Regularized Variational Bayesian Learning of Echo State Networks with Delay&Sum Readout

Article *in* Neural Computation · December 2011

DOI: 10.1162/NECO_a_00253 · Source: PubMed

Dmitriy Shutin
German Aerospace Center (DLR)
121 PUBLICATIONS **1,157** CITATIONS

Sanjeev R Kulkarni
Princeton University
254 PUBLICATIONS **7,311** CITATIONS

H. Vincent Poor
Princeton University
2,985 PUBLICATIONS **128,853** CITATIONS

Regularized Variational Bayesian Learning of Echo State Networks with Delay&Sum Readout

Dmitriy Shutin[1], **Christoph Zechner**[2], **Sanjeev R. Kulkarni**[1], **H. Vincent Poor**[1]

[1]Department of Electrical Engineering, E-QUAD Olden Street, Princeton University, Princeton, 08544 NJ, U.S.A.

[2]Automatic Control Laboratory, Department of Information Technology and Electrical Engineering, ETH Zürich, Physikstr. 3, 8092 Zürich, Switzerland.

Keywords: Variational Bayesian inference, reservoir computing, echo state networks.

Abstract

In this work a variational Bayesian framework for efficient training of echo state networks (ESNs) with automatic regularization and delay&sum (D&S) readout adaptation is proposed. The algorithm uses a classical batch learning of ESNs. By treating the network echo states as fixed basis functions parametrized with delay parameters, a variational Bayesian ESN training scheme is proposed. The variational approach allows for a seamless combination of sparse Bayesian learning ideas and variational Bayesian Space-Alternating Generalized Expectation-Maximization (VB-SAGE) algorithm for estimating parameters of superimposed signals. While the former method realizes automatic regularization of ESNs, which also determines which echo states and input signals are relevant for "explaining" the desired signal, the latter method provides a basis for joint estimation of D&S readout parameters. The proposed training algorithm can naturally be extended to ESNs with fixed filter neurons. It also generalizes the recently proposed expectation-maximization-based D&S readout adaptation method. The

proposed algorithm was tested on synthetic data prediction tasks as well as on dynamic handwritten character recognition.

1 Introduction

Echo state networks (ESNs) and reservoir computing in general represent a powerful class of recurrent neural networks (Jaeger et al., 2007; Verstraeten et al., 2007); they are particularly useful for nonparametric modeling of nonlinear dynamical systems. Due to a very simple training procedure ESNs have found applications in many areas of signal processing, including speech recognition and audio processing, system modeling and prediction, and filtering (Han and Wang, 2009; Verstraeten et al., 2006; Xia et al., 2008; Holzmann and Hauser, 2010), to name just a few.

A typical ESN allows learning a nonlinear dependence between an M-dimensional input signal $\boldsymbol{u}[n]$ and a P-dimensional output signal $\boldsymbol{y}[n]$ of a nonlinear dynamical system characterized by a nonlinear difference equation $\boldsymbol{y}[n] = g(\boldsymbol{y}[n-1], \ldots, \boldsymbol{y}[n-k], \ldots, \boldsymbol{u}[n], \ldots, \boldsymbol{u}[n-l], \ldots)$, where the mapping $g(\cdot)$ is typically unknown. The goal of ESN-based modeling is to approximate this mapping by (i) creating a random network of interconnected neurons, called a reservoir, and (ii) linearly combining the reservoir outputs and network input signals to form the desired network response. The operation of an ESN with L neurons can be formally described by a system of two equations:

$$\begin{aligned} \boldsymbol{x}[n+1] =& f(\boldsymbol{C}_u^T \boldsymbol{u}[n+1] + \boldsymbol{C}_x^T \boldsymbol{x}[n] + \boldsymbol{C}_y^T \boldsymbol{y}[n]) \qquad (1) \\ \boldsymbol{y}[n] =& \boldsymbol{W}[\boldsymbol{x}[n]^T, \boldsymbol{u}[n]^T]^T. \qquad (2) \end{aligned}$$

Equation (1) is the state equation of the ESN; it specifies how the responses of L neurons $\boldsymbol{x}[n] = [x_1[n], \ldots, x_L[n]]^T$ are evolving over time. In (1) the function $f : \mathbb{R}^L \mapsto \mathbb{R}^L$ is a vector-valued neuron activation function, e.g., a hyperbolic tangent, applied to each element of its argument. The matrices $\boldsymbol{C}_x \in \mathbb{R}^{L\times L}$, $\boldsymbol{C}_u \in \mathbb{R}^{M\times L}$, and $\boldsymbol{C}_y \in \mathbb{R}^{P\times L}$ are respectively the neuron interconnection weights, input signal weights, and output feedback weights. Typically, the entries of these matrices are generated and fixed during the network design stage (Jaeger, 2001; Lukoševičius and Jaeger, 2009).

Equation (2) is the output equation of the network. It states that the output of the network is formed as a linear[1] combination of network states $\boldsymbol{x}[n]$ and network inputs $\boldsymbol{u}[n]$. Under certain conditions (Jaeger, 2001) a sequence of neuron states $\boldsymbol{x}[n]$ forms echoes – a temporal basis used for reconstructing the output signal $\boldsymbol{y}[n]$. The "dynamics" of the training data are thus encoded in the echoes generated by the reservoir. The ESN training then reduces to finding an optimal estimate of $\boldsymbol{W}$ so as to minimize the squared distance between the network output and the desired network response. Notice that since $\boldsymbol{W}$ enters (2) linearly, the ESN training requires solving a system of linear equations.

Despite the simple learning procedure, a straightforward application of ESNs has two practical shortcomings: (i) an estimation of $\boldsymbol{W}$, especially for large ESNs with many neurons, requires regularization (Jaeger, 2001) and (ii) simple ESNs have been shown to fail for certain learning problems, e.g., they cannot learn multiple attractors at the same time (Jaeger, 2007). Regularization has been extensively studied in the literature within the context of ill-posed problems; the Moore-Penrose inverse (Golub and Van Loan, 1996) and ridge regression (Bishop, 2006), also known as Tikhonov regularization, are standard approaches to finding a regularized solution to the linear least-squares (LS) estimation problem. The "universality" of ESNs can be significantly boosted by introducing filter neurons and delay&sum (D&S) readouts in the ESN structure (Holzmann and Hauser, 2010; Wustlich and Siewert, 2007; Zechner and Shutin, 2010). Equipping neurons with additional filters will result in neurons that are "specialized" to more relevant frequency bands. This is achieved by applying a linear time-invariant filter to the output of the neuron activation function in (1). Introducing delays makes it possible to shift the reservoir signals in time and provides a computationally inexpensive method to vastly improve the memory capacity of the network. The parameters of such filter neurons and the corresponding readout delays can be chosen randomly during the network initialization or heuristically through trial and error (Wustlich and Siewert, 2007; Holzmann and Hauser, 2010).

The regularization of ESN LS-based training on the one hand, and optimization of D&S readout parameters and filter neurons on the other hand are typically two unconnected optimization steps. Motivated by the lack of a formal optimization frame-

[1] In general one can also reconstruct the desired output $\boldsymbol{y}[n]$ as $\boldsymbol{y}[n] = s(\tilde{\boldsymbol{y}}[n])$, where $s : \mathbb{R}^P \mapsto \mathbb{R}^P$ is a bijective mapping and $\tilde{\boldsymbol{y}}[n] = \boldsymbol{W}[\boldsymbol{x}[n]^T, \boldsymbol{u}[n]^T]^T$ (Lukoševičius and Jaeger, 2009).

work that combines both regularization and ESN parameter adaptation, and inspired by the recent developments of the variational Bayesian methods (Bishop, 2006; Beal, 2003) for sparse Bayesian learning (SBL) (Shutin et al., 2011b; Seeger and Wipf, 2010; Tzikas et al., 2008; Tipping, 2001; Bishop and Tipping, 2000) and variational nonlinear parameter estimation (Shutin and Fleury, 2011), we propose a variational Bayesian ESN training framework. In the new framework the ESN training is formulated as a variational Bayesian inference problem on a directed acyclic graph (DAG) (Bishop, 2006). Specifically, the unknown network parameters are jointly estimated by minimizing the Kullback-Leibler divergence between the true posterior probability density function (pdf) of the network parameters and a variational approximation to this posterior. The estimation of the output coefficients $\boldsymbol{W}$ and regularization parameters is realized using ideas inspired by variational SBL approach (Bishop and Tipping, 2000). This not only allows for an automatic regularization of the network, but also provides quantitative information about the relative relevance or importance of individual neurons and network input signals. The estimation of D&S readout parameters is implemented using the variational Bayesian Space-Alternating Generalized Expectation-Maximization (VB-SAGE) algorithm which was originally proposed for the variational estimation of superimposed signal parameters (Shutin and Fleury, 2011). The VB-SAGE framework allows for a monotonic decrease of the Kullback-Leibler divergence between the two pdfs with respect to only a subset of the parameters of interest using latent variables, also called admissible hidden data — an analog of the complete data in the expectation-maximization (EM) framework (Bishop, 2006). We demonstrate that latent variables reduce the complexity of the objective function for estimating delay parameters of a single neuron, which leads to a more efficient numerical optimization.

Previously we have considered the application of the VB-SAGE algorithm within the ESN training framework (Zechner and Shutin, 2010). However, in (Zechner and Shutin, 2010) the automatic regularization has not been a part of the estimation scheme. Also, the variational approximation used in (Zechner and Shutin, 2010) assumes a statistical independence between the elements of $\boldsymbol{w}$. Although this assumption significantly simplifies the variational inference of the ESN weight coefficients, it leads to a poorer performance of the trained models and does not generalize the classical pseudoinverse-based ESN training. Here we do not impose any independence assumptions on the elements of $\boldsymbol{w}$. We demonstrate that the proposed variational ESN training framework generalizes the existing techniques for ESN training. In particular, the Tikhonov-like regularization of ESNs (Jaeger, 2001) and expectation-maximization (EM)-based estimation of D&S readout parameters (Holzmann and Hauser, 2010) are obtained as special cases of the proposed variational Bayesian ESN training. Moreover, the proposed algorithm automatically regularizes the obtained solution by taking into account the training data and the amount of additive noise.

The rest of the paper is organized as follows. In Section 2 we discuss the extended ESN model and explain the variables involved; in Section 3 we formulate the probabilistic model and discuss the variational inference of model parameters; in Section 4 we discuss the implementation and initialization of the learning algorithm. Finally, in Section 5 we consider several learning examples to demonstrate the performance of the proposed scheme.

Throughout the paper we shall make use of the following notation. Vectors are represented as boldface lowercase letters, e.g., $\boldsymbol{x}$, and matrices as boldface uppercase letters, e.g., $\boldsymbol{X}$. For vectors and matrices $(\cdot)^T$ denotes the transpose. Notation $\boldsymbol{A} = [\boldsymbol{X}, \boldsymbol{Y}]$ is used to denote a matrix $\boldsymbol{A}$ obtained by concatenating matrices $\boldsymbol{X}$ and $\boldsymbol{Y}$; it is assumed that $\boldsymbol{X}$ and $\boldsymbol{Y}$ have the same number of rows. Sets are represented as calligraphic uppercase letters, e.g., $\mathcal{S}$. We use $\mathcal{I} = \{1, \ldots, L\}$ to denote an index set of L neurons. With a slight abuse of notation we write $\boldsymbol{x}_{\mathcal{I}}$ to denote a set of random variables $\{\boldsymbol{x}_k : \text{s.t.}\, k \in \mathcal{I}\}$; also, for $l \in \mathcal{I}$, $\boldsymbol{x}_{\bar{l}}$ denotes a set $\{\boldsymbol{x}_k : \text{s.t.}\, k \in \mathcal{I} \backslash \{l\}\}$. Two types of proportionality are used: $x \propto y$ denotes $x = \alpha y$, and $x \propto^e y$ denotes $\mathrm{e}^x = \mathrm{e}^\beta \mathrm{e}^y$ and thus $x = \beta + y$, for arbitrary constants α, and β. We use $\mathbb{E}_{q(\boldsymbol{x})}\{f(\boldsymbol{x})\}$ to denote the expectation of a function $f(\boldsymbol{x})$ with respect to a probability density $q(\boldsymbol{x})$. Finally, $\mathrm{N}(\boldsymbol{x}|\boldsymbol{a}, \boldsymbol{B})$ denotes a multivariate Gaussian probability density function (pdf) with a mean $\boldsymbol{a}$ and a covariance matrix $\boldsymbol{B}$; $\mathrm{Ga}(x|a,b) = b^a x^{a-1} \exp(-bx)/\Gamma(a)$ denotes a gamma pdf with parameters a and b.

2 Extended ESN model

Consider a standard batch ESN learning problem with N training samples $\{y[n], \boldsymbol{u}[n]\}_{n=n_0}^{n_0+N-1}$ and N echo state samples $\{\boldsymbol{x}[n]\}_{n=n_0}^{n_0+N-1}$ generated with an untrained network. The time index $n_0 \geq 0$ is chosen such as to make sure that the ESN transients due to the initialization of the network fade out. For simplicity we restrict ourselves to a scalar output signal $y[n]$. Considering a general P-dimensional output signal merely leads to a more complicated probabilistic signal model without adding any new aspects relevant to the understanding of the new proposed concepts and methods. [2]

Let $\boldsymbol{x}_l(\tau_l) = \big[x_l[n_0 - \tau_l], \ldots, x_l[n_0 + N - 1 - \tau_l]\big]^T$ denote a vector of echo state samples $x_l[n]$ of the lth neuron delayed by τ_l. We will collect these vectors in an $N \times L$ matrix $\boldsymbol{X}(\boldsymbol{\tau}) = [\boldsymbol{x}_1(\tau_1), \ldots, \boldsymbol{x}_L(\tau_L)]$, where $\boldsymbol{\tau} = [\tau_1, \ldots, \tau_L]^T$. In order to ensure the causality of the ESN with D&S readouts we will assume that $x_l[n_0 + i - \tau_l] = 0$ when $n_0 + i - \tau_l < 0$, for any $\tau_l \geq 0$ and $i = 0, \ldots, N-1$. Similarly, we collect N samples of the mth input signal $u_m[n]$ in a vector $\boldsymbol{u}_m = [u_m[n_0], \ldots, u_m[n_0 + N - 1]]^T$ and define an $N \times M$ matrix $\boldsymbol{U} = [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_M]$. Now, the output equation of an ESN with D&S readout can be rewriten in the following form:

$$\boldsymbol{y} = \boldsymbol{\Phi}(\boldsymbol{\tau})\boldsymbol{w} + \boldsymbol{\xi}, \tag{3}$$

where $\boldsymbol{y} = [y[n_0], \ldots, y[n_0 + N - 1]]^T$ is a desired output of the network that is represented as a linear combination of column-vectors in $\boldsymbol{\Phi}(\boldsymbol{\tau}) = [\boldsymbol{X}(\boldsymbol{\tau}), \boldsymbol{U}]$ perturbed by a random vector $\boldsymbol{\xi} = [\xi[n_0], \ldots, \xi[n_0 + N - 1]]^T$. This perturbation models a random error between the predicted network response $\boldsymbol{\Phi}(\boldsymbol{\tau})\boldsymbol{w}$ and the desired response $\boldsymbol{y}$. We will assume that each element of $\boldsymbol{\xi}$ is drawn independently from a zero mean Gaussian distribution with variance σ^2. Let us also point out that for the scalar output $y[n]$ the output weight matrix $\boldsymbol{W}$ in (2) reduces to a vector $\boldsymbol{w}$.

Observe that delays $\boldsymbol{\tau}$ do not influence the generation of echo states $x_l[n]$; they simply "shift" the signals $x_l[n]$ before they are linearly combined to form the network output, hence leading to the D&S readout terminology. When filter neurons are employed in the reservoir, the generation of echo states becomes depend on the parameters of the neuron filters. In this case the output of the lth neuron activation function is computed as $\tilde{x}_l[n+1] = f(\boldsymbol{c}_{ul}^T \boldsymbol{u}[n+1] + \boldsymbol{c}_{xl}^T \boldsymbol{x}[n] + \boldsymbol{c}_{yl}^T \boldsymbol{y}[n])$, where $\boldsymbol{c}_{ul}$, $\boldsymbol{c}_{xl}$, and $\boldsymbol{c}_{yl}$ are the

[2] Note that in general each element signal in a P-dimensional network output signal might have a different variance. This case can be accounted for by an appropriate, albeit more elaborate, noise model in a relatively straightforward fashion. Specifically, it will lead to the introduction of non-circular noise covariance matrices. This case is left outside the scope of the paper.

lth column vectors from the matrices $\boldsymbol{C}_u$, $\boldsymbol{C}_x$, and $\boldsymbol{C}_y$ respectively. The filtered echo state, i.e., the filter neuron output signal, is then computed as

$$x_l[n+1] = \sum_{k=0}^{\infty} h_l[k]\tilde{x}_l[n+1-k],$$

where $h_l[n]$ is an impulse response of a stable linear time-invariant filter, e.g., a band-pass filter.[3] Typically it is assumed that transfer functions of all neuron filters $h_l[n]$, $l = 1, \ldots, L$, are fixed at the network design stage (Holzmann and Hauser, 2010; Wustlich and Siewert, 2007). This is essentially a simplifying assumption; adapting filter parameters is complicated due to a recurrent inter-dependency of the neurons in the network. Although it is possible to construct an algorithm to estimate neuron filters $h_l[n]$ (Zechner and Shutin, 2010), there are no theoretical convergence or monotonicity guarantees for this learning scheme. Henceforth we assume that the parameters of neuron filters are fixed at the design stage and the adaptation of the filter neuron parameters is left outside the scope of this paper. For all our experiments in Section 5 we assume ESNs without filter neurons, which are obtained by choosing $h_l[n] = \delta[n]$, where $\delta[n]$ is a discrete-time unit impulse.[4]

Let us point out that in a batch learning regime the columns of the matrix $\boldsymbol{\Phi}(\boldsymbol{\tau})$ corresponding to the generated echo states can be interpreted as parametric basis functions, parametrized by parameters $\boldsymbol{\tau}$. In what follows we explain how this can be exploited to formulate a variational Bayesian framework to jointly estimate the D&S readout parameters and train the network.

3 Bayesian ESN learning

We first note that an ESN training is equivalent to the maximization of the log-likelihood function

$$\log p(\boldsymbol{y}|\boldsymbol{\tau}, \boldsymbol{w}) \propto^e -\frac{1}{2\sigma^2} \|\boldsymbol{y} - \boldsymbol{\Phi}(\boldsymbol{\tau})\boldsymbol{w}\|^2 . \tag{4}$$

[3] Note that practically the filter $h_l[n]$ can represent a linear time-invariant system with an infinite impulse response, as well as with a finite impulse response.

[4] The ESN training algorithm proposed in this work can be easily extended to ESN with arbitrary, although fixed, filter neurons. This extension merely leads to a more complicated signal model without adding any new aspect relevant to the understanding of the new proposed concepts and methods.

Notice that even when parameters $\boldsymbol{\tau}$ are assumed to be fixed, the estimation of $\boldsymbol{w}$ from (4) typically requires a regularization (Lukoševičius and Jaeger, 2009; Jaeger, 2001). Bayesian methods introduce regularization by imposing constraints on the model parameters using priors. Consider a prior pdf $p(\boldsymbol{\tau}, \boldsymbol{w}|\boldsymbol{\alpha})$, where $\boldsymbol{\alpha}$ is a vector of prior parameters. This prior leads to a posterior pdf that in the log-domain can be expressed as

$$\log p(\boldsymbol{\tau}, \boldsymbol{w}|\boldsymbol{y}, \boldsymbol{\alpha}) \propto^{e} -\frac{1}{2\sigma^2} \|\boldsymbol{y} - \boldsymbol{\Phi}(\boldsymbol{\tau})\boldsymbol{w}\|^2 + \log p(\boldsymbol{\tau}, \boldsymbol{w}|\boldsymbol{\alpha}), \tag{5}$$

where $\log p(\boldsymbol{\tau}, \boldsymbol{w}|\boldsymbol{\alpha})$ performs the role of a regularizing function. Depending on the choice of $p(\boldsymbol{\tau}, \boldsymbol{w}|\boldsymbol{\alpha})$ different forms of the regularizing function can be constructed. Henceforth we will assume that the prior $p(\boldsymbol{\tau}, \boldsymbol{w}|\boldsymbol{\alpha})$ factors as

$$p(\boldsymbol{\tau}, \boldsymbol{w}|\boldsymbol{\alpha}) = p(\boldsymbol{\tau})p(\boldsymbol{w}|\boldsymbol{\alpha}). \tag{6}$$

The motivation behind this assumption is the following: through the prior $p(\boldsymbol{w}|\boldsymbol{\alpha})$ we can control the contribution of individual basis functions in $\boldsymbol{\Phi}(\boldsymbol{\tau})$ irrespective of their form, which is specified by the parameters $\boldsymbol{\tau}$. The prior $p(\boldsymbol{w}|\boldsymbol{\alpha})$ is assumed to fully factor as $p(\boldsymbol{w}|\boldsymbol{\alpha}) = \prod_{k=1}^{K} p(w_k|\alpha_k)$, where $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_K]^T$, $K = L + M$, and $p(w_k|\alpha_k)$ is selected as a zero mean symmetric pdf with the prior parameter α_l inversely proportional to the width of $p(w_k|\alpha_k)$. Such factorization of the prior enables a more flexible control over the importance of each column in $\boldsymbol{\Phi}(\boldsymbol{\tau})$ through the coefficients $\boldsymbol{\alpha}$: a large value of α_k drives the posterior mean of the corresponding weight w_k towards zero, thus effectively suppressing the corresponding basis function in $\boldsymbol{\Phi}(\boldsymbol{\tau})$ and leading to a regularized solution. Such a formulation of the prior is related to sparse Bayesian learning (SBL) (Shutin et al., 2011b; Tzikas et al., 2008; Tipping, 2001; Bishop and Tipping, 2000). In our work we will select $p(w_k|\alpha_k)$ as a Gaussian pdf with zero mean and variance α_k^{-1}. This choice corresponds to a penalty function $\sum_k \alpha_k |w_k|^2$ in (5), which is a weighted ℓ_2 norm of the weight vector $\boldsymbol{w}$ (Bishop, 2006).[5] Such form of the penalty leads to a Tikhonov-like regularization[6] of the original estimation problem

[5] It is also possible to extend the inference procedure discussed in the paper to Laplacian priors $p(w_k|\alpha_k)$. This selection leads to an ℓ_1-type of log-likelihood penalty $\sum_k \alpha_k |w_k|$ and Least-Absolute Shrinkage and Selection Operator (LASSO) regression. We leave this development outside the scope of this work.

[6] Strictly speaking, this is so when $p(\boldsymbol{\tau}) \propto 1$.

(4) with parameters $\boldsymbol{\alpha}$ acting as regularization parameters. Additionally, the Gaussian prior $p(\boldsymbol{w}|\boldsymbol{\alpha})$ and the Gaussian likelihood of $\boldsymbol{w}$ in (4) form a conjugate family (Bishop, 2006), which in turn allows for a computation of the posterior distribution of $\boldsymbol{w}$ in closed form.

Within the SBL framework the parameters $\boldsymbol{\alpha}$ are then determined from

$$p(\boldsymbol{y}|\boldsymbol{\alpha}) = \int p(\boldsymbol{y}|\boldsymbol{\tau}, \boldsymbol{w})p(\boldsymbol{\tau}, \boldsymbol{w}|\boldsymbol{\alpha})\mathrm{d}\boldsymbol{\tau}\mathrm{d}\boldsymbol{w}, \tag{7}$$

which is also known as the marginal likelihood function, or evidence (Tipping, 2001; Tipping and Faul, 2003). Unfortunately, the nonlinear dependence of the integrand in (7) on the parameters $\boldsymbol{\tau}$ precludes the exact evaluation of the marginal $p(\boldsymbol{y}|\boldsymbol{\alpha})$. Additionally, this nonlinear dependency significantly complicates the optimization of the posterior (5). This motivates the use of approximation techniques to estimate the parameters $\boldsymbol{w}$, $\boldsymbol{\tau}$ and $\boldsymbol{\alpha}$ of the extended ESN model.

3.1 Variational Bayesian inference

Note that the joint estimation of ESN parameters $\boldsymbol{w}$, $\boldsymbol{\tau}$ and regularization parameters $\boldsymbol{\alpha}$ is equivalent to the maximization of the posterior pdf

$$p(\boldsymbol{\tau}, \boldsymbol{w}, \boldsymbol{\alpha}|\boldsymbol{y}) \propto p(\boldsymbol{\tau}, \boldsymbol{w}|\boldsymbol{y}, \boldsymbol{\alpha})p(\boldsymbol{y}|\boldsymbol{\alpha})p(\boldsymbol{\alpha}), \tag{8}$$

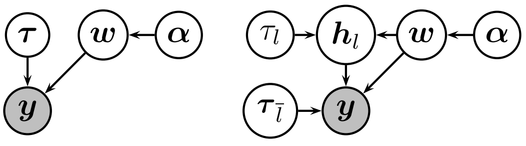
which involves (5), (7), and the prior $p(\boldsymbol{\alpha})$. Instead of computing (8) directly we approximate it with a proxy pdf $q(\boldsymbol{\tau}, \boldsymbol{w}, \boldsymbol{\alpha})$ using variational Bayesian inference methods (Beal, 2003; Bishop, 2006).

Variational inference is realized by maximizing the lower bound on the marginal log-likelihood $\log p(\boldsymbol{y})$

$$\log p(\boldsymbol{y}) \geq \int q(\boldsymbol{\tau}, \boldsymbol{w}, \boldsymbol{\alpha}) \log \frac{p(\boldsymbol{\tau}, \boldsymbol{w}, \boldsymbol{\alpha}, \boldsymbol{y})}{q(\boldsymbol{\tau}, \boldsymbol{w}, \boldsymbol{\alpha})}\mathrm{d}\boldsymbol{\tau}\mathrm{d}\boldsymbol{w}\mathrm{d}\boldsymbol{\alpha} \tag{9}$$

with respect to $q(\boldsymbol{\tau}, \boldsymbol{w}, \boldsymbol{\alpha})$. It is known (Beal, 2003; Bishop, 2006) that the density $q(\boldsymbol{\tau}, \boldsymbol{w}, \boldsymbol{\alpha})$ that maximizes the lower bound in (9) also minimizes the Kullback-Leibler divergence between $q(\boldsymbol{\tau}, \boldsymbol{w}, \boldsymbol{\alpha})$ and often “intractable” true posterioir pdf $p(\boldsymbol{\tau}, \boldsymbol{w}, \boldsymbol{\alpha}|\boldsymbol{y})$.

Observe that optimizing the lower bound in (9) requires specifying both the approximating pdf and the joint pdf. Using (4) and (6) it is easy to conclude that the joint pdf



(b)

Figure 1: a) A graphical model for estimating $\boldsymbol{\tau}$, $\boldsymbol{w}$, and $\boldsymbol{\alpha}$; b) a graphical model with the admissible hidden data $\boldsymbol{h}_l$ for estimating the delay parameter τ_l of the lth neuron.

$p(\boldsymbol{\tau}, \boldsymbol{w}, \boldsymbol{\alpha}, \boldsymbol{y})$ can be represented as follows:

$$p(\boldsymbol{\tau}, \boldsymbol{w}, \boldsymbol{\alpha}, \boldsymbol{y}) = p(\boldsymbol{y}|\boldsymbol{\tau}, \boldsymbol{w})p(\boldsymbol{w}|\boldsymbol{\alpha})p(\boldsymbol{\tau})p(\boldsymbol{\alpha}). \tag{10}$$

A DAG in Fig. 1a captures this factorization using a graphical model. Based on the Bayesian ESN model discussed in Section (3) it is easy to conclude that $p(\boldsymbol{y}|\boldsymbol{\tau}, \boldsymbol{w}) = \mathrm{N}(\boldsymbol{y}|\boldsymbol{\Phi}(\boldsymbol{\tau})\boldsymbol{w}, \sigma^2\boldsymbol{I})$ and $p(\boldsymbol{w}|\boldsymbol{\alpha}) = \mathrm{N}(\boldsymbol{w}|\boldsymbol{0}, \boldsymbol{A}^{-1})$, where $\boldsymbol{A} = \mathrm{diag}\{\boldsymbol{\alpha}\}$. The choice of priors $p(\boldsymbol{\tau})$ and $p(\boldsymbol{\alpha})$ is arbitrary in general. We will, however, assume that both priors factor as $p(\boldsymbol{\tau}) = \prod_{l=1}^{L} p(\tau_l)$ and $p(\boldsymbol{\alpha}) = \prod_{k=1}^{K} p(\alpha_k)$. The choice of $p(\tau_l)$ is arbitrary in the context of our work. As we will show later, any desired form of $p(\tau_l)$ can be used in the algorithm. The prior $p(\alpha_k)$, also called a hyperprior, is selected as a gamma pdf, i.e., $p(\alpha_k) = \mathrm{Ga}(\alpha_k|a_k, b_k)$, with the prior parameters a_k and b_k chosen so as to ensure the desired form of the prior. Practically we will select $a_k = b_k = 0$ to render this prior non-informative (Tipping, 2001). Such formulation of the hyperprior is related to automatic relevance determination (ARD) (Neal, 1996; MacKay, 1994). Let us stress that the ARD formulation of the hyperprior distribution also leads to a number of very efficient inference algorithms (Shutin et al., 2011b; Tipping and Faul, 2003).

The approximating pdf $q(\boldsymbol{\tau}, \boldsymbol{w}, \boldsymbol{\alpha})$ is typically a free parameter. However, to make the optimization of the bound in (9) tractable one typically assumes a suitable factorization of $q(\boldsymbol{\tau}, \boldsymbol{w}, \boldsymbol{\alpha})$ and constrains individual approximating factors to some classes of parametric pdfs. Henceforth we will assume that

$$q(\boldsymbol{\tau}, \boldsymbol{w}, \boldsymbol{\alpha}) = q(\boldsymbol{w}) \prod_{k=1}^{K} q(\alpha_k) \prod_{j=1}^{L} q(\tau_j). \tag{11}$$

The motivation behind such factorization is the following. Selection $q(\boldsymbol{\alpha}) = \prod_{k=1}^{K} q(\alpha_k)$ follows from the assumption that $p(\boldsymbol{w}, \boldsymbol{\alpha}) = \prod_{k=1}^{K} p(w_k|\alpha_k)p(\alpha_k)$. The assumption

$q(\boldsymbol{\tau}) = \prod_{j=1}^{L} q(\tau_j)$ is mainly done for computational reasons. Essentially, such factorization allows one to reduce a nonlinear L dimensional optimization with respect to $q(\boldsymbol{\tau})$ to a series of L simpler one-dimensional nonlinear optimizations with respect to $q(\tau_l)$, which makes the numerical estimation problem much simpler.

Consider a random variable $a \in \{\tau_1, \ldots, \tau_L, \boldsymbol{w}, \alpha_1, \ldots, \alpha_K\}$, and assume we are interested in finding $q(a)$ that maximizes the lower bound (9). Define now

$$\tilde{p}(a) \propto \exp\left(\mathbb{E}_{q(\mathcal{MB}(a))}\left\{ \log p(a|\mathcal{MB}(a))\right\}\right), \tag{12}$$

where $\mathcal{MB}(a)$ is a Markov blanket[7] of the variable a. It is then easy to show that an unconstrained (form-free) variational solution for $q(a)$, $a \in \{\tau_1, \ldots, \tau_L, \boldsymbol{w}, \alpha_1, \ldots, \alpha_K\}$, that maximizes the bound (9) is found as $q(a) = \tilde{p}(a)$. If $q(a)$ is constrained to some suitable class of density functions $\mathcal{Q}(a)$, then a constrained solution is obtained by solving

$$q(a) = \underset{q^*(a) \in \mathcal{Q}(a)}{\arg\min}\ D_{\mathrm{KL}}(q^*(a) \| \tilde{p}(a)). \tag{13}$$

Note that an unconstrained solution naturally gives a tighter bound on $\log p(\boldsymbol{y})$ in (9).

Now let us return to the approximating pdf $q(\boldsymbol{\tau}, \boldsymbol{w}, \boldsymbol{\alpha})$. In the case of $q(\boldsymbol{w})$ it is easy to verify that $\log \tilde{p}(\boldsymbol{w})$ computed from (12) is quadratic in $\boldsymbol{w}$, which implies that $\tilde{p}(\boldsymbol{w})$ must be a Gaussian pdf. This can be easily verified by noting that the posterior $p(\boldsymbol{w}|\mathcal{MB}(\boldsymbol{w})) = p(\boldsymbol{w}|\boldsymbol{y}, \boldsymbol{\alpha}, \boldsymbol{\tau}))$ is proportional to a product of two Gaussian pdfs: $p(\boldsymbol{w}|\boldsymbol{y}, \boldsymbol{\alpha}, \boldsymbol{\tau})) \propto p(\boldsymbol{y}|\boldsymbol{w}, \boldsymbol{\tau})p(\boldsymbol{w}|\boldsymbol{\alpha})$. Therefore, selecting $q(\boldsymbol{w}) = N(\boldsymbol{w}|\widehat{\boldsymbol{w}}, \widehat{\boldsymbol{S}^w})$ is equivalent to a form-free variational solution for this factor. Following the same line of argument it can be shown that $\tilde{p}(\alpha_k)$, $k = 1, \ldots, K$, is a gamma pdf. Therefore, selecting $q(\alpha_k) = \mathrm{Ga}(\alpha_k|\widehat{a}_k, \widehat{b}_k)$ corresponds to a form-free variational solution for $q(\alpha_k)$. As a single exception to the above cases we restrict $q(\tau_l)$ to a set of Dirac measures $\mathcal{Q}(\tau_l) = \{\delta(\tau_l - \widehat{\tau}_l)|\widehat{\tau}_l \in \{0, \ldots, N-1\}\}$, $l = 1, \ldots, L$. By doing so we restrict ourselves to the integer point estimate of the lth neuron delay τ_l. While other forms of the pdfs can be assumed here, their study is left outside the scope of this paper.

Now, the variational inference reduces to the estimation of the variational parameters $\widehat{\boldsymbol{w}}$, $\widehat{\boldsymbol{S}^w}$, $\widehat{a}_k$, $\widehat{b}_k$, $k = 1, \ldots, K$, using (12) and $\widehat{\tau}_l$, $l = 1, \ldots, L$, using (13).

[7] The Markov blanket of a variable node in a DAG is a set of nodes that includes parent nodes, children nodes, and co-parents of the children nodes (Bishop, 2006).

Should our estimation problem be independent of $\boldsymbol{\tau}$, the solution to the variational inference of $q(\boldsymbol{w})$ and $q(\boldsymbol{\alpha})$ can be easily computed (Bishop and Tipping, 2000; Shutin et al., 2011a; Tipping and Faul, 2003). Unfortunately, the nonlinear dependence of $\boldsymbol{X}(\boldsymbol{\tau})$ on $\boldsymbol{\tau}$ significantly complicates the evaluation of (12) and (13). In fact, when $\boldsymbol{y}$ is observed the variables $\boldsymbol{w}$ and $\boldsymbol{\tau}$ become conditionally dependent (Bishop, 2006); the variational estimation of a single factor $q(\tau_l)$ would thus require computing the expectation with respect to $\mathcal{MB}(\tau_l) = \{\boldsymbol{w}, \tau_1, \ldots, \tau_{l-1}, \tau_{l+1}, \ldots, \tau_L\}$ in (12). As a consequence, the straightforward variational estimation of $q(\tau_l)$ might become computationally quite costly due to the correlations between the elements of $\boldsymbol{w}$, especially when the number of columns in $\boldsymbol{\Phi}(\boldsymbol{\tau})$ is high. Although this approach is practically realizable, these numerical difficulties can be efficiently circumvented by appealing to the EM-type of inference schemes used for estimating parameters of superimposed signals (Feder and Weinstein, 1988; Fleury et al., 1999; Shutin and Fleury, 2011). Here we propose to use one such algorithm known as the VB-SAGE algorithm (Shutin and Fleury, 2011).

The VB-SAGE algorithm — a variational extension of the original SAGE algorithm (Fessler and Hero, 1994) — allows one to simplify the optimization of the bound in (9) by introducing latent variables termed admissible hidden data. Within the VB-SAGE algorithm the admissible hidden data is introduced for only a subset of parameters of interest; this distinguishes the VB-SAGE framework from a closely related EM framework and its variational extensions (Sung et al., 2008a,b; Palmer et al., 2006; Beal, 2003; Attias, 1999), where complete data is introduced for all the unknown parameters. In our case we would like to simplify the inference of a single delay parameter τ_l. The VB-SAGE algorithm is then used to maximize the bound in (9) with respect to $q(\tau_l)$ by performing a variational inference on a new graph that has been appropriately extended with latent variables. The monotonicity property of the VB-SAGE algorithm guarantees that this optimization strategy necessarily improves the variational bound in (9) (Shutin and Fleury, 2011).

3.2 Variational Bayesian Space-Alternating inference

Let us begin by formally defining the notion of admissible hidden data. Let $\mathcal{P} = \{\mathcal{P}_s, \mathcal{P}_{\bar{s}}\}$ be a set of all the unknown parameters[8], and let $\mathcal{P}_s$ be a subset of parameters we wish to update.

Definition 1. *Given a measurement $\boldsymbol{y}$, $\boldsymbol{h}_s$ is said to be admissible hidden data with respect to $\mathcal{P}_s$ if the factorization*

$$p(\boldsymbol{h}_s, \boldsymbol{y}, \mathcal{P}) = p(\boldsymbol{y}|\boldsymbol{h}_s, \mathcal{P}_{\bar{s}})p(\boldsymbol{h}_s, \mathcal{P}) \tag{14}$$

is satisfied (Shutin and Fleury, 2011; Fessler and Hero, 1994).

The purpose of the hidden data is to make the update procedure for the subset $\mathcal{P}_s$ a tractable optimization problem. Now, let us re-inspect (3). We observe that the network output is represented as a superposition of L neuron responses $\boldsymbol{X}(\boldsymbol{\tau}) = [\boldsymbol{x}_1(\tau_1), \ldots, \boldsymbol{x}_L(\tau_L)]$ and M input signals $\boldsymbol{U} = [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_M]$. Obviously, the weight vector $\boldsymbol{w}$ can be partitioned as $\boldsymbol{w} = [\boldsymbol{w}_x^T, \boldsymbol{w}_u^T]^T$, where $\boldsymbol{w}_x$ and $\boldsymbol{w}_u$ are respectively the weighting coefficients for the echo states $\boldsymbol{X}(\boldsymbol{\tau})$ and the input signals $\boldsymbol{U}$. In what follows we formulate the learning problem so as to estimate the delay τ_l of a single neuron.

Lemma 1. *Let w_{xl} denotes the lth element from the vector $\boldsymbol{w}_x$. Decompose the total perturbation $\boldsymbol{\xi}$ in* (3) *into two statistically independent parts such that $\boldsymbol{\xi} = \boldsymbol{\xi}_l + \boldsymbol{\eta}_l$, where $\mathbb{E}\{\boldsymbol{\xi}_l\boldsymbol{\xi}_l^T\} = \beta_l\sigma^2\boldsymbol{I}$ and $\mathbb{E}\{\boldsymbol{\eta}_l\boldsymbol{\eta}_l^T\} = (1-\beta_l)\sigma^2\boldsymbol{I}$ for some $0 \leq \beta_l \leq 1$.*

Then, a variable

$$\boldsymbol{h}_l = \boldsymbol{x}_l(\tau_l)w_{xl} + \boldsymbol{\xi}_l, \tag{15}$$

is admissible hidden data with respect to τ_l.

Proof. With the new variable $\boldsymbol{h}_l$ the ESN output expression (3) can be rewritten as

$$\boldsymbol{y} = \boldsymbol{h}_l + \boldsymbol{X}_{\bar{l}}(\boldsymbol{\tau}_{\bar{l}})\boldsymbol{w}_{x\bar{l}} + \boldsymbol{U}\boldsymbol{w}_u + \boldsymbol{\eta}_l, \tag{16}$$

where $\boldsymbol{X}_{\bar{l}}(\boldsymbol{\tau}_{\bar{l}}) = [\boldsymbol{x}_1(\tau_1), \ldots, \boldsymbol{x}_{l-1}(\tau_{l-1}), \boldsymbol{x}_{l+1}(\tau_{l+1}), \ldots, \boldsymbol{x}_L(\tau_L)]$ is an $N \times L - 1$ matrix of delayed echo states with the response of the lth neuron removed and $\boldsymbol{w}_{x\bar{l}}$ is vector

[8] We will assume that $\mathcal{P}_s \bigcap \mathcal{P}_{\bar{s}} = \emptyset$.

with $L-1$ elements obtained by removing the lth weight w_{xl} from $\boldsymbol{w}_x$. Then, the modified graphical model that accounts for the introduced variable $\boldsymbol{h}_l$ can be represented as shown in Fig. 1b, from which it immediately follows that the new joint pdf factors as

$$\begin{aligned} p(\boldsymbol{y}, \boldsymbol{h}_l, \boldsymbol{\tau}, \boldsymbol{w}, \boldsymbol{\alpha}) =& p(\boldsymbol{y}|\boldsymbol{h}_l, \boldsymbol{\tau}_{\bar{l}}, \boldsymbol{w})\times \\ & p(\boldsymbol{h}_l|\boldsymbol{w}, \tau_l)p(\boldsymbol{\tau})p(\boldsymbol{w}|\boldsymbol{\alpha})p(\boldsymbol{\alpha}). \end{aligned} \tag{17}$$

By comparing (17) and (14) we conclude that $\boldsymbol{h}_l$ defined in (15) is admissible hidden data with respect to $\mathcal{P}_s \equiv \{\tau_l\}$. □

The key quantities in (17) that distinguish it from (10) is the likelihood on the admissible hidden data $p(\boldsymbol{y}|\boldsymbol{h}_l, \boldsymbol{\tau}_{\bar{l}}, \boldsymbol{w}) = \mathrm{N}\Big(\boldsymbol{y}|\left(\boldsymbol{h}_l + \boldsymbol{X}_{\bar{l}}(\boldsymbol{\tau}_{\bar{l}})\boldsymbol{w}_{x\bar{l}} + \boldsymbol{U}\boldsymbol{w}_u\right), (1-\beta_l)\sigma^2\boldsymbol{I}\Big)$ and the new likelihood of τ_l, which is now a function of $\boldsymbol{w}$ and $\boldsymbol{h}_l$. From (15) it follows that $p(\boldsymbol{h}_l|\boldsymbol{w}, \tau_l) = p(\boldsymbol{h}_l|w_{xl}, \tau_l)$, where $p(\boldsymbol{h}_l|w_{xl}, \tau_l) = \mathrm{N}(\boldsymbol{h}_l|\boldsymbol{x}_l(\tau_l)w_{xl}, \beta_l\sigma_l^2\boldsymbol{I})$. The VB-SAGE-based inference of $q(\tau_l)$ now incorporates two steps: (i) a variational inference of the admissible hidden data $\boldsymbol{h}_l$ using the augmented graph in Fig. 1b, which forms the VB-SAGE-E-step of the scheme, followed by the (ii) a variational inference of $q(\tau_l)$, which is the VB-SAGE-M-step of the algorithm. Notice that the E-step of the VB-SAGE algorithm requires extending the approximating pdf (11) such as to account for the admissible hidden data $\boldsymbol{h}_l$. We assume that

$$q(\boldsymbol{\tau}, \boldsymbol{w}, \boldsymbol{\alpha}, \boldsymbol{h}_l) = q(\boldsymbol{\tau}, \boldsymbol{w}, \boldsymbol{\alpha})q(\boldsymbol{h}_l). \tag{18}$$

The same factorization of the approximating pdf also underpins the variational extension of the EM algorithm (Attias, 1999). Once the joint pdf (17) and the approximating pdf (18) are specified, the variational inference of $q(\boldsymbol{h}_l)$ and $q(\tau_l)$ is realized following the standard variational inference on a DAG, i.e., the expressions (12) and (13) are evaluated to estimate the corresponding approximating factors, albeit using the new graph in Fig. 1b to determine the Markov blanket of the updated variables.

It has been shown (Shutin and Fleury, 2011) that in order to guarantee the monotonic increase the variational lower bound with respect to $q(\tau_l)$ using the VB-SAGE algorithm, it suffice to estimate the approximating pdf $q(\boldsymbol{h}_l)$ of the admissible hidden data as a form-free solution (12) and select $\beta_l = 1$. In our case these constraints are easily satisfied. Note that β_l is in general a free parameter. However, setting $\beta_l = 1$ is convenient since it has been proven that for models linear in their parameters this choice leads to a fast convergence of the algorithm already in the early iteration steps (Fessler and Hero, 1994); the same choice has been also adopted in (Fleury et al., 1999; Shutin and Fleury, 2011). In case of $q(\boldsymbol{h}_l)$ it follows that due to (15) and (16) it is easy to demonstrate that $\log \tilde{p}(\boldsymbol{h}_l)$ is quadratic in $\boldsymbol{h}_l$ since $p(\boldsymbol{h}_l|\mathcal{MB}(\boldsymbol{h}_l)) \propto p(\boldsymbol{y}|\boldsymbol{h}_l, \boldsymbol{\tau}_{\bar{l}}, \boldsymbol{w})p(\boldsymbol{h}_l|w_{xl}, \tau_l)$ is Gaussian. Thus, by selecting $q(\boldsymbol{h}_l) = \mathrm{N}(\boldsymbol{h}_l|\widehat{\boldsymbol{h}}_l, \widehat{\boldsymbol{S}_l^h})$ the monotonicity property of the VB-SAGE scheme is guaranteed.

3.3 Variational estimation expressions

Here we provide the estimation expressions for the variational parameters of the approximating factors of $q(\boldsymbol{\tau}, \boldsymbol{w}, \boldsymbol{\alpha}, \boldsymbol{h}_l)$. The updated value of a variational parameter is denoted by $(\cdot)'$.

We begin with the variational estimation of $q(\boldsymbol{w})$. By evaluating $\log \tilde{p}(\boldsymbol{w})$ from (12), which is quadratic in $\boldsymbol{w}$, and minding that $q(\boldsymbol{w}) = \mathrm{N}(\boldsymbol{w}|\widehat{\boldsymbol{w}}, \widehat{\boldsymbol{S}^w})$, we find the updated variational parameters $\widehat{\boldsymbol{w}}$ and $\widehat{\boldsymbol{S}^w}$ as

$$\begin{aligned}(\widehat{\boldsymbol{S}^w})' &= \left(\sigma^{-2}\mathbb{E}_{q(\boldsymbol{\tau})}\left\{\boldsymbol{\Phi}(\boldsymbol{\tau})^T\boldsymbol{\Phi}(\boldsymbol{\tau})\right\} + \mathbb{E}_{q(\boldsymbol{\alpha})}\{\boldsymbol{A}\}\right)^{-1} = \left(\sigma^{-2}\widehat{\boldsymbol{\Phi}}^T\widehat{\boldsymbol{\Phi}} + \widehat{\boldsymbol{A}}\right)^{-1}, \\ \widehat{\boldsymbol{w}}' &= \sigma^{-2}(\widehat{\boldsymbol{S}^w})'\mathbb{E}_{q(\boldsymbol{\tau})}\{\boldsymbol{\Phi}(\boldsymbol{\tau})^T\}\boldsymbol{y} = \sigma^{-2}(\widehat{\boldsymbol{S}^w})'\widehat{\boldsymbol{\Phi}}^T\boldsymbol{y}.\end{aligned} \tag{19}$$

Here we defined $\widehat{\boldsymbol{\Phi}} = [\boldsymbol{X}(\widehat{\boldsymbol{\tau}}), \boldsymbol{U}]$ and $\widehat{\boldsymbol{A}} = \mathrm{diag}\{\widehat{\boldsymbol{\alpha}}\}$, where $\widehat{\boldsymbol{\alpha}} = [\widehat{\alpha}_1, \ldots, \widehat{\alpha}_K]^T$ and $\widehat{\alpha}_k = \mathbb{E}_{q(\alpha_k)}(\alpha_k) = \widehat{a}_k/\widehat{b}_k$, $k = 1, \ldots, K$. Let us stress that (19) is essentially a Tikhonov-like regularized solution for the coefficients $\boldsymbol{w}$, with $\widehat{\boldsymbol{\alpha}}$ acting as the regularization parameters.

Following the same inference steps we compute the variational parameters of the pdfs $q(\alpha_k) = \mathrm{Ga}(\alpha_k|\widehat{a}_k, \widehat{b}_k)$, $k = 1, \ldots, K$, as follows

$$\begin{aligned}\widehat{a}_k' &= a_k + 1/2, \\ \widehat{b}_k' &= b_k + \frac{1}{2}\mathbb{E}_{q(\boldsymbol{w})}\left\{|w_k|^2\right\} = b_k + \frac{1}{2}\left(|\widehat{w}_k|^2 + \widehat{S_k^w}\right), k = 1, \ldots, K.\end{aligned} \tag{20}$$

In (20) $\widehat{w}_k$ is a kth element of the vector $\widehat{\boldsymbol{w}}$, and $\widehat{S_k^w}$ is the kth element on the main diagonal of posterior covariance matrix $\widehat{\boldsymbol{S}^w}$.

Now, let us consider the VB-SAGE-based estimation of delay parameters $\boldsymbol{\tau}$. For each neuron the inference includes a VB-SAGE-E-step to estimate $q(\boldsymbol{h}_l)$ and a VB-SAGE-M-step to update the corresponding pdf $q(\tau_l)$. The VB-SAGE-E-step involves a computation of the expectation of $\log p(\boldsymbol{h}_l|\mathcal{MB}(\boldsymbol{h}_l))$ with respect to $\mathcal{MB}(\boldsymbol{h}_l) =$

$\{\boldsymbol{\tau}, \boldsymbol{w}, \boldsymbol{y}\}$. As we have mentioned earlier, $q(\boldsymbol{h}_l)$ should be selected as $q(\boldsymbol{h}_l) = \tilde{p}(\boldsymbol{h}_l)$ to ensure the monotonicity of the algorithm. Since $\log \tilde{p}(\boldsymbol{h}_l)$ is quadratic in $\boldsymbol{h}_l$, the variational parameters of $q(\boldsymbol{h}_l) = N(\boldsymbol{h}_l | \widehat{\boldsymbol{h}}_l, \widehat{\boldsymbol{S}}_l^h)$ can be easily computed as

$$\begin{aligned} \boldsymbol{h}_l' &= \boldsymbol{x}_l(\widehat{\tau}_l)\widehat{w}_{xl} + \beta_l \left(\boldsymbol{y} - \widehat{\boldsymbol{\Phi}}\widehat{\boldsymbol{w}}\right), \\ (\widehat{\boldsymbol{S}_l^h})' &= \beta_l(1-\beta_l)\sigma^2\boldsymbol{I}. \end{aligned} \tag{21}$$

Observe that with $\beta_1 = 1$, $\widehat{\boldsymbol{S}_l^h} \to 0$, i.e., $q(\boldsymbol{h}_l)$ collapses to a Dirac distribution. Now, the VB-SAGE-M-step involves a computation of the expectation of $\log p(\tau_l | \mathcal{MB}(\tau_l))$ with respect to $\mathcal{MB}(\tau_l) = \{\boldsymbol{w}, \boldsymbol{h}_l\}$. Since $q(\tau_l) = \delta(\tau_l - \widehat{\tau}_l)$, the solution to (13) is obtained by finding $\widehat{\tau}_l$ as a solution to the following optimization problem:

$$\widehat{\tau}_l' = \underset{\tau_l \in \{0,\dots,N-1\}}{\arg\max} \left\{ \log p(\tau_l) - \frac{1}{2\beta_l\sigma^2}\|\widehat{\boldsymbol{h}}_l - \widehat{w}_{xl}\boldsymbol{x}_l(\tau_l)\|^2 - \frac{\widehat{S_{xl}^w}}{2\beta_l\sigma^2}\|\boldsymbol{x}_l(\tau_l)\|^2 \right\}. \tag{22}$$

i.e., $\widehat{\tau}_l$ is found such that $q(\tau_l)$ is centered at the maximum of the $\tilde{p}(\tau_l)$; naturally, $\widehat{\tau}_l$ is the maximum a posteriori estimate of the delay parameter τ_l. In (22) $\widehat{S_{xl}^w}$ is the element on the main diagonal of $\widehat{\boldsymbol{S}^w}$ that corresponds to the posterior variance of the lth echo state weight w_{xl}. Notice that the estimation of the delay τ_l requires numerical optimization, which, however, can be implemented as a simple one-dimensional line search on the domain of $q(\tau_l)$. The VB-SAGE-E-step (21) and VB-SAGE-M-step (22) are then iteratively repeated for all L neurons. Let us also mention that due to $q(\boldsymbol{\tau})$ being fully factorizable, the neurons can be processed in any desired order.

In (Holzmann and Hauser, 2010) the authors propose a similar iterative EM-based scheme for D&S readout optimization. Their algorithm is in many respects inspired by the ideas of the original SAGE algorithm (Fessler and Hero, 1994). The authors propose to estimate the delay τ_l of the lth neuron by first subtracting the influence of the other echo states and network input signals from the desired network response $\boldsymbol{y}$ to compute the residual signal. This realizes the E-step of the scheme. The delay τ_l is then found as a value that maximizes the absolute value of the correlation between the computed residual and the echo state $x_l[n]$; this constitutes the M-step of the algorithm. Although the scheme is very effective, several heuristics are employed that distinguish it from the algorithm proposed in this work. Specifically, the weights of the echo states are estimated in two steps: during the D&S readout parameter updates, the weight w_{xl} of the lth echo state is computed as a projection of $\boldsymbol{x}_l(\tau_l)$ on the residual signal; then, once the delay parameters of the D&S readout have converged, the Moore-Penrose pseudoinverse is used to estimate the weights $\boldsymbol{w}$ one more time. Also, the objective function used to compute the delay parameters of the D&S readout differ from that obtained with the standard SAGE algorithm. Let us now show that (21) and (22) are the generalizations of this approach.

First, we note that with $\beta_l = 1$ the expression (21) naturally realizes the "interference cancellation" scheme of (Holzmann and Hauser, 2010). Indeed, in this case (21) reads

$$\boldsymbol{h}'_l = \boldsymbol{y} - (\boldsymbol{X}_{\bar{l}}(\widehat{\boldsymbol{\tau}_{\bar{l}}})\widehat{\boldsymbol{w}_{x\bar{l}}} + \boldsymbol{U}\widehat{\boldsymbol{w}_u}), \tag{23}$$

where $\widehat{\boldsymbol{\tau}_{\bar{l}}}$, $\widehat{\boldsymbol{w}_{x\bar{l}}}$, and $\widehat{\boldsymbol{w}_u}$ are the expectations of $\boldsymbol{\tau}_{\bar{l}}$, $\boldsymbol{w}_{x\bar{l}}$, and $\boldsymbol{w}_u$, respectively. In other words, in (23) all input signals and responses of the other neurons but the response of the lth neuron are subtracted from the target signal $\boldsymbol{y}$. The similarity between the VB-SAGE-E-step and the E-step of the scheme proposed in (Holzmann and Hauser, 2010) comes quite naturally, since both schemes use the SAGE algorithm as a starting point. The actual distinction lies in the way the D&S readout parameters are estimated. In their work the authors (Holzmann and Hauser, 2010) depart from the SAGE algorithm and use a heuristic to estimate the delay τ_l. Specifically, $\widehat{\tau}_l$ is found as a maximizer of the absolute value of the correlation $|\widehat{\boldsymbol{h}}_l^T \boldsymbol{x}_l(\tau_l)|$. Under certain assumptions, the objective function (22) can be shown to be very similar to that used in (Holzmann and Hauser, 2010).

Observe, that since τ_l is a delay parameter for the echo state $x_l[n]$, we can assume that the term $\|\boldsymbol{x}_l(\tau_l)\|^2$ is independent of the delay τ_l. This allows us to neglect the last "regularization" term $\frac{1}{2\beta_l\sigma^2}\widehat{S^w_{xl}}\|\boldsymbol{x}_l(\tau_l)\|^2$ in (22). Furthermore, when the prior $p(\tau_l)$ is assumed to be flat, i.e., $p(\tau) \propto 1$, it follows that the optimization problem (22) becomes equivalent to

$$\widehat{\tau}'_l = \underset{\tau_l \in \{0,\dots,N-1\}}{\arg\max} \{\widehat{w}_{xl}\widehat{\boldsymbol{h}}_l^T \boldsymbol{x}_l(\tau_l)\}, \tag{24}$$

which estimates τ_l on a grid such as to maximize the correlation between $\widehat{\boldsymbol{h}}_l$ and $\boldsymbol{x}_l(\tau_l)$.[9] The objective function used in (Holzmann and Hauser, 2010) is thus an "incoherent"

[9] In can be shown that in this case the minimum of $\|\widehat{\boldsymbol{h}}_l - \widehat{w}_{xl}\boldsymbol{x}_l(\tau_l)\|^2$ is achieved when the correlation between $\widehat{\boldsymbol{h}}_l$ and $\widehat{w}_{xl}\boldsymbol{x}_l(\tau_l)$ is maximized.

version of (24), where the weight $\widehat{w}_{xl}$ is ignored and only the magnitude of the correlation $\widehat{\boldsymbol{h}}_l^T \boldsymbol{x}_l(\tau_l)$ is maximized with respect to τ_l.

4 Implementation issues and algorithm initialization

In order to initialize the algorithm a simple strategy can be used that allows for an inference of the initial variational approximation from the training data $\{y[n], \boldsymbol{u}[n]\}_{n=n_0}^{n_0+N-1}$. For that we start with an empty model, i.e., assuming all variational parameters to be 0. The iterations of the algorithm are then sequentially update all variational factors. In Algorithm 1 we summarize the main steps of the proposed algorithm. Note that in the step 3 of the algorithm we initialize $\alpha_l = \epsilon$. The choice of ϵ is in general application dependent; we will discuss it in more details in Sec. 5.

Algorithm 1 Variational Bayesian ESN training

```
1:  Construct an ESN with $L$ neurons, D&S readouts and neuron filters.
2:  Use training data $\{y[n], \boldsymbol{u}[n]\}$ to generate echo states $x_l[n]$, $l = 1, \dots, L$.
3:  Initialize $\sigma^2$, $\widehat{\boldsymbol{\tau}}$, $\widehat{\boldsymbol{\alpha}}$, and $\widehat{\boldsymbol{w}}$.
4:  while not converged do
5:      for $l = 1 \dots L$ do
6:          Estimate $\widehat{\boldsymbol{h}}_l$ from (21) and update $\widehat{\tau}_l$ from (22)
7:      end for
8:      Update $\widehat{\boldsymbol{S}^w}$ and $\widehat{\boldsymbol{w}}$ from (19).
9:      Update $\widehat{a}_k$, $\widehat{b}_k$, $\forall k$ from (20) and recompute $\widehat{\boldsymbol{\alpha}}$,
10: end while
```

An important part of the initialization procedure is a selection of the additive perturbation variance σ^2. When signal $y[n]$ is known to be noisy, the variance σ^2 should be selected to reflect this. In general, a large value of σ^2 leads to a more aggressive regularization and makes the network less sensitive to variations in $y[n]$.

The iterative nature of the algorithm requires a stopping criterion for parameter updates. In our experiments it has been empirically determined that after 5–6 update iterations the improvment of the algorithm performance is insignificant; thus, a total of 6 iterations are used.

4.1 Computational complexity of the algorithm

Incorporation of automatic regularization in the ESN training scheme as well as estimation of D&S readout parameters increases the computational complexity of the network training. Quite naturally, when D&S readout parameters and regularization parameters are fixed, the variational Bayesian ESN training reduces to an instance of the classical ridge regression-based estimate of $\boldsymbol{w}$. This requires inverting a $K \times K$ posterior covariance matrix $\widehat{\boldsymbol{S}^{w}}$, an operation that has a computational complexity $\mathcal{O}(K^3)$.

The estimation of regularization parameters $\widehat{\boldsymbol{\alpha}}$ using (20) has complexity $\mathcal{O}(K)$. Compared to the computation of the weights $\boldsymbol{w}$, the estimation of regularization parameters poses an insignificant increase of the total computational complexity. Recently, a new fast variational SBL (FV-SBL) scheme has been proposed (Shutin et al., 2011a,b) to accelerate the convergence of sparsity parameter update expressions (20) in the case when hyperpriors $p(\alpha_k)$, $k = 1, \ldots, K$, are chosen to be non-informative. The scheme exploits the fact that the lower bound in (9) is convex with respect to the factorization (11); in other words, the factors in (11) can be updated in any order without compromising the monotonic increase of the variational lower bound (Bishop, 2006). Then, for a fixed k, the stationary point of variational updates (20) and (19) repeated *ad infinitum* can be computed in closed form assuming that the other variational parameters are fixed. All K variational factors $q(\alpha_k)$, $k = 1, \ldots, K$, are then updated sequentially, with the complexity of a single update being on the order of $\mathcal{O}(K^2)$. Although in general the total complexity remains $\mathcal{O}(K^3)$, the update of a single component can be performed more efficiently. Furthermore, fewer iterations are typically needed to estimate the regularization parameters.

The estimation of D&S readout parameters also increases the total computational complexity. Specifically, the estimation of the delay parameter for each neuron from (22) requires evaluating the admissible hidden data from (21), an $\mathcal{O}(NK)$ operation, and solving the optimization problem (22), which is an $\mathcal{O}(N^2)$ operation. For L neurons this results in a total computational complexity on the order of $\mathcal{O}(LNK + LN^2)$; i.e., it is quadratic[10] in both the number of learning samples N as well as in the number of neurons L; yet this increase is still dominated by $\mathcal{O}(K^3)$ complexity of estimating

[10] Recall that $K = M + L$.

the network weights. Note that an ESN with D&S readout typically requires fewer neurons as compared to the standard ESN to achieve the same memory capacity; in other words, training a smaller ESN with tunable D&S readouts is typically more efficient than training a standard ESN with many neurons.

5 Simulation results

In this section we compare the performance of the proposed variational Bayesian learning of ESNs with other state-of-the-art ESN training algorithms using synthetic as well as real-world data. In the first experiment, described in Section 5.1 we train an ESN predictor to forecast a chaotic time series generated with a *Mackey-Glass* system, which is often used to benchmark ESN learning schemes (Jaeger, 2001; Holzmann, 2008). In the second experiment, described in Section 5.2, we apply the ESN training schemes to a recognition of handwritten symbols based on measured dynamic pen trajectory data.

In both experiments we compare an extended ESN trained with the proposed VB-SAGE algorithm (which we will further term VB-ESN) to the performance of i) a standard ESN (STD-ESN), ii) an ESN trained using Moore-Penrose pseudoinverse and reservoir extended with fixed D&S readout (EXT-ESN), and iii) an ESN with D&S readout that is trained using the algorithm proposed in (Holzmann and Hauser, 2010) (further in the text we will refer to this algorithm as HH-ESN).

5.1 Time-series prediction

In this experiment we apply ESNs to predict a chaotic time series generated with a Mackey-Glass system. Similar experiments have also been performed in (Jaeger, 2001; Holzmann, 2008) to benchmark the performance of different ESN training schemes. We assume that an input signal to an ESN is a constant signal $u[n] = 0.02$ and an output signal is generated using the Mackey-Glass differential equation

$$\frac{\mathrm{d}y(t)}{\mathrm{d}t} = \beta \frac{y(t-\tau_{mg})}{1 + y(t-\tau_{mg})^n} - \gamma y(t), \tag{25}$$

where γ, β, n and τ_{mg} are the parameters of the system. Following (Jaeger, 2001; Holzmann, 2008) we select these parameters as follows: $\gamma = 0.1$, $\beta = 0.2$, $n = 10$ and

$\tau_{mg} = 30$. This choice guarantees that the Mackey-Glass system converges to a chaotic attractor.

The reservoir coefficients $\boldsymbol{C}_x$, $\boldsymbol{C}_y$ and $\boldsymbol{C}_u$ are randomly generated by uniformly drawing samples from the interval $[-1, 1]$. For all tested networks the connectivity of the reservoir is set to 5% and the connectivity matrix $\boldsymbol{C}_x$ is normalized so as to have a spectral radius of 0.8. To avoid instabilities due to the nonlinear feedback mechanism (Jaeger, 2001), a small zero-mean white additive disturbance with variance 1×10^{-6} was added to the feedback signal $\boldsymbol{C}_y^T \boldsymbol{y}[n]$ in the state transition equation (1). We set the size of the reservoir for all tested ESNs to $L = 200$ neurons, unless explicitly stated otherwise. The variance of the additive noise $\boldsymbol{\xi}$ was set in this experiment to $\sigma^2 = 10^{-10}$.

For the EXT-ESN algorithm the time delays are generated randomly by independently drawing samples from the interval $[0, 100]$; 50% of the generated delay values are then set to zero, which ensures that a particular number of echo state functions enter the ESN output without a time delay. Similarly, the VB-ESN and HH-ESN algorithms use this initialization to generate the initial values of neuron delays.

In the case of the VB-SAGE algorithm it is important to mention that due to the iterative structure of the algorithm the proper initialization of the network parameters plays an important role. To obtain a consistent starting point, the initial values of the weights $\boldsymbol{w}$ are drawn from the Gaussian distribution with zero mean and covariance matrix $\boldsymbol{A}^{-1}$, where $\boldsymbol{A} = \mathrm{diag}\{\boldsymbol{\alpha}\}$. Obviously, the initial choice of $\boldsymbol{\alpha}$ controls the algorithm's emphasis on the estimation of the time delays $\boldsymbol{\tau}$. Small initial values of $\boldsymbol{\alpha}$ lead to weak regularization of the weights during the early iterations. We have observed that this often drives the algorithm to a local optimum, with the values of $\boldsymbol{\tau}$ "frozen" at the initial values. Setting initial values of $\boldsymbol{\alpha}$ to large numbers corresponds to the initial weights $\boldsymbol{w}$ being close to zero; as a result, the training algorithm essentially "de-regularizes" the solution, which, as our extensive simulations show, leads to better estimation results. In our experiments we set $\alpha_k = \epsilon = 10^{10}$, $k = 1 \ldots K$. The same strategy is also used to initialize the weights of the HH-ESN algorithm.

The training and testing of the ESNs are then realized as follows. First, 3300 samples of Mackey-Glass time series are generated. The first 3000 samples are used to train the network and the remaining 300 are used to validate the network performance. The network is then run from a zero initial state in teacher-forced mode (Jaeger, 2001) using the first 3000 samples of the time series; further, the initial 1000 samples of the resulting network trajectory are discarded to ensure that the system settles at the chaotic attractor. The remaining $N = 2000$ samples are used to train the network and estimate its parameters.

Once the coefficients of the network are estimated, the trained network is run for 300 time steps to generate the predicted trajectory by feeding the output of the trained network back into the reservoir. The performance of the trained network is evaluated by measuring the normalized root-mean squared error between the 300 samples of the true trajectory $y_{\text{true}}[n]$ and the trajectory $y_{\text{trained}}[n]$ generated by the trained network; the corresponding results are then averaged over $N_{\text{MC}} = 300$ independent Monte Carlo simulations, where for each simulation run a new ESN is generated and trained using a new realization of the Mackey-Glass time series. The normalized root-mean squared error between $y_{\text{true}}[n]$ and $y_{\text{trained}}[n]$ is computed as follows:

$$\text{NRMSE}[n] = \sqrt{\frac{1}{N_{\text{MC}}} \sum_{i=1}^{N_{\text{MC}}} \frac{\left| y_{\text{true}}^{[i]}[n] - y_{\text{trained}}^{[i]}[n] \right|^2}{\sigma_{y_{\text{true}}}^2}}, \tag{26}$$

where the superscript $[i]$ denotes the signal computed during the ith Monte Carlo simulation run, and $\sigma_{y_{\text{true}}}^2$ is the variance of the true time series $y_{\text{true}}[n]$. Naturally, the longer both systems remain synchronized, i.e., the more slowly $\text{NRMSE}[n]$ grows as a function of n, the better the performance of the trained model is.

In Fig. 2 we plot the estimated performance of the compared algorithms. Observe that the predicted output signal obtained with the STD-ESN scheme diverges much faster from the true signal as compared to the other algorithms; even doubling the size of the network from 200 to 400 neurons does not improve the performance. Introducing the random D&S readout, however, does help. However, although the EXT-ESN scheme with $L = 200$ neurons outperforms both STD-ESN schemes, its performance is below that of the VB-ESN and HH-ESN algorithms. The latter two schemes deliver the lowest prediction error. These algorithms still have a 30 dB performance gain over the STD-ESN and EXT-ESN after 300 time steps. Solely boosting the size of the EXT-ESN to 400 neurons makes this scheme perform on par with VB-ESN and HH-ESN with 200 neurons each. Thus, learning the optimal parameters of the D&S readout allows significantly reducing the required size of the network. It should be mentioned that in this

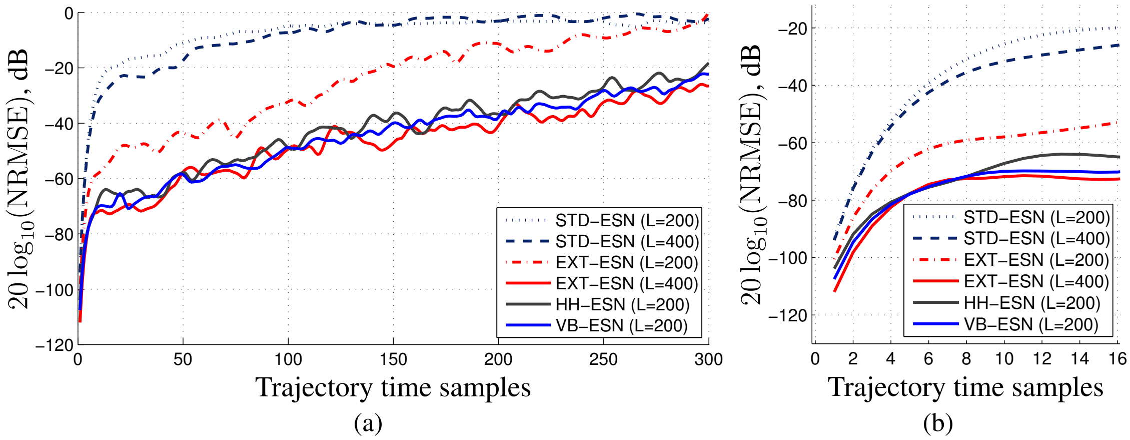


Figure 2: An error between the true Mackey-Glass time series and the predicted response for a) 300 time steps; b) a zoom-in into the error evolution for time steps between $n = 1$ and $n = 16$

example the performance of both VB-ESN and HH-ESN schemes is nearly identical. Let us look into the performance of these two scheme a bit closer.

For that we analyze the estimation results for the delays $\boldsymbol{\tau}$. In Figures 3a and 3b we plot the histograms of the estimated D&S readout parameters with non-zero delay values computed with the HH-ESN and VB-ESN methods. Interestingly, the histogram for

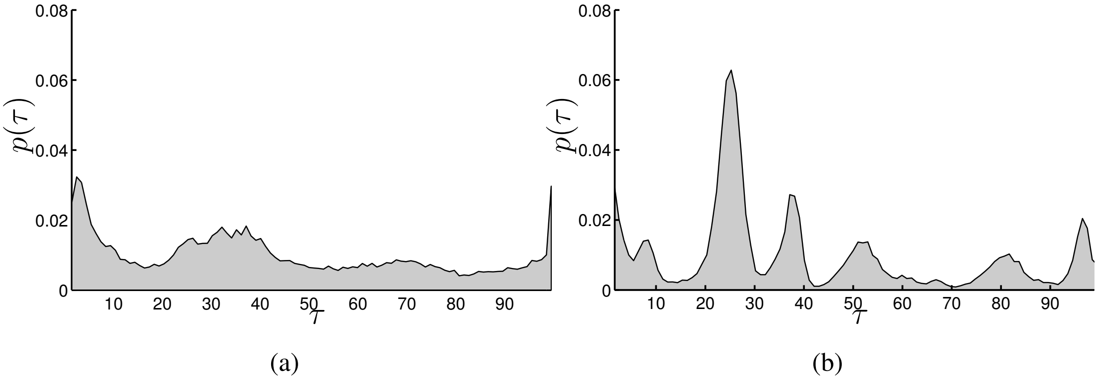


Figure 3: Histogram of the estimated D&S readout parameters. a) HH-ESN, b)VB-ESN.

the D&S readout parameters computed with the VB-ESN algorithm shows strong peaks in the range between $\tau = 20$ and $\tau = 60$, which covers the original delay parameter τ_{mg} of the Mackey-Glass system. In fact, this is not a mere coincidence; experiments with different values of the delay parameter τ_{mg} indicate that the VB-SAGE algorithm indeed sets many of the D&S readout delays to the value closest to the true delay τ_{mg}. In contrast, the delay estimation with the HH-ESN algorithm seems to result in more uniform values of the estimated delays and thus shows only weak peaks around the true delay parameter. Based on the obtained simulation results we can claim that the exact estimates of the D&S readout parameters are not pivotal for the successful prediction of the Mackey-Glass time series and deviations in the delay parameter estimates can be compensated to a certain extent by the estimation of output weights. Also, due to a very low noise level, the distinction between the Bayesian regularization used in the VB-ESN and Moore-Pensore pseudoinverse-based regularization is also minimal. This explains the similarity of the prediction results obtained with the two schemes.

It is also important to stress that there is a statistical dependency between the estimated delay parameters $\boldsymbol{\tau}$ and estimated regularization parameters $\boldsymbol{\alpha}$. Recall that $\boldsymbol{\alpha}$ reflect the importance of the particular echo states or input signals: the higher the value of α_k, the more regularization is applied to the kth column in the matrix $\mathbf{\Phi}(\tau)$ in (3) and, thus, the less relevant this column is in predicting the output signal. Thus, parameters $\boldsymbol{\alpha}$ can be used to measure the relative "quality" of individual neuron echo states. In Fig. 4 we plot the values of regularization parameters $\boldsymbol{\alpha}$ versus the corresponding D&S readout delays parameters for the Mackey-Glass time series prediction example. Notice

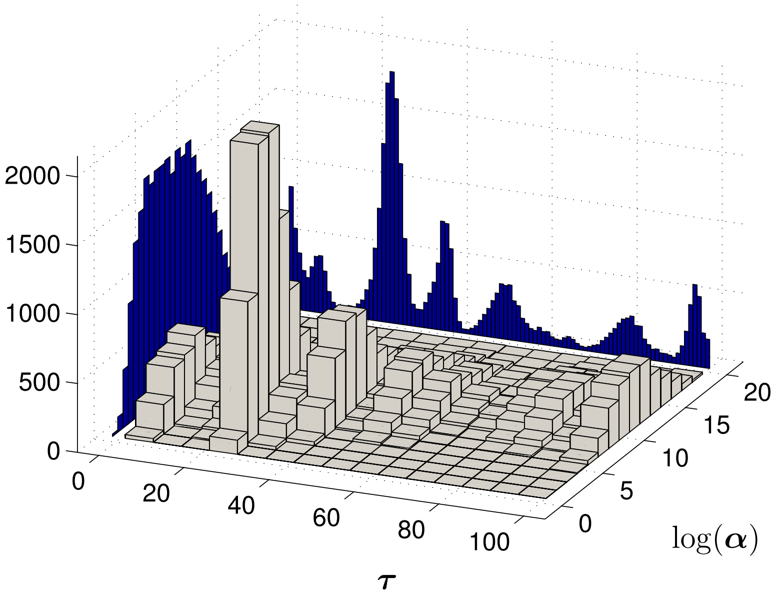


Figure 4: An empirical distribution of the D&S readout parameters $\boldsymbol{\tau}$ and regularization parameters $\boldsymbol{\alpha}$ for $L = 200$ neurons.

that the histogram has a strong peak at $\tau \approx 25$ and $\tau \approx 35$ with relatively small values of α, which indicates an importance of the echo states with these delays for representing the desired output signal.

5.2 Handwritten character recognition

Here we assess the performance of the proposed algorithm using measured multidimensional pen trajectory data for handwritten character recognition. It was already demonstrated (Zechner, 2010) that ESNs can successfully handle dynamic handwriting data. Here we adopt the same experimental setup as used in (Zechner, 2010) to test the performance of the VB-SAGE algorithm.

The following simulations are carried out using samples from the WILLIAMS database (Williams, 2010), available at the UCI Machine Learning Repository (Frank and Asuncion, 2010). The database contains 2858 character trajectories from an English alphabet, where only letters that can be written as a single stroke were recorded (i.e., 20 character classes). Furthermore, each trajectory is represented as a three-dimensional time series, featuring $X-$ and $Y-$ velocities as well as the pen pressure. The measured data in the repository have been smoothed using a Gaussian filter with a variance set to 4 (see (Williams, 2010) for further details). For our purposes we will consider recognition of only a subset of characters from the repository, namely, “a”, “b”, “c”, “d”, “e”, “g”, and “h”;[11] the corresponding 2D patterns for these characters are shown in Figure 5. It has been noted in (Lukoševičius and Jaeger, 2009) that there are two ESN configurations for a classification task. First, one can design and train a single ESN with as many outputs

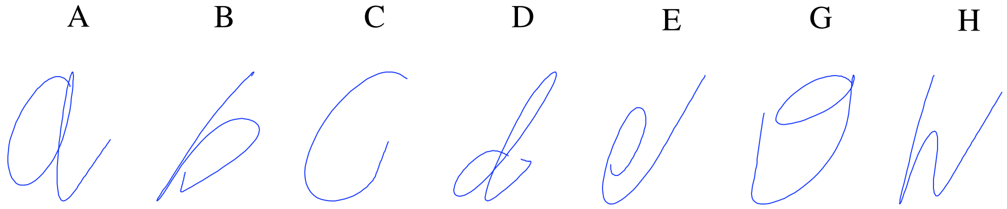


Figure 5: Sample X-Y-trajectories for letters “a”, “b”, “c”, “d”, “e”, “g”, and “h” from the WILLIAMS database.

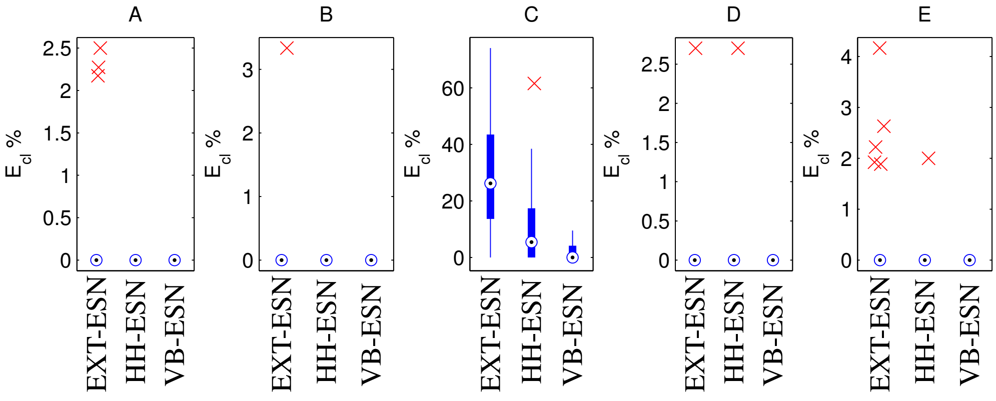
[11] The letter “f” was not recorded for the WILLIAMS database as it cannot be represented as a single stroke.

as class labels; the classes are then predicted by the output with the largest amplitude. Alternatively, one can train several ESN predictors, one for each class; then, given a test signal, the class label is selected by choosing the ESN predictor that achieves the smallest prediction error. In this work we use the first approach.

Let us now describe the settings for the ESN network parameters and the design of the input signals. To this end we introduce an index set $\mathcal{C} = 1, \ldots, 7$ with each element corresponding to one of the 7 letters. As an input signal $\boldsymbol{u}[n] \equiv \boldsymbol{u}_c[n]$ we use the trajectory corresponding to the class $c \in \mathcal{C}$; the corresponding output signal of the ESN $\boldsymbol{y}[n] = [y_1[n], \ldots, y_7[n]]^T$ is then set to zero except for the element $y_c[n]$, $c \in \mathcal{C}$, which is selected as a Gaussian pulse with variance 1 centered at the time instance corresponding to 70% of the input trajectory length. During the testing a class estimate is obtained by selecting the output element of $\boldsymbol{y}[n]$ which shows the maximum value within the input trajectory's time window. The reservoir parameters for this classification task are selected as in the signal prediction example except for the reservoir connectivity, which is set to 10%. Also, no output feedback is used, i.e., $\boldsymbol{C}_y = 0\boldsymbol{I}$. For each of the EXT-ESN algorithms 30% of the D&S readout delays are set to zero, whereas the remaining 70% are uniformly drawn from the interval $[0, 100]$. The algorithms VB-ESN and HH-ESN use this initialization to generate the initial values of the D&S readout parameters.

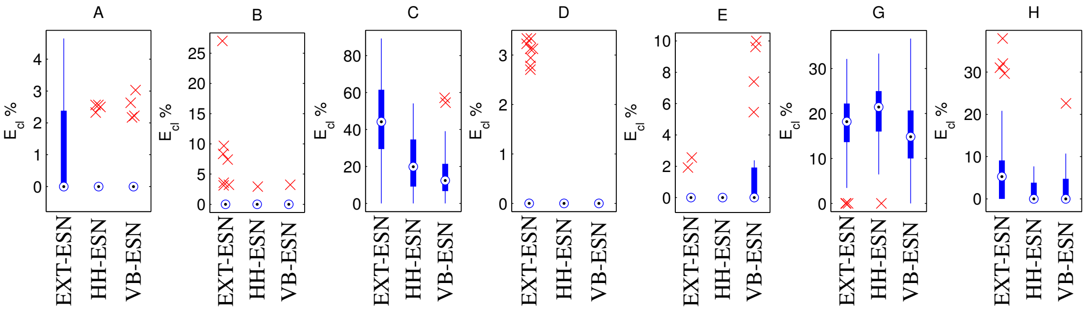
The classification tasks were performed using five as well as seven character classes, i.e., {"a", "b", "c", "d", "e"} and {"a", "b", "c", "d", "e", "g", "h"}. In both cases 60% of the character instances are used for training and the remaining 40% are used for testing. As the performance measure we compute a per-class classification error rate E_{cl} as the number of incorrect classifications per class over the number of tested examples in this class and the total classification error rate E_{total}, which is the number of incorrect classifications for all letters over the total number of tested examples. To better assess the classification performance of the compared algorithms we estimate E_{cl} and E_{total} over 50 independent runs; for each run a new ESN reservoir is generated and trained and the corresponding classification errors are estimated.

In Fig. 6 we summarize the distributions of the per class classification errors E_{cl} using box-and-whiskers plots for 5 letter case; in Fig. 7 the classification errors for the 7 letter case are presented. The edges of the boxes are 25 and 75 percentiles of the estimated classification errors and the central mark denotes the median. Whiskers



(a) (b) (c) (d) (e)

Figure 6: Classification results for 5 letters case.



(a) (b) (c) (d) (e) (f) (g)

Figure 7: Classification results for 7 letters case.

illustrate the degree of error dispersion; they extend from the box to the most extreme data value within $1.5 \times \mathrm{IQR}$, where IQR is the interquantile range of the sample. The data with values beyond the ends of the whiskers, marked with crosses, are treated as outliers.

In the 5 letter case the compared algorithms successfully classify the symbols with a single exception of the letter "c". Note that in contrast to previous example the VB-ESN by far outperforms the other schemes; moreover, the distinction between the VB-ESN and HH-ESN schemes is now much more apparent. The HH-ESN and EXT-ESN schemes also produce more outliers as compared to the VB-ESN algorithm. The failure of all three schemes to achieve low classification error rate for the letter "c" can be explained by its similarity to the letter "e". The analysis of the confusion matrix, shown in Table 1a, reveals that the letter "c" is indeed often predicted as "e". This leads to a higher dispersion of the classification errors, as can be seen in Fig. 6c. Interestingly, the reverse is not true: letter “e” is less often confused with “c”. Notice that the VB-ESN scheme is more successful in properly classifying “c”, having the smallest dispersion of E_{cl}. The same tendency is observed when when total classification error is analyzed.

	A	B	C	D	**E**
A	1859	0	1	0	2
B	0	1424	0	1	0
C	8	0	761	0	**327**
D	0	0	0	1623	1
E	0	0	6	0	2219

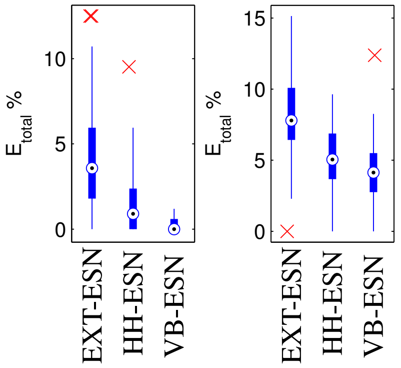
(a) 5 letter case

	A	B	C	D	**E**	G	H
A	5655	0	1	0	23	0	0
B	0	4307	0	20	0	0	3
C	20	0	2288	1	**919**	0	0
D	0	0	0	4819	8	0	0
E	0	0	30	0	6577	0	0
G	**681**	0	1	7	16	3223	0
H	4	42	9	106	0	0	3286

(b) 7 letter case

Table 1: Confusion matrices for a) 5 letter case and b) 7 letter case computed jointly by EXT-ESN, HH-ESN and VB-ESN schemes. Rows correspond to actual letters and columns to predictions.

In the 5 letter case (see Fig. 8a) the VB-ESN has much lower dispersion of the classification error as compared to HH-ESN and EXT-ESN cases. These results clearly demonstrate that the proposed joint optimization of the D&S readout parameters and regularization parameters significantly improves the performance of the trained models as compared to the other training schemes.



(a) (b)

Figure 8: Total classification error E_{total} for (a) 5 letters case and (b) 7 letter case.

In the 7 letter classification scenario we see that increasing the number of classes makes the classification clearly more difficult. Specifically, all schemes make now more errors. Similarly to the 5 letter case, the letter “c” is often confused with the letter “e”, as can be seen from the confusion matrix in Table 1b. Also, the letter “g” is often confused with “a” by all compared algorithms. Interestingly, the HH-ESN is able to classifying the letter “e” without errors, while VB-SAGE is not (see Fig 7e). This can be explained as follows: increasing the complexity of the classification problem with a fixed reservoir size not only increases the classification errors, but, in a multi-class classifier that we employ here, also “redistributes” the errors between the classes. If we consider the distribution of the total classification error, shown in Fig. 8b, we will see that in contrast to the 5 letter case, the performance of all schemes degrades; yet, the performance of the VB-ESN algorithm is still slightly better than that of the other compared schemes.

6 Conclusion

In this work we have proposed a variational Bayesian approach to automatic regularization and training of extended Echo State Networks with Delay&Sum readouts. The proposed training framework combines sparse Bayesian learning methods with variational Bayesian Space-Alternating Generalized Expectation-Maximization (VB-SAGE) algorithm. We have demonstrated that the proposed scheme allows for an optimal regularization of the training algorithm, with regularization parameters being determined automatically by the input-output signals, additive noise, and the structure of the reservoir. The standard Tikhonov-like regularization of ESN training is obtained as a special case of the proposed approach. The estimated regularization parameters also provide an objective measure of the weights’ importance: excessive regularization required for some of the echo states or input signals indicates the irrelevance of these signals to the approximation of the target signal. This importance information, together with the estimated delay parameters of the D&S readout, can be potentially used for relating the structure of the reservoir and neuron responses to different features of the training data. However, a detailed analysis is required to support this claim, which is beyond the scope of this paper.

In addition to automatic regularization, the standard ESN structure has also been extended with tunable Delay&Sum readouts and filter neurons and, when filter neurons are fixed, the D&S readout parameters can be efficiently estimated using the VB-SAGE algorithm. Although in general the optimization of neuron parameters leads to an intractable nonlinear optimization, the variational approach allows for a reduction of the optimization problem to a sequence of simpler one-dimensional searches with respect to the delay parameter of each neuron. The VB-SAGE-based estimation of the D&S readout parameters generalizes the ad-hoc EM-based D&S readout parameter estimation proposed by Holzmann and Hauser in (Holzmann and Hauser, 2010). Thus, the variational Bayesian framework for ESN training presented in this work generalizes some of the existing approaches to regularization and D&S readout parameter estimation, while at the same time providing a formal optimization framework for joint ESN training, regularization and parameter estimation.

The proposed estimation scheme has been applied to forecasting a chaotic time series generated with a Mackey-Glass system and dynamic handwritten symbol recognition problem. Our results demonstrate that for time-series prediction the proposed variational approach outperforms a simple extended ESN with random D&S readout parameters and performs on par only with the algorithm proposed in (Holzmann and Hauser, 2010). However, in a handwritten character recognition problem the advantages of the proposed training algorithm become more apparent. Specifically, the proposed training scheme consistently outperforms the other algorithms, while only marginally increasing the computational complexity as compared to the training scheme discussed in (Holzmann and Hauser, 2010).

Acknowledgement

The authors would like to thank the reviewers for providing valuable comments that helped to significantly improve the manuscript.

This research was supported in part by an Erwin Schrödinger Postdoctoral Fellowship, Austrian Science Fund Project J2909-N23, in part by the U. S. Office of Naval Research under Grant N00014-09-1-0342, and in part by the Center for Science of Information (CSoI), an NSF Science and Technology Center, under grant agreement CCF-0939370, and by the U.S. Army Research Office under grant number W911NF-07-1-0185.

References

Attias, H. (1999). Inferring parameters and structure of latent variable models by variational Bayes. In *Proc. 15th. Conference on Uncertainty in Artificial Intelligence, UAI '99*, volume 2, Stockholm, Sweden.

Beal, M. (2003). *Variational Algorithm for Approximate Bayesian Inference*. PhD thesis, University College London.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer, New York, NY.

Bishop, C. M. and Tipping, M. E. (2000). Variational relevance vector machines. In *Proc. 16th Conference on Uncertainty in Artificial Intelligence, UAI '00*, pages 46–53, San Francisco, CA, USA.

Feder, M. and Weinstein, E. (1988). Parameter estimation of superimposed signals using the EM algorithm. *IEEE Transactions on Acoustics, Speech, and Signal Processing.*, 36(4):477–489.

Fessler, J. and Hero, A. (1994). Space-alternating generalized expectation-maximization algorithm. *IEEE Transactions on Signal Processing*, 42(10):2664–2677.

Fleury, B., Tschudin, M., Heddergott, R., Dahlhaus, D., and Pedersen, K. I. (1999). Channel parameter estimation in mobile radio environments using the SAGE algorithm. *IEEE Journal on Selected Areas in Communications*, 17(3):434–450.

Frank, A. and Asuncion, A. (2010). UCI machine learning repository. http://archive.ics.uci.edu/ml/.

Golub, G. H. and Van Loan, C. F. (1996). *Matrix Computations (Johns Hopkins Studies in Mathematical Sciences)*. The Johns Hopkins University Press, Baltimore, MD, 3rd edition.

Han, M. and Wang, Y. (2009). Nonlinear time series online prediction using reservoir Kalman filter. In *Proc. International Joint Conference on Neural Networks IJCNN '09*, pages 1090–1094, Atlanta, Georgia.

Holzmann, G. (2008). Echo State Networks with Filter Neurons and a Delay&Sum Readout with Applications in Audio Signal Processing. Master's thesis, Graz University of Technology.

Holzmann, G. and Hauser, H. (2010). Echo state networks with filter neurons and a delay&sum readout. *Neural Networks*, 23(2):244 – 256.

Jaeger, H. (2001). The echo state approach to analyzing and training recurrent neural networks. Technical Report 148, German National Research Institute for Computer Science, Sankt Augustin, Germany.

Jaeger, H. (2007). Discovering multiscale dynamical features with hierarchical echo state networks. Technical report 10, School of Engineering and Science, Jacobs University, Bremen, Germany.

Jaeger, H., Maass, W., and Principe, J. (2007). Special issue on echo state networks and liquid state machines. *Neural Networks*, 20(3):287–289.

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

MacKay, D. J. C. (1994). Bayesian Methods for Backpropagation Networks. In Domany, E., van Hemmen, J. L., and Schulten, K., editors, *Models of Neural Networks III*, chapter 6, pages 211–254. Springer-Verlag, New York, NY.

Neal, R. (1996). *Bayesian Learning for Neural Networks*, volume 118 of *Lecture Notes in Statistics*. Springer-Verlag, New York, NY.

Palmer, J., Wipf, D., Kreutz-Delgado, K., and Rao, B. (2006). Variational EM algorithms for non-Gaussian latent variable models. In Weiss, Y., Schölkopf, B., and Platt, J., editors, *Advances in Neural Information Processing Systems 18*, pages 1059–1066, Cambridge, MA. MIT Press.

Seeger, M. and Wipf, D. (2010). Variational Bayesian inference techniques. *IEEE Signal Processing Magazine*, 27(6):81 –91.

Shutin, D., Buchgraber, T., Kulkarni, S. R., and Poor, H. V. (2011a). Fast adaptive variational sparse Bayesian learning with automatic relevance determination. In

Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP'10, pages 2180–2183, Prague, Czech Republic.

Shutin, D., Buchgraber, T., Kulkarni, S. R., and Poor, H. V. (2011b). Fast variational sparse Bayesian learning with automatic relevance determination for superimposed signals. *IEEE Transactions on Signal Processing*. (to appear).

Shutin, D. and Fleury, B. H. (2011). Sparse variational Bayesian SAGE algorithm with application to the estimation of multipath wireless channels. *IEEE Transactions on Signal Processing*, 59(8):3609 –3623.

Sung, J., Ghahramani, Z., and Bang, S.-Y. (2008a). Latent-space variational Bayes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(12):2236–2242.

Sung, J., Ghahramani, Z., and Bang, S.-Y. (2008b). Second-order latent-space variational Bayes for approximate Bayesian inference. *IEEE Signal Processing Letters*, 15:918–921.

Tipping, M. (2001). Sparse Bayesian learning and the relevance vector machine. *Journal of Machine Learning Research*, 1:211–244.

Tipping, M. E. and Faul, A. C. (2003). Fast marginal likelihood maximisation for sparse Bayesian models. In *Proc. 9th International Workshop on Artificaial Intelligence and Statistics*, Key West, FL, USA.

Tzikas, D. G., Likas, A. C., and Galatsanos, N. P. (2008). The variational approximation for Bayesian inference. *IEEE Signal Processing Magazine*, 25(6):131–146.

Verstraeten, D., Schrauwen, B., and Stroobandt, D. (2006). Reservoir-based techniques for speech recognition. In *Proc. International Joint Conference on Neural Networks IJCNN '06*, pages 1050–1053, Vancouver, BC, Canada.

Verstraeten, D., Schrauwen, B., and Stroobandt, D. (2007). An experimental unification of reservoir computing methods. *Neural Networks*, 20(3):391–403.

Williams, B. H. (2010). UCI character trajectories. http://archive.ics.uci.edu/ml/datasets/.

Wustlich, W. and Siewert, U. (2007). Echo-State Networks with Band-Pass Neurons: Towards Generic Time-Scale-Independent Reservoir Structures. Technical report, PLANET Intelligent Systems GmbH., Raben Steinfeld,Germany.

Xia, Y., Mandic, D. P., Hulle, M., and Principe, J. C. (2008). A complex echo state network for nonlinear adaptive filtering. In *Proc. IEEE Workshop on Machine Learning for Signal Processing MLSP'08*, pages 404–408, Cancun, Mexico.

Zechner, C. (2010). Variational Bayesian Reservoir Computing and its Applications to Handwriting Recognition. Master's thesis, Graz University of Technology.

Zechner, C. and Shutin, D. (2010). Bayesian learning of echo state networks with tunable filters and delay&sum readouts. In *Proc. of International Conference on Acoustics Speech and Signal Processing, ICASSP'10*, Dallas, TX.