n} = A W_{ik}^{out} \delta_{N_i}^n - A W_{ik}^{out} \lambda_i^n. \tag{13}$$

Again following Eq. (11),

$$\frac{\partial x_k^n}{\partial W_{kl}^{n-1}} = \alpha_k \frac{\partial f_k(a_k^n)}{\partial a_k^n} x_l^{n-1}, \tag{14}$$

where a_k^n is the k -th element of the vector $W^{inp} u^n + W^{n-1} x^{n-1}$ and $f_k(a_k^n)$ is the k -th element of the resulting state vector. We define α_k as the inverse of the time constant for reservoir unit k , $0 < \alpha_k = 1/\tau_k < 1$. With Eqs. (9), (10), (12), (13) and (14), the update rule in Eq. (8) is completed.

4.1.1 Adapting neuronal time constants

Gradient descent is also used for adapting the time constants of the reservoir neurons. Note that in this case time constant is not a global parameter anymore. Let $\alpha_j^n = 1/\tau_j^n = g(\alpha_j^n) = 1/(1 + \exp(\alpha_j^n))$, to have a continuity over all α_j^n and to keep them in the desired range.

$$\frac{\partial \ell_i^n}{\partial \alpha_j^{n-1}} = \frac{\partial \ell_i^n}{\partial x_j^n} \frac{\partial x_j^n}{\partial \alpha_j^{n-1}} \tag{15}$$

$$\frac{\partial x_j^n}{\partial \alpha_j^{n-1}} = \left\{ -x_j^{n-1} + f_j(W^{inp} u^n + W^{n-1} x^{n-1}) \right\} \frac{dg(\alpha_j^{n-1})}{d\alpha_j^{n-1}} \tag{16}$$

$$\alpha_j^n \leftarrow \alpha_j^{n-1} + \eta_\alpha \frac{\partial \ell^n}{\partial \alpha_j^{n-1}}, \tag{17}$$

The complexity of the readout-only learning is $O(LN)$ for each time step, where L is the number of readout units. Running the trained ESN on the test input stream has also a complexity of $O(LN)$ for sparse reservoirs, where the number of connections linearly scales with the number of reservoir units. All the reservoirs we use belong to this sparse type. In this case, the complexity of the one-step log-likelihood propagation is also $O(LN)$.

4.2 Existence of local maxima and confidence intervals

Due to deep architecture and reservoir transfer function, we cannot guarantee that gradient descent in the

reservoir parameters yields uniquely true, i.e. globally optimal, parameters. Point process log-likelihood is not concave with respect to the whole set of parameters. Reservoir adaptation, as a result, is not a tool for finding a globally optimal equivalent of a given BNN. The quality of reservoir adaptation is evaluated based on its improvement on the predictive performance over fixed reservoirs. For a fixed reservoir, on the other hand, learning reduces to readout-only learning, and hence, to adapting the parameters of a Generalized Linear Model (McCullagh and Nelder 1989; Paninski 2004), which maps reservoir states and input to the conditional intensity. In this case, point-process log-likelihood is a concave function of readout parameters (see Appendix 2) and does not have non-global local maxima with respect to them (Paninski 2004). This shows that gradient descent in the readout parameters will asymptotically result in a global maximum. The absence of local minima holds for readout parameters under a fixed reservoir or if an adaptive reservoir is fixed after some epochs of training. Note that we take an online (stochastic) gradient descent approach in this work. Although online gradient descent takes a stochastic path in the parameter space for maximization of the log-likelihood, empirical evidence suggests that the average behavior of gradient descent is not affected by online updates (Bottou 2004).

Upon training, the observed Fisher Information Matrix can be used to approximate the covariance matrix and confidence intervals on the readout parameters,

$$\Sigma_{out}^{-1} = -\nabla_{W_{out}}^2 \mathcal{L}(W^{out}|N),$$

where Σ_{out} is the estimation of the covariance matrix and $\nabla_{W_{out}}^2 \mathcal{L}(W^{out}|N)$ is the Hessian of the log-likelihood. The Hessian matrix is computed at the final point estimates of the readout parameters. Confidence intervals are computed using the standard deviations obtained from the diagonal of the covariance matrix. It should be noted that we do not perform a full probabilistic learning of the parameters. We approximate the distribution on the readout parameters by the asymptotic normal distribution of the *maximum likelihood* estimator (Davison 2003; Pawitan 2001).

5 Evaluation of the learned models

We compared our results with feed-forward and recurrent architectures to the baseline method where only the readout weights were adapted. For different ESN types and architectures, we comparatively evaluated

their capabilities on modeling BNN *stimulus-response* relations by testing the predictive power of the ESN on the observed spike trains. The continuous ESN output signal was tested for compatibility with the actual observed spikes.

Receiver Operating Characteristic (ROC) curve analysis was employed to test the quality of prediction. ROC curves are extensively used in signal detection theory and machine learning to evaluate prediction performance on binary variables. In a binary classification problem (here spike or no spike), a ROC curve is a plot of true positive rate vs. false positive rate. True positive rate is defined as

$$TPR = \frac{\# \text{ True Positives}}{\# \text{ All Positives}}.$$

Similarly, false positive rate is defined as

$$FPR = \frac{\# \text{ False Positives}}{\# \text{ All Negatives}}.$$

As the ESN output is continuous, true positive rate and false positive rate will vary with respect to a moving threshold. An example ROC curve is shown in Fig. 6 (bottom). In contrast to classification accuracy, the area under a ROC curve (AUC) is more robust with respect to prior class probabilities (Bradley 1997). An AUC around 0.5 indicates a random prediction whereas an AUC for a perfect prediction will equal 1.0.

6 Experimental results

We employed Echo State Networks with different types and sizes of reservoirs on spike prediction tasks, where the spikes were recorded from simulations of random cortical networks and from cultured networks of cortical neurons. We investigated whether ESNs successfully predict output spikes of BNNs when they are presented with the same input streams.

6.1 Simulations of random cortical networks

We simulated 10 surrogate cortical neural networks of 1000 neurons each with 0.1 random connectivity. We used the Izhikevich neuron model and included spike-timing dependent plasticity (STDP) in our simulations (Izhikevich 2006). At each 5 ms, one of the 100 input channels was randomly selected from a uniform distribution and a pulse of 5 ms width was sent to the network. Each input channel had excitatory projections

Table 1 Areas under ROC curves (in %) for the prediction of activity in 10 simulated biological neural networks with fixed recurrent (R-*fxd*), feed-forward adaptive (FF-*adp*) and recurrent adaptive (R-*adp*) reservoirs

Res. size:	100			500			1000		
Res. type:	R- <i>fxd</i>	FF- <i>adp</i>	R- <i>adp</i>	R- <i>fxd</i>	FF- <i>adp</i>	R- <i>adp</i>	R- <i>fxd</i>	FF- <i>adp</i>	R- <i>adp</i>
BNN1	74.0	78.3	76.8	81.3	81.4	80.6	81.7	82.3	82.1
BNN2	76.7	80.6	78.1	84.1	84.8	84.1	84.7	85.5	85.3
BNN3	69.4	73.5	70.7	77.0	77.6	77.3	77.4	78.1	78.6
BNN4	76.6	78.9	78.0	81.4	81.6	81.2	82.3	82.9	82.6
BNN5	74.9	78.0	77.4	81.5	81.4	81.2	81.7	82.2	82.2
BNN6	79.0	82.0	81.3	84.8	84.9	84.7	84.9	85.3	85.5
BNN7	72.0	74.4	74.1	78.2	79.0	78.7	79.2	79.5	79.9
BNN8	71.7	73.0	72.3	76.8	77.5	76.5	78.1	78.1	78.5
BNN9	72.3	74.9	74.6	78.5	78.4	78.4	78.8	79.2	79.1
BNN10	79.0	80.7	79.0	83.8	83.9	83.4	83.9	84.5	84.3
mean	74.6	77.4●*	76.2●	80.7	81.1●*	80.6	81.3	81.8●	81.8●

● denotes statistical significance for having outperformed the non-adaptive recurrent reservoir of the same size. * denotes a statistical significance for having outperformed the other two reservoir types of the same size (signed-rank test, $p < 0.05$)

to 10 % of the neurons in the network. The networks were simulated for 2 h in real time with STDP and 0.5 h in real time with frozen synapses thereafter. In each network, the spikes from a randomly selected excitatory neuron were recorded with 5 ms discretization. We then trained ESNs to estimate the conditional

intensity of the selected neuron’s spiking process from the history of the input pulses.

LI-ESNs with three different reservoir types of various sizes were used for this task, namely recurrent fixed recurrent reservoirs (R-*fxd*), recurrent adaptive reservoirs (R-*adp*) and feed-forward adaptive (FF-*adp*)

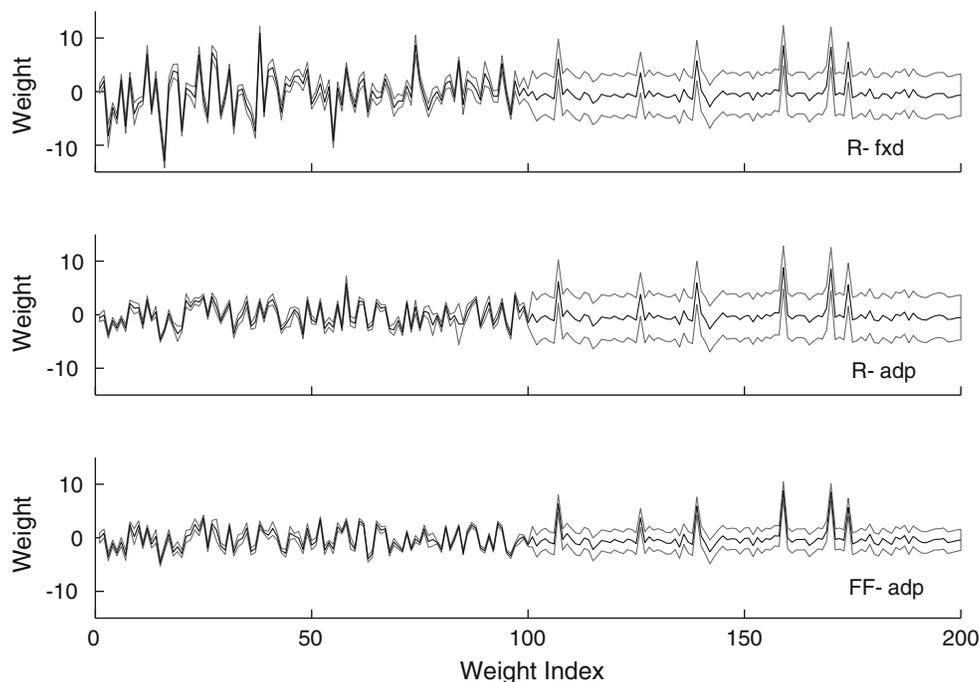


Fig. 5 Confidence intervals on the readout parameters of 3 reservoir types of size 100 for a selected biological neural network. The learned readout weights and 99% confidence intervals ($\pm 2.576\sigma$) are shown. Confidence intervals are computed by assuming a multivariate normal distribution and an infinite prior covariance matrix. Hence, inverse of the observed Fisher information matrix

is used to compute confidence intervals. The weights indexed from 1 to 100 are reservoir-to-output parameters, whereas the weights indexed from 101 to 200 are direct input-to-output parameters. Greater uncertainty for input-to-output parameters are noticeable, which results from low density of the input pulses per channel

reservoirs. Reservoir connectivity and each individual time constant were adapted using the method described in Section 4. For each of the 10 surrogate biological neural networks, we used reservoirs of sizes 100, 500 and 1000. A separate random sparse reservoir for each size and for each surrogate biological network was generated, where each reservoir unit was connected to 10 other units. For each random reservoir, a feed-forward reservoir was generated by inverting the edges that cause recurrence as described in Fig. 4. Note that the same sized feed-forward and the recurrent reservoirs for the same biological neural network had the same topology apart from edge directions. This yielded 60 different reservoirs for the whole experiment.

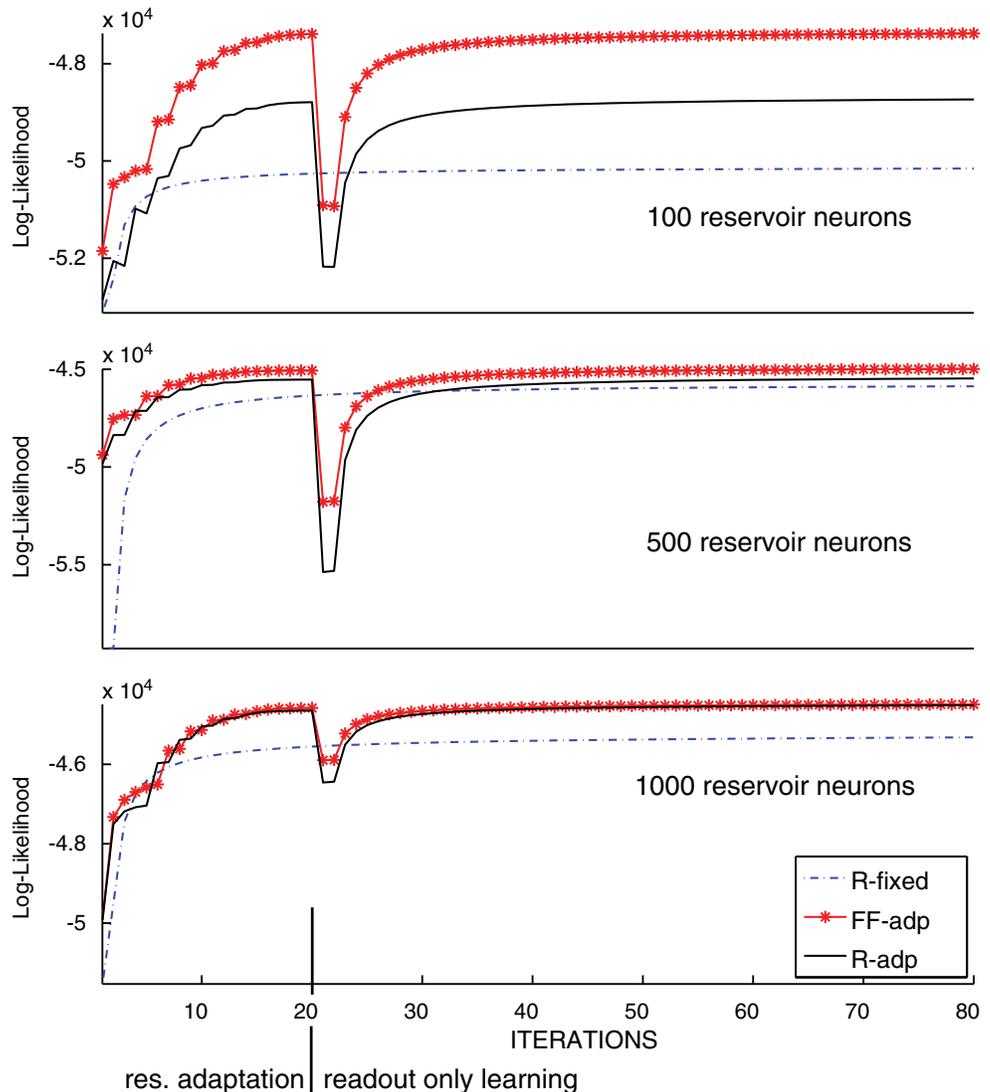
From each of the 10 BNN simulations we recorded data for 30 min in real time. Using a discretization time bin of 5 ms, the simulation yielded time series data for 360,000 time steps, of which 328,000 time steps for were

used for training and 30,000 time steps were used for testing in each sub-experiment. For the adaptive reservoirs, the training phase included 20 full adaptation iterations of the reservoirs and 60 readout adaptation iterations. One iteration covered one forward traversal of the training data in time. No learning was performed on the test data.

Results of the ROC curve analysis show that the ESN approach performs well in conditional intensity estimation (Table 1). For all the experiments on 10 different BNNs, the estimated conditional intensity predicts output firing far better than random prediction. For BNN2, point estimates and uncertainty measures of readout parameters after readout-only-training were also presented. (Fig. 5, see Section 4.2 for computation of uncertainty measures).

We further investigated whether the predictive power of ESNs with feed-forward adaptive reservoirs

Fig. 6 Changes of the training log-likelihood with the number of iterations for reservoirs with different number of neurons (N) and reservoir types (color coded). The performance of the reservoirs is shown for fixed recurrent (blue, dashed), adaptive recurrent (black, continuous) and adaptive feed-forward types (red, *)



of leaky integrator neurons is comparable to that of recurrent reservoirs. Adaptive reservoirs are in general better than non-adaptive reservoirs. This difference, however, vanishes with increasing reservoir size (Table 1). In these experiments, feed-forward adaptive reservoirs significantly outperformed the non-adaptive recurrent reservoirs of all sizes. Furthermore, they significantly outperformed recurrent adaptive reservoirs for sizes of 100 and 500. Although adaptation brought significant improvement also to recurrent reservoirs for sizes 100 and 1000, feed-forward adaptation performed in general better than recurrent adaptation in estimating the conditional intensity. Small reservoirs (i.e. for $N = 100$) of the feed-forward architecture were drastically superior to the recurrent reservoirs.

One can gain some insight on the comparative performances of different reservoirs by looking at the development of the log-likelihood during training (Fig. 6). Note that we adapted the reservoir connectivity and the time constants only in the first 20 iterations. The remaining 60 iterations included only learning of readout parameters. This allowed for observing the effects of two learning phases separately. The log-likelihood clearly increases with fewer iterations in adaptive reservoirs. A sudden drop of training log-likelihood is remarkable at the transition from the full training phase to the readout training phase. Although this transition

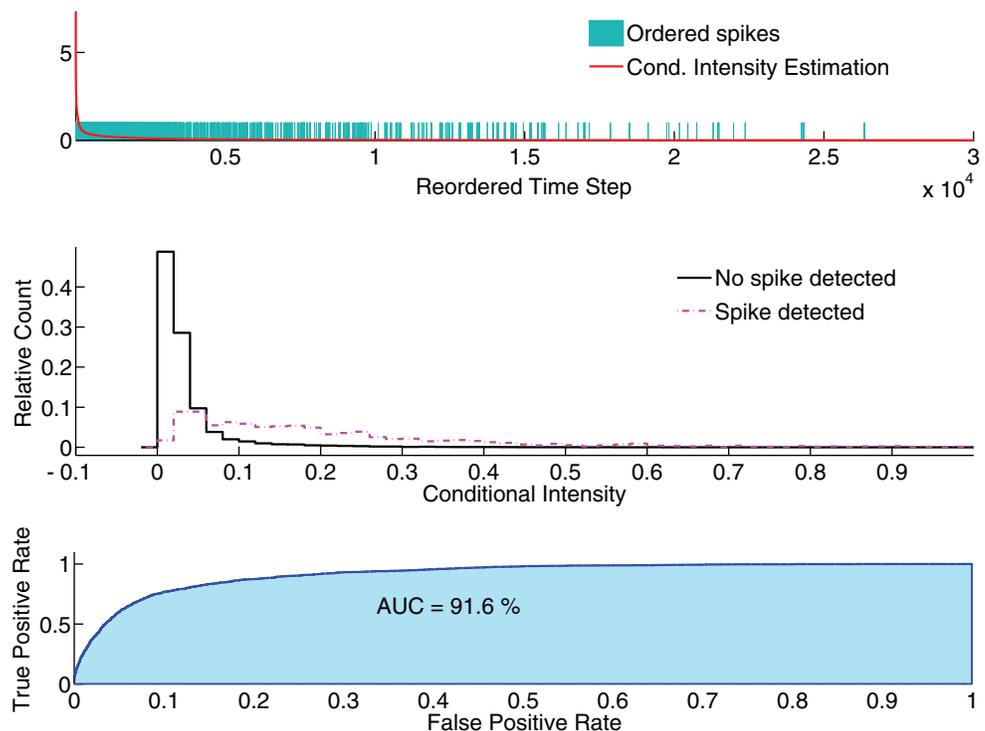
significantly reduces the fitness and might make the adaptive reservoirs worse than non-adaptive ones, previous full reservoir training pays out in the upcoming iterations.

Note that Table 1 refers only to the prediction of output spikes from input pulse trains. If the spike history of the output neuron is also taken into account, the prediction performance increases considerably. In this case, the output spike history is also fed into the ESN through an additional input channel. For BNN2, we visualize the conditional intensity estimation that takes also the output spike history into account (Fig. 7). Note the increased area under the ROC curve when compared to the second row in Table 1. Cross-correlation coefficients around 0 time lag between the conditional intensity estimation and the target spike train also improve when the output spike history is used (Fig. 8), indicating an increased similarity of estimations and observations. Cross-correlation coefficients are computed as

$$c_{s\lambda}(m) = \frac{1}{T-1} \frac{\sum_n [s(n+m) - \bar{s}] [\lambda(n) - \bar{\lambda}]}{\sigma_s \sigma_\lambda},$$

where m is the time lag, s is the binary valued discrete-time signal for the target spike train and $\lambda(n)$ is the

Fig. 7 Estimated conditional intensity for a selected biological neural network (BNN2) using input history and the output spike history together. Conditional intensity estimations λ for all time steps in the testing period are shown in decreasing order (top). A bar is shown if there was indeed a spike observed in the corresponding time step (top). Distributions of conditional intensity, λ , for time steps with observed spikes and without spikes (middle). By a varying threshold on λ , true positive rates vs. false positive rates can be calculated (bottom)



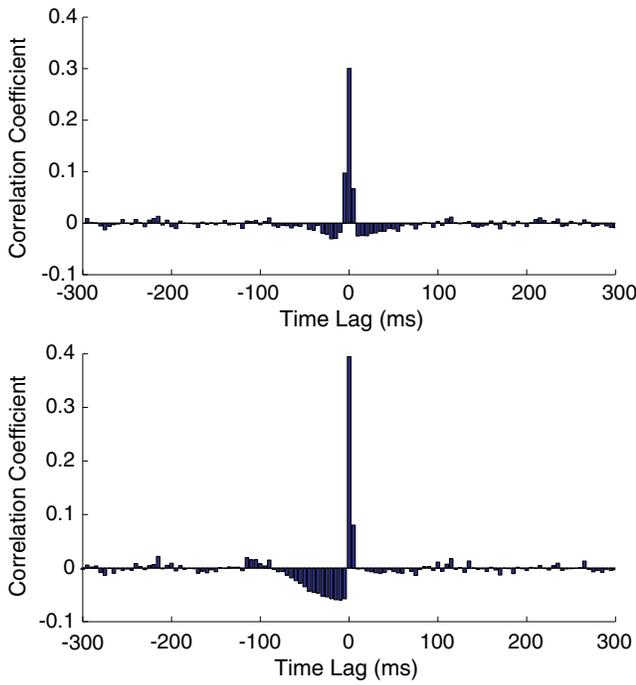
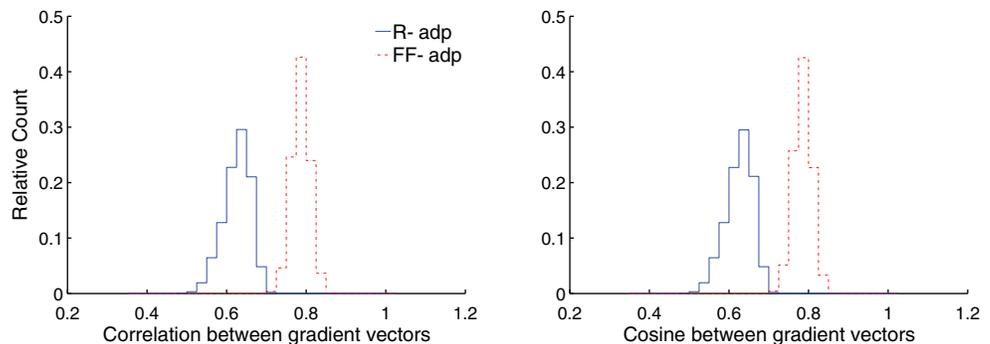


Fig. 8 Cross-correlation coefficients between the conditional intensity estimation and the observed spike patterns for the output neuron of a selected biological neural network (BNN2) with feed-forward ESN adaptation for a reservoir of 1000 neurons. Using only the input stream, AUC = 85.5% (top). Using the input stream and the spike history of the output neurons, AUC = 91.6% (bottom). Each bar shows the correlation coefficient for 1 time step of the ESN (5 ms). Around 0 time lag, the conditional intensity that uses the spike history has a higher correlation with the observed spikes. Note that there is negative correlation for some negative time lags ($-90 < \text{timelag} < 0$). This results from the fact that conditional intensity estimation is very low after an observed spike since the algorithm learns the negative auto-correlation of the output spikes for this time lag interval. In other words, the learned model is reflecting the inter-spike interval in its output

conditional intensity estimation at time step n . \tilde{s} and $\tilde{\lambda}$ stand for mean values, whereas σ_s and σ_λ denote standard deviations. T is the data length in time steps.

Fig. 9 Distributions of the correlations (left) and cosines (right) between fully recursive and one-step computations of gradient vectors, $\nabla \ell^n$ and $\nabla^{n-1} \ell^n$, respectively. Results are shown for a selected sparse recurrent reservoir of size 100 and its feed-forward version



As for ROC curve areas, the peak around zero lag indicates the similarity between the conditional intensity estimation and the reference spike train.

6.1.1 Approximating gradient vectors with one-step propagation

In order to investigate the consequences of one-step approximation to gradient descent, we compared the gradient vectors obtained by full gradient computation and those obtained by one-step approximation. Full gradient computation would replace Eqs. (14) and (16) as

$$\frac{\partial x_j^n}{\partial W_{kl}} = (1 - \alpha_j) \frac{\partial x_j^{n-1}}{\partial W_{kl}} + \alpha_j \frac{\partial f_j(a_j^n)}{\partial a_j^n} \left[\delta_{jk} x_l^{n-1} + \sum_m W_{jm} \frac{\partial x_m^{n-1}}{\partial W_{kl}} \right]$$

and

$$\frac{\partial x_j^n}{\partial \alpha_k} = (1 - \alpha_j) \frac{\partial x_j^{n-1}}{\partial \alpha_k} + \left[\alpha_j \frac{\partial f_j(a_j^n)}{\partial a_j^n} \sum_m W_{jm} \frac{\partial x_m^{n-1}}{\partial \alpha_k} \right] + \delta_{jk} \left[-x_j^{n-1} + f_j(a_j^n) \right],$$

respectively. The resulting learning rule would be an adaptation of the Real Time Recurrent Learning (Williams and Zipser 1989) for point processes, which is computationally very expensive for our data and reservoir sizes. It is, however, still viable to compare the gradients obtained by one-step propagation to those obtained by full gradient computation for a limited

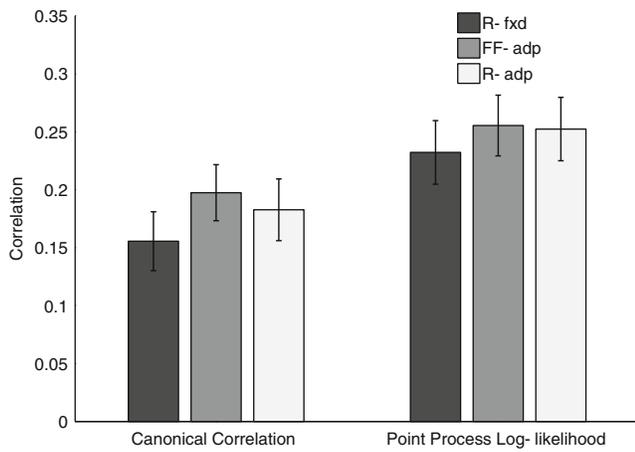


Fig. 10 Correlation of reservoir activity with the observed spike activity for a selected recurrent reservoir with 100 neurons and its feed-forward version. Canonical correlation analysis (CCA) results also indicate that output spikes are encoded in reservoir activity. Point process readout learning proves more efficient in relating the reservoir activity to output spikes than CCA

number of time steps. Here, we compared gradient vectors of the log likelihood with respect to all reservoir parameters for 19,000 time steps. We computed the

correlation and the cosine between full and one-step gradients for each time step separately,

$$c^n = \frac{1}{R-1} \frac{\sum_r \left(\frac{\partial \ell^n}{\partial W_r^{n-1}} - \mu_{one}^n \right) \left(\frac{\partial \ell^n}{\partial W_r} - \mu_{full}^n \right)}{\sigma_{one}^n \sigma_{full}^n}$$

$$\cos(\Omega^n) = \frac{\langle \nabla^{n-1} \ell^n, \nabla \ell^n \rangle}{|\nabla^{n-1} \ell^n| |\nabla \ell^n|},$$

where $\nabla^{n-1} \ell^n$ and $\nabla \ell^n$ are the gradient vectors at time step n obtained by one-step propagation and by full gradient computation, respectively; $\frac{\partial \ell^n}{\partial W_r^{n-1}}$ and $\frac{\partial \ell^n}{\partial W_r}$ are their r -th components; $\mu_{one/full}^n$ and $\sigma_{one/full}^n$ are the mean values and the standard deviations for the components of the gradient vector; c^n and Ω^n are the correlation and the angle between the full and one-step gradient vectors at time step n , respectively.

The distribution of the correlations and cosines provides insight to the consequences of one-step approximation in feed-forward and recurrent reservoirs. Over 19,000 time steps, the correlations between the one-step

Fig. 11 Normalized input triggered averages (see Eq. (18)) for 20 different input pulses on 40 different reservoir neurons (*background, gray*). Randomly selected 30 combinations, are highlighted (*darker, red*) for three different reservoir types. Adaptive reservoirs display more diffused memory traces, i.e. their time lags for the peaks vary more

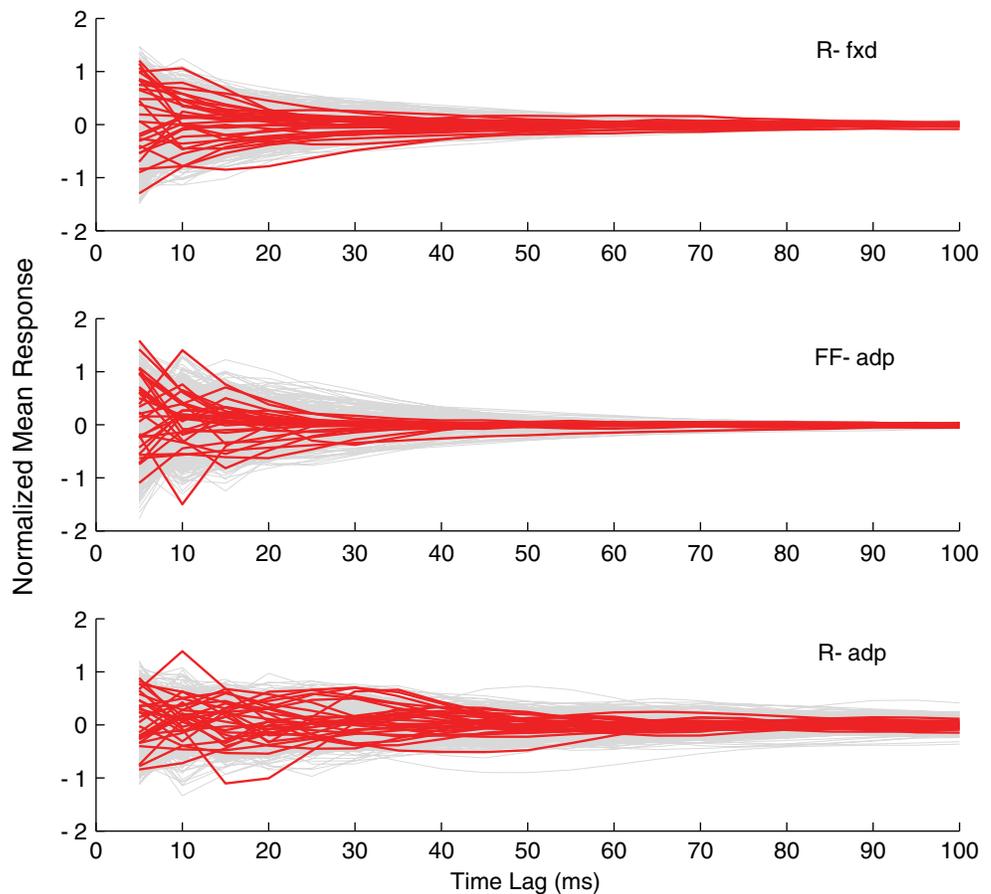
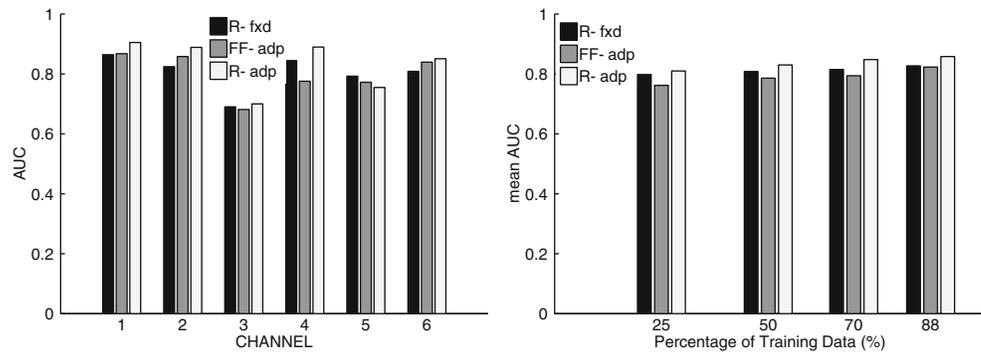


Fig. 12 Areas under ROC curves (AUC) for the event prediction task at the active output electrodes (*left*) and aggregated AUC for different ratios of training data (*right*) using fixed recurrent (R-fxd), feed-forward adaptive (FF-adp) and recurrent adaptive (R-adp) reservoirs, all with 500 neurons



and full gradient vectors in a selected random reservoir are distributed around a mean value of 0.61 with a standard deviation of 0.034. The mean and standard values are 0.77 and 0.022 for the corresponding feed-forward reservoir (Fig. 9). Cosines constitute a very similar distribution, as the gradient vector components for each time step have approximately a mean value of 0. Strong correlations between the full and one-step gradients deliver additional explanation for the better performance adaptive reservoirs. The fact that the log-likelihood propagates to every reservoir parameter in one-step (due to full reservoir-to-output connectivity) and that one-step gradient is the dominant component of the full gradient information are possible reasons for the detected similarity. Stronger correlations for feed-forward reservoirs are in agreement with the rel-

ative better performance of the feed-forward reservoir adaptation.

6.1.2 Information encoded in reservoir activity

A canonical correlation analysis of the reservoir activity and the output spiking activity also revealed that output spikes are encoded in the reservoir activity. Prior to readout-only learning, we recorded the reservoir states for one traversal of the training data and performed a canonical correlation analysis (CCA) (Chatfield and Collins 2000) with the observed training spikes. We then used parameters obtained by CCA for relating reservoir states to output spikes in the test data and detected maximum correlations for all of the 10 experiments in Table 1. Although not as good as in readout

Fig. 13 ROC curve analysis of the event timing prediction for active output electrode 1 using a recurrent adaptive reservoir with 500 neurons without using the spike history. Conditional intensity estimations λ for all time steps in the testing period are shown in decreasing order (*top*). A bar for each observed spike in the corresponding time step (*top*). Distributions of conditional intensity, λ , for time steps with and without observed spikes (*middle*). By a varying threshold on λ , true positive rates vs. false positive rates can be calculated (*bottom*)

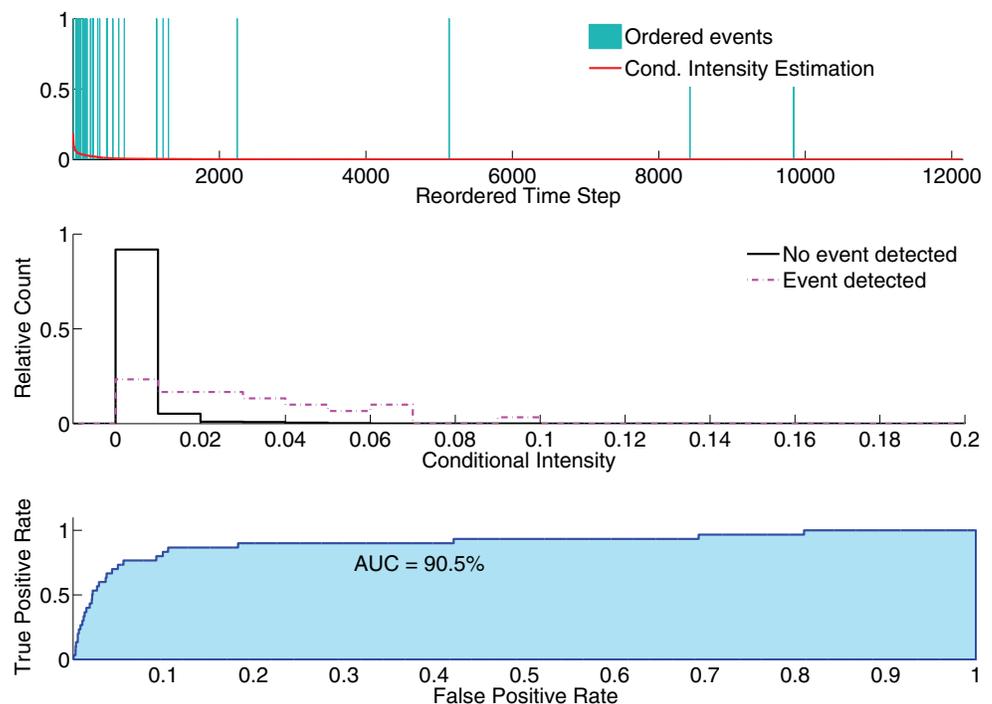


Table 2 Areas under ROC curves for the event timing prediction task aggregated over active output electrodes

Res. size	10	30	50	100	300	500	1000			
Without history										
R-fxd	59.9	57.9	74.3	78.9	78.3	82.7	83.2			
FF-adp	59.3	64.1	72.5	76.3	83.8	82.3	83.9			
R-adp	63.9	68.9	69.2	76.2	82.3	85.8	84.2			
With history										
R-fxd	57.2	62.4	69.9	76.7	79.8	82.5	82.1			
FF-adp	57.3	66.4	69.6	76.6	80.1	82.1	83.4			
R-adp	59.8	68.1	74.3	79.7	79.4	85.1	83.6			
Rate based prediction										
Kernel(ms):	3	5	10	20	30	50	70	100	150	250
AUC:	60.7	64.0	69.3	75.4	73.9	71.3	69.9	66.5	61.2	54.6

learning, CCA also reveals the correlation of the reservoir activity with output spikes (Fig. 10), indicating the spike encoding in the reservoir activity.

In addition to CCA of reservoir states and output spikes, we further visualized the memory traces of the input pulses in reservoir states. We compared how past inputs are reflected in the reservoir activity for different reservoir types by considering input triggered averages (ITA). We define the input triggered average for an input-neuron pair as

$$ITA_{hj}(m) = \frac{\sum_{n=1}^T u_h^n x_j^{n+m}}{\sum_{n=1}^T u_h^n},$$

where m is the time lag, u_h^n is the input pulse from channel h at time step n (1 or 0), x_j^{n+m} is the state of the j -th neuron at time step $n + m$. The normalized ITA is then

$$\widehat{ITA}_{hj}(m) = \frac{ITA_{hj}(m) - \bar{x}_j}{\sigma_{x_j}}, \tag{18}$$

where \bar{x}_j and σ_{x_j} are the mean value and the standard deviation for x_j , respectively. Figure 11 depicts the \widehat{ITA}_{hj} curves for 800 input neuron pairs in 3 different reservoir types for a selected biological neural network. Memory traces for adaptive reservoirs, especially for

Fig. 14 Cross-correlation coefficients between the conditional intensity estimation and the observed output events for a selected (1) active output electrode without using the output spike history (res. size of 500). Coefficients are shown for fixed recurrent (a), feed-forward adaptive (b), and recurrent adaptive reservoirs (c). Each bar shows the correlation coefficient for 5 time steps of the ESN (5 ms). As a baseline a prediction based on event rate (without an ESN, conditional intensity proportional to event rate) is also shown (d). The recurrent adaptive reservoir performs better here, considering the height of the peak and the width of the cross-correlogram

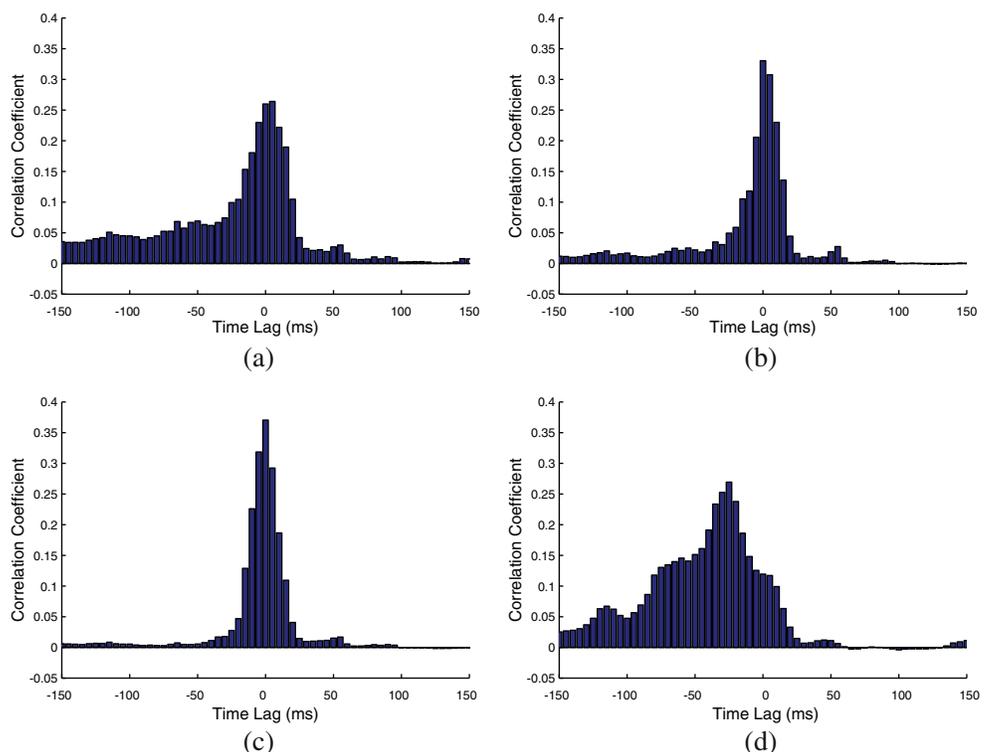


Table 3 Mean AUC (%) in the event timing prediction task in 3 different cultures

Channel	Culture 1			Culture 2			Culture 3		
	R-fxd	FF-adp	R-adp	R-fxd	FF-adp	R-adp	R-fxd	FF-adp	R-adp
All	81.6	83.1	83.6	82.7	82.6	82.2	66.7	66.4	67.1
Best	84.2	87.0	89.3	92.3	91.9	92.2	83.8	84.7	82.2
Worst	78.8	77.8	76.6	75.6	73.9	74.2	50.4	48.3	53.8

Results are aggregated over 10 random reservoirs with size 500 for all active output electrodes, for the best active electrode and for the worst active electrode

recurrent adaptive reservoirs, are more scattered over time. This increased heterogeneity in the responses is very likely a reflection of time constants in adaptive reservoirs.

6.2 Prediction of spontaneous events in neuronal cultures

To test our approach on living neural networks, we aimed at predicting spontaneous events in neuronal cultures. We defined an event as a group of spikes recorded by the same MEA electrode, whose maximum inter spike interval is less than 60 ms. The time of the first spike in the group is regarded as the event time. It should be noted again that the activity in neuronal cultures is typically a sequence of network bursts. If there were at least 100 ms between two successive events, we regarded them as parts of different bursts. These criteria mostly excluded isolated spikes from network bursts and clustered fast repetitive firings at an electrode, e.g. a cellular burst, into a single event. We recorded spikes from a neuronal cell culture for 30 min, detecting bursts and events according to the above definition. We used the data for training except for the last 200 s, which were reserved for testing. We selected approximately 3/4th of the 60 MEA electrodes and treated their events as the input stream; and the remaining electrodes as output. This selection was based on spatial order of the electrodes, i.e. input and output electrodes occupied

two distinct areas in the culture. If an electrode had never recorded spikes in the training event train, it was regarded as inactive and was excluded from the learning and testing processes. The evaluation task was to estimate the conditional intensity of the output events for each output electrode at time step n using the total input event stream until time step n (1 ms bin size). Note that only very few events occur outside the bursts. Therefore, the algorithm performs learning and prediction only during bursts.

Based on estimates for all time steps during network bursts in the last 200 s, we evaluated the predictive performance of the learned ESN models using ROC curve analysis on the target events and estimated conditional intensity. We selected the electrodes that recorded at least 15 events within the evaluation window of 200 s.

Figure 12 (left) shows the AUC for each electrode using reservoirs of different types all with 500 neurons. Note that the prediction results in Fig. 12 is based only on input event streams, i.e. the event history of the output electrodes is excluded. Except for output electrode 3, all AUC measures are far above the random level. This supports the notion that most output neurons in the culture do not fire randomly but they rather follow patterns (Shahaf et al. 2008). Even electrode 3 is above the chance level but it has low predictability compared to other electrodes. Aggregated test data AUC over active electrodes (except electrode 3) for different ratios of training data is shown in Fig. 12 (right). As expected, AUC improved with more training data. The

Table 4 Pairwise comparison of the reservoir types and methods in the event timing prediction task in 3 different cultures using 10 random reservoirs all with size 500

	Culture 1			Culture 2			Culture 3		
	# output el.: 6			# output el.: 10			# output el.: 9		
	R-fxd	FF-adp	R-adp	R-fxd	FF-adp	R-adp	R-fxd	FF-adp	R-adp
R-fxd	0	0	1	0	1	3	0	2	1
FF-adp	2	0	0	1	0	2	1	0	2
R-adp	3	2	0	1	0	0	2	3	0

Each entry in the table denotes the number of electrodes, for which the method in the row significantly outperforms the method in the column (signed-rank test, $p < 0.05$)

Table 5 AUC (%) for the next-event prediction task aggregated over all active output electrodes in a selected neuronal culture (1)

Res. size	10	30	50	100	300	500	1000
R-ffd	66.2	71.6	73.7	79.4	82.1	82.9	83.3
FF-adp	71.3	77.9	79.8	81.5	82.5	83.3	83.1
R-adp	71.8	72.7	79.4	81.2	82.3	82.4	82.3

improvement, however, was not dramatic. For a selected output electrode the details of the ROC analysis is illustrated in Fig. 13.

Table 2 shows aggregated areas under ROC curves over all active electrodes except electrode 3 for a selected culture (Culture 1), without (top) and with (middle) using the output event history. Each entry in the table was obtained with a single reservoir of the corresponding size. As expected, the prediction performance usually increased with the reservoir size. Using the event history of the output electrodes did not improve the prediction performance, thus, no additional information could be found in the output event history. Note that utilizing the event history results in an approximately 25% increase in the input dimensionality. Possible redundancy of the event history with the already existing input might have caused a suboptimal performance of the learning algorithm, decreasing the prediction accuracy in several reservoirs. Feed-forward adaptive and recurrent adaptive reservoirs performed better than the fixed reservoirs for most reservoir sizes.

As a baseline for comparison purposes, we used the input event rate (regardless of the spatial information, i.e. without using the electrode number) for the conditional intensity estimation (Table 2, below). Although this method proved to have some predictive power especially for a rate kernel of 20 ms, it is obviously not as good as ESN predictors. Cross-correlation analysis reveals further information about comparative performances of different methods (Fig. 14). The higher peak and the smaller width of the cross-correlogram is an indicator for the better performance of recurrent adaptive reservoirs. To further compare different reservoir types, we experimented with 3 different cultures and employed 10 randomly generated reservoirs of size 500. We used the acyclic forms of the same reservoirs for feed-forward adaptation. In 2 of these 3 cultures (1 and 3), the recurrent adaptive reservoirs had the best average predictive power (Table 3). In the same 2 cultures, the prediction of the recurrent adaptive reservoirs significantly outperformed the others for several output electrodes (Table 4).

Fig. 15 Estimated conditional intensity for a selected output electrode (5) of the neuronal culture in the next-event-prediction task using a feed-forward adaptive reservoir with 500 neurons. Conditional intensity estimations λ for all time steps in the testing period are shown in decreasing order (top). A bar is shown if there was indeed a spike observed in the corresponding time step (top). Distributions of conditional intensity, λ , for time steps where there was an observed spike and where there was no spike (middle). By a varying threshold on λ , true positive rates vs. false positive rates can be calculated (bottom)

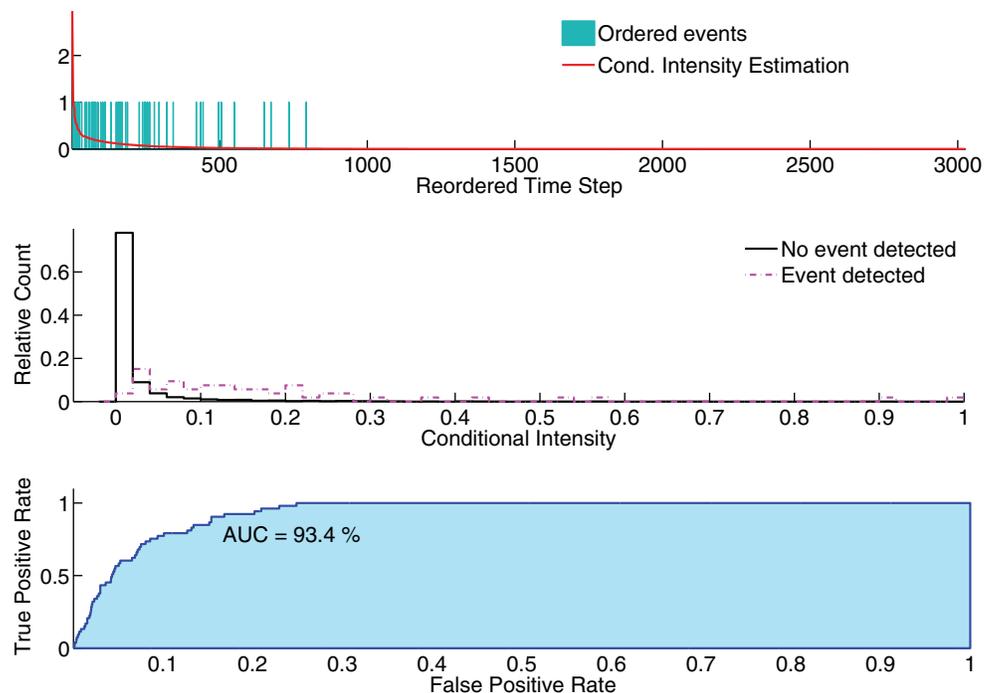


Table 6 Mean AUC (%) in the next-event prediction task in 3 different cultures

Channel	Culture 1			Culture 2			Culture 3		
	R-fxd	FF-adp	R-adp	R-fxd	FF-adp	R-adp	R-fxd	FF-adp	R-adp
All	81.9	82.6	82.2	75.2	75.7	75.4	72.7	73.1	72.8
Best	93.1	93.4	93.2	88	89.1	88.5	82	82.3	81.8
Worst	54.7	56.3	55.2	62.4	60.4	62.5	50.5	54.3	54.2

Results are aggregated over 10 random reservoirs with size 300 for all active output electrodes, for the best active electrode and for the worst active electrode

6.3 Next-event prediction in neuronal cultures

Recent findings indicate that the temporal order of the events in neuronal cultures carries significant information about the type of stimuli (Shahaf et al. 2008). In this experimental setting, we investigated whether the ESNs can model the structure in the temporal order of the events. We deleted timing information of the events from the previous setting and, as a result, we obtained data of temporally ordered events. More precisely, we used operational time steps for ESN analysis, i.e. each event appeared only in 1 time step and each time step contained only 1 event.

In this setting, we treated every electrode as output and used the whole event history for prediction. The task was again to estimate the conditional intensity of the output events for each electrode at time step n using the total event stream until time step n . Note that point process modeling is not necessarily the optimal tool for this data, where each abstract time step can contain only 1 event. The general framework taken in the current work, however, can still be employed for predictive modeling of temporal event orders. Again, ROC curve analysis indicates a good average prediction quality with larger reservoirs, feed-forward adaptive reservoirs outperforming the recurrent reservoirs up to sizes of 500 (Table 5, Fig. 15).

To further compare different reservoir types, 10 random reservoirs of 300 neurons together with their acyclic forms were tested on 3 different cultures (Table 6). The predictability varied with respect to the

electrode that recorded the event, with AUC ranging from 50.5% to 93.4%. In pairwise comparison of the 3 reservoir types, FF-adp reservoirs significantly (signed-rank test, $p < 0.05$) outperformed other methods for many electrodes (Table 7). Conversely, the number of electrodes, for which the other methods performed better, was much smaller.

7 Conclusion

The implications of our results are manifold. Firstly, our results indicate that reservoir computing is a potential candidate for modeling neural activity including neural encoding and decoding. With their expressive power for different activity measures (e.g. spike rates, correlations etc.), reservoir computing tools might help for analysis of neural data. In our experiments, ESNs of leaky integrator neurons proved successful for modeling *response-dynamics* (e.g. stimulus-response relations and spatio-temporal dynamics) of simulated and cultured biological neural networks.

On the methodological side, we showed that ESN learning algorithms can be adapted for *event data*, such as spikes or spike groups, using a point process framework. We proposed a reservoir adaptation method for *event data*, which can be used to adapt connectivity and time constants of the reservoir neurons. The experimental results indicate that reservoir adaptation can significantly improve the ESN performance over readout-only training.

Table 7 Pairwise comparison of the reservoir types and methods in the next-event prediction task in 3 different cultures using 10 random reservoirs all with size 300

	Culture 1			Culture 2			Culture 3		
	# output el.: 26			# output el.: 34			# output el.: 38		
	R-fxd	FF-adp	R-adp	R-fxd	FF-adp	R-adp	R-fxd	FF-adp	R-adp
R-fxd	0	5	5	0	6	5	0	6	11
FF-adp	11	0	8	22	0	20	12	0	17
R-adp	7	1	0	6	1	0	7	0	0

Each entry in the table denotes the number of electrodes, for which the method in the row significantly outperforms the method in the column (signed-rank test, $p < 0.05$)

We utilized feed-forward networks with leaky-integrator neurons as reservoirs with a comparable predictive power to recurrent reservoirs when trained with log-likelihood propagation. In modeling *stimulus-response* relations of simulated BNNs, feed-forward reservoir adaptation outperformed other methods up to 500 neurons. This outperformance was statistically significant. For the event timing prediction task in neuronal cultures, adaptive recurrent reservoir adaptation outperformed the other methods (in 2 of 3 cultures), whereas feed-forward adaptation were better in the next-event type prediction task in all 3 cultures. This might indicate that the type of encoding in neural systems (order or timing) favors different architectures for decoding. An analysis of the structure-coding relations, however, is beyond the scope of this note. Why feed-forward reservoir adaptation can work better than recurrent reservoir adaptation necessitates also more theoretical analysis. We manually optimized global parameters (spectral radius, learning rates and A) for recurrent fixed reservoirs. Recurrent adaptive reservoirs started from these values. Although we can think of no obvious disadvantage for recurrent reservoirs, feed-forward adaptation outperformed recurrent adaptation in some tasks. We experimentally showed that one-step propagation approximates the gradients better in feed-forward reservoirs than in recurrent ones. In our opinion, better structuring of the reservoir parameters, i.e. separation of the memory parameters from the reservoir connectivity, might further underlie the relative better performance of feed-forward adaptation. For instance, a small connectivity change in the recurrent adaptation might have a more dramatic effect on the reservoir memory than in case of feed-forward adaptation. Our findings on feed-forward networks are also in accordance with the recent work by Ganguli et al. (2008), Goldman (2009), Murphy and Miller (2009), who show that stable fading memory can be realized by feed-forward or functionally feed-forward networks and that feed-forward networks have desirable properties in terms of trade-off between noise amplification and memory capacity.

Acknowledgements This work was supported in part by the German BMBF (01GQ0420), EC-NEURO (12788) and BFNT (01GQ0830) grants. The authors would like to thank Luc De Raedt, Herbert Jaeger and anonymous reviewers for fruitful discussions and helpful feedback. We also appreciate the help of Steffen Kandler, Samorah Okujeni and Oliver Weihberger with culture preparations and recordings.

Open Access This article is distributed under the terms of the Creative Commons Attribution Noncommercial License which permits any noncommercial use, distribution, and reproduction

in any medium, provided the original author(s) and source are credited.

Appendix 1: Implementation details

For all the experimental settings, we generated recurrent sparse random reservoirs. Regardless of the reservoir size, each reservoir unit was connected to 10 other units. The initial connection strengths were drawn from a uniform distribution over the interval $[-0.5, 0.5]$. The entries of the input weight matrix W^{in} were drawn from a uniform distribution over $[-1, 1]$. We normalized the connectivity so that the maximum absolute eigenvalue, i.e. the spectral radius, was 1, with which the ESNs were experimentally detected to work well on all experiments. We additionally generated the feed-forward form of the each recurrent reservoir as described in Fig. 4. After changing from recurrent to feed-forward architecture, we did not manually modify the connectivity parameters. The leakage time constants, $\tau_j = 1/\alpha_j$, were initialized according to

$$\alpha_j = 1 / \left(1 + \exp(\alpha'_j) \right),$$

where α'_j were uniformly drawn from $[-1.5, 1.5]$.

Note that the initial value of the spectral radius and the initial range for time constants were manually arranged to work well with recurrent fixed reservoirs. Adaptive reservoirs used this fixed settings as starting parameters.

For fixed reservoirs, we used 80 epochs of readout-only training. Initially, we set $\eta_{out} = 0.7$ and gradually decreased the output learning rate with

$$\eta_{out} \leftarrow \frac{\eta_{out}}{E + 1},$$

where E was the epoch number. This was done for fast learning and convergence of the algorithm.

For adaptive reservoirs, we used 20 epochs for reservoir and readout training and 60 epochs for readout-only training. The readout-only learning employed the same time dependent η_{out} as the fixed reservoirs. For reservoir training, we set the initial values $\eta_{out} = \eta_{res} = \eta_{\alpha} = \eta = 0.2$ and decreased it with

$$\eta \leftarrow 0.5 \eta,$$

when the increase in the training log-likelihood dropped below 0.0003 per output channel and time step.

The parameter A in $f^{out}(\xi) = \exp(A \xi)$ was set to 0.2, which was manually found to work well for the selected learning rates.

The ESN learning algorithm described in this note was implemented in MATLAB and C++ languages.

The results reported here were produced by the C++ implementation. The source code is available under http://www.informatik.uni-freiburg.de/~guerel/PP_ESN.zip (currently only MATLAB).

Appendix 2: Concavity of the log-likelihood with respect to readout parameters

The second order partial derivative of the instantaneous log-likelihood with respect to output parameters is computed as

$$\frac{\partial^2 \ell_i^n}{\partial W_{ij}^{out} \partial W_{ik}^{out}} = -A^2 x_j^n x_k^n \lambda_i^n,$$

resulting in a Hessian,

$$H_{\ell_i^n} = -A^2 \lambda_i^n [x^n; u^n] [x^n; u^n]^T,$$

where $[x^n; u^n]^T$ is the transpose of the column vector $[x^n; u^n]$. $H_{\ell_i^n}$ is obviously a non-positive definite matrix, since for any non-zero row vector b

$$\begin{aligned} b H_{\ell_i^n} b^T &= -(A^2 \lambda_i^n) b [x^n; u^n] [x^n; u^n]^T b^T \\ &= -(A^2 \lambda_i^n) (b [x^n; u^n])^2 \\ &\leq 0. \end{aligned}$$

Consequently, the Hessian of the total log-likelihood is also non-positive definite as $H_{\mathcal{L}} = \sum_i \sum_n H_{\ell_i^n}$. Therefore, we conclude that under fixed reservoir parameters, the log-likelihood is a concave function with respect to output parameters. This indicates that output parameters will not get stuck in the local maxima for a fixed reservoir. For adaptive reservoirs, the concavity holds for the readout-only learning phase of the training. An overview of the concavity of the point process log-likelihood for generalized linear models and its extension to different conditional intensity functions is given in Paninski (2004).

References

Abeles, M., Bergman, H., Margalit, E., & Vaadia, E. (1993). Spatiotemporal firing patterns in the frontal cortex of behaving monkeys. *Journal of Neurophysiology*, *70*, 1629–1638.

Auer, P., Burgsteiner, H., & Maass, W. (2008). A learning rule for very simple universal approximators consisting of a single layer of perceptrons. *Neural Networks*, *21*(5), 786–795.

Bengio, Y., Simard, P., & Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, *5*(2), 57–166.

Bottou, L. (2004). Stochastic learning. In O. Bousquet & U. von Luxburg (Eds.), *Advanced lectures on machine*

learning. Lecture notes in artificial intelligence, LNAI (Vol. 3176, pp. 146–168). Berlin: Springer Verlag.

Bradley, A. P. (1997). The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern Recognition*, *30*(7), 1145–1159.

Brown, E. N., Barbieri, R., Eden, U. T., & Frank, L. M. (2003). Likelihood methods for neural spike train data analysis. In J. Feng (Ed.), *Computational neuroscience: A comprehensive approach* (chapter 9). London: CRC.

Brown, E. N., Nguyen, D. P., Frank, L. M., Wilson, M. A., & Solo, V. (2001). An analysis of neural receptive field plasticity by point process adaptive filtering. *PNAS*, *98*(21), 12261–12266.

Brown, R. E., & Milner, P. M. (2003). The legacy of Donald O. Hebb: More than the Hebb synapse. *Nature Reviews Neuroscience*, *4*(12), 1013–1019.

Chao, Z. C., Bakkum, D. J., & Potter, S. M. (2008). Shaping embodied neural networks for adaptive goal-directed behavior. *PLoS Computational Biology*, *4*(3), e1000042.

Chatfield, C., & Collins, A. J. (2000). *Introduction to multivariate analysis, reprint*. Boca Raton: CRC.

Chornoboy, E. S., Schramm, L. P., & Karr, A. (1988). Maximum likelihood identification of neuronal point process systems. *Biological Cybernetics*, *59*(9), 265–275.

Cox, D. R., & Isham, V. (1980). Point processes. In *CRC monographs on statistics & applied probability*. London: Chapman & Hall/CRC.

Daley, D. J., & Vere-Jones D. (2003). *An introduction to the theory of point processes* (2nd ed., Vol. 1). New York: Springer.

Davison, A. C. (2003). *Statistical models*. Cambridge: Cambridge University Press.

Egert, U., Knott, T., Schwarz, C., Nawrot, M., Brandt, A., Rotter, S., et al. (2002). Mea-tools: An open source toolbox for the analysis of multielectrode-data with matlab. *Journal of Neuroscience Methods*, *177*, 33–42.

Ferber, J., Rutten, W. L. C., Stegenga, J., Wolters, P. S., Ramakers, G. J. A., & Pelt van, J. (2007). Conditional firing probabilities in cultured neuronal networks: A stable underlying structure in widely varying spontaneous activity patterns. *Journal of Neural Engineering*, *4*(2), 54–67.

Ganguli, S., Huh, D., & Sompolinsky, H. (2008). Memory traces in dynamical systems. *Proceedings of the National Academy of Sciences of the United States of America*, *105*(48), 18970–18975.

Goldman, M. S. (2009). Memory without feedback in a neural network. *Neuron*, *61*(4), 621–634.

Hebb, D. O. (1949). *The organization of behavior: A neuropsychological theory*. New York: Wiley.

Hodgkin, A. L., & Huxley, A. F. (1952). A quantitative description of membrane current and its application to conduction and excitation in nerve. *Journal of Physiology (London)*, *117*, 500–544.

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

Jaeger, H. (2001). *The "echo state" approach to analysing and training recurrent neural networks*. GMD report 148, GMD - German National Research Institute for Computer Science.

Jaeger, H. (2003). Adaptive nonlinear system identification with echo state networks. In S. T. S. Becker & K. Obermayer (Eds.), *Advances in neural information processing systems* (Vol. 15, pp. 593–600). Cambridge: MIT.

Jaeger, H., & Eck, D. (2006). Can't get you out of my head: A connectionist model of cyclic rehearsal. In I. Wachsmuth & G. Knoblich (Eds.), *ZiF workshop. Lecture notes in computer science* (Vol. 4930, pp. 310–335). New York: Springer.

- Jaeger, H., & Haas, H. (2004). Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication. *Science*, *304*(5667), 78–80.
- Jaeger, H., Lukosevicius, M., Popovici, D., & Siewert, U. (2007). Optimization and applications of echo state networks with leaky-integrator neurons. *Neural Networks*, *20*(3), 335–352.
- Jimbo, Y., Tateno, T., & Robinson, H. P. C. (1999). Simultaneous induction of pathway-specific potentiation and depression in networks of cortical neurons. *Biophysical Journal*, *76*(2), 670–678.
- Kandel, E. R., Schwartz, J. H., & Jessell, T. M. (2000). *Principles of neural science*. London: McGraw Hill
- Lindsey, B. G., Morris, K. F., Shannon, R., & Gerstein, G. L. (1997). Repeated patterns of distributed synchrony in neuronal assemblies. *Journal of Neurophysiology*, *78*(3), 1714–1719.
- Maass, W., Joshi, P., & Sontag, E. D. (2007). Computational aspects of feedback in neural circuits. *PLoS Computational Biology*, *3*(1), e165.
- Maass, W., Natschläger, T., & Markram, H. (2002). Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural Computation*, *14*(11), 2531–2560.
- Maass, W., Natschläger, T., & Markram, H. (2004). Fading memory and kernel properties of generic cortical microcircuit models. *Journal of Physiology—Paris*, *98*(4–6), 315–330.
- Mainen, Z. F., & Sejnowski, T. J. (1995). Reliability of spike timing in neocortical neurons. *Science*, *268*, 1503–1506.
- Mannila, H., Toivonen, H., & Verkamo, A. I. (1997). Discovery of frequent episodes in event sequences. *Data Mining and Knowledge Discovery*, *1*(3), 259–289.
- Marom, S., & Shahaf, G. (2002). Development, learning and memory in large random networks of cortical neurons: Lessons beyond anatomy. *Quarterly Reviews of Biophysics*, *35*(1), 63–87.
- Mccullagh, P., & Nelder, J. (1989). *Generalized linear models* (2nd ed.). London: Chapman & Hall/CRC.
- Murphy, B. K., & Miller, K. D. (2009). Balanced amplification: A new mechanism of selective amplification of neural activity patterns. *Neuron*, *61*(4), 635–648.
- Okatan, M., Wilson, M. A., & Brown, E. N. (2005). Analyzing functional connectivity using a network likelihood model of ensemble neural spiking activity. *Neural Computation*, *17*(9), 1927–1961.
- Paninski, L. (2004). Maximum likelihood estimation of cascade point-process neural encoding models. *Network: Computation in Neural Systems*, *15*(4), 243–262.
- Pawitan, Y. (2001). *In all likelihood: Statistical modelling and inference using likelihood*. New York: Oxford University Press.
- Prut, Y., Vaadia, E., Bergman, H., Haalman, I., Slovin, H., & Abeles, M. (1998). Spatiotemporal structure of cortical activity: Properties and behavioral relevance. *Journal of Neurophysiology*, *79*(6), 2857–2874.
- Rajaram, S., Graepel, T., & Herbrich, R. (2005). Poisson networks: A model for structured point processes. In *Proceedings of the AI STATS 2005 workshop*.
- Riehle, A., Grün, S., Diesmann, M., & Aertsen, A. (1997). Spike synchronization and rate modulation differentially involved in motor cortical function. *Science*, *278*, 1950–1953.
- Rieke, F., Warland, D., van Steveninck, R. d. R., & Bialek, W. (1999). Spikes: Exploring the neural code. In *Computational neuroscience*. Cambridge: MIT.
- Rolston, J. D., Wagenaar, D. A., & Potter, S. M. (2007). Precisely timed spatiotemporal patterns of neural activity in dissociated cortical cultures. *Neuroscience*, *148*(1), 294–303.
- Ruaro, M. E., Bonifazi, P., & Torre, V. (2005). Toward the neurocomputer: Image processing and pattern recognition with neuronal cultures. *IEEE Transactions on Biomedical Engineering*, *52*(3), 371–183.
- Shahaf, G., Eytan, D., Gal, A., Kermany, E., Lyakhov, V., Zrenner, C., et al. (2008). Order-based representation in random networks of cortical neurons. *PLoS Computational Biology*, *4*(11), e1000228.
- Shahaf, G., & Marom, S. (2001). Learning in networks of cortical neurons. *Journal of Neuroscience*, *21*(22), 8782–8788.
- Steil, J. (2004). Backpropagation-decorrelation: Online recurrent learning with $O(n)$ complexity. In *Proceedings of the IJCNN* (Vol. 1, pp. 843–848).
- Steil, J. (2007). Online reservoir adaptation by intrinsic plasticity for backpropagation-decorrelation and echo state learning. *Neural networks: The Official Journal of the International Neural Network Society*, *20*(3), 353–364.
- Villa, A. E. P., Tetko, I. V., Hyland, B., & Najem, A. (1999). Spatiotemporal activity patterns of rat cortical neurons predict responses in a conditioned task. *Proceedings of the National Academy of Sciences*, *96*(3), 1106–1111.
- Wagenaar, D. A., DeMarse, T. B., & Potter, S. M. (2005). Meabench: A toolset for multi-electrode data acquisition and on-line analysis. In *Proc. 2nd int. IEEE EMBS conference on neural engineering*.
- Williams, R. J., & Zipser, D. (1989). A learning algorithm for continually running fully recurrent neural networks. *Neural Computation*, *1*(2), 270–280.