Relative entropy minimizing noisy non-linear neural network to approximate stochastic processes

Mathieu N Galtier, Camille Marini, Gilles Wainrib, H. Jaeger

Relative entropy minimizing noisy non-linear neural network to approximate stochastic processes

Mathieu N Galtier[1], Camille Marini[2,3], Gilles Wainrib[4], and Herbert Jaeger[1]

[1]School of Engineering and Science, Jacobs University Bremen gGmbH, 28759 Bremen, Germany
[2]Institut für Meereskunde, Zentrum für Meeres- und Klimaforschung, Universität Hamburg, Hamburg, Germany
[3]MINES ParisTech, 1, rue Claude Daunesse, F-06904 Sophia Antipolis Cedex, France
[4]Laboratoire Analyse Géométrie et Applications, Université Paris XIII, France

April 14, 2014

Abstract

A method is provided for designing and training noise-driven recurrent neural networks as models of stochastic processes. The method unifies and generalizes two known separate modeling approaches, Echo State Networks (ESN) and Linear Inverse Modeling (LIM), under the common principle of relative entropy minimization. The power of the new method is demonstrated on a stochastic approximation of the El Niño phenomenon studied in climate research.

1 Introduction

Blackbox modeling methods for stochastic systems have a broad range of applications in physics, biology, economy or the social sciences. Generally speaking, a model of a stochastic system is a representation of the conditional distribution of the system's future given the present state (Markov models) or some part or the entire system past. There is a large variety of such stochastic predictors among which we focus on generic methods which do not depend on the type of data considered. The Auto-Regressive-Moving-Average (ARMA) models [Box et al., 2013] form a class of linear stochastic approximators which has led to many derivative works and is widely used in engineering applications. In particular, it covers the case of multivariate linear stochastic differential equations (SDE), or Ornstein-Uhlenbeck processes, which is the basic structure used in the Linear Inverse Modeling (LIM) theory [Penland and Magorian, 1993]. ARMA models are generally learnt by optimizing a least squares measure of the prediction error. A notable characteristic of ARMA models is that the dimension of the underlying SDE is identical to the observable dimension of the target time series. By contrast, dynamic Bayesian networks [Murphy, 2002], with Hidden Markov Models (HMM)[Baum and Petrie, 1966, Rabiner, 1989] as the most widely employed special case, rely on hidden variables. HMM are trained by maximum likelihood schemes, typically with some version of the expectation maximization algorithm [Dempster et al., 1977, Moon, 1996]. A problem with dynamic Bayesian networks, inherited from their simpler static counterparts, is that inference (e.g. prediction) quickly becomes computationally expensive when the dependency structure of hidden variables is not particularly simple (as it is in HMMs). The Temporal Restricted Boltzmann Machine [Sutskever and Hinton, 2006], a recent addition to the spectrum of such models, is a point in case. With the advent of kernel machines in machine learning community, models based on Gaussian Processes have been designed to approximate stochastic processes [Rasmussen, 2006]. A critical point regarding these models lies in their computational complexity when working with long time series. There is also a large body of literature about online adaptive predictors, e.g.

Kalman filters [Haykin, 2005]. In this paper however we focus on non-adaptive models trained on all available training data using a batch algorithm.

Recurrent neural networks (RNNs) have also been used in various ways for approximating stochastic dynamical systems. In their basic forms [Williams and Zipser, 1995, Pearlmutter, 1995], RNNs are models of deterministic dynamical systems; if trained on data sampled from stochastic sources, at exploitation time such RNNs will not propose future distributions but only a single expected mean future trajectory. RNNs represent, in principle, a promising model class because they are dense in interesting classes of target systems, implying that arbitrarily accurate models can in principle be found [Funahashi and Nakamura, 1993, Sontag, 1997]. Gradient-descent based learning algorithms for RNNs are typically computationally expensive and cannot be guaranteed to converge. Since about a decade, an alternative approach to RNN design and training, now generally called reservoir computing [Jaeger and Haas, 2004, Maass et al., 2002], has overcome the problem of learning complexity. The key idea in this field is not to train all parameters of an RNN but only the weights of connections leading from the RNN "body" (called reservoir) to the output neurons. Here we will build on a particular instantiation of reservoir computing called Echo State Networks (ESNs).

Although deterministic models at the outset, neural network architectures for predicting future distributions have been variously proposed [Husmaier and Taylor, 1997, Buesing et al., 2011], or neural networks were embedded as components in hybrid models of stochastic systems [Krogh and Riis, 1999, Chatzis and Demiris, 2011]. Here we propose a novel way to use RNNs in a stochastic framework based on the way stochasticity is taken into account in LIM. LIM consists in tuning both the drift and the diffusion term of an Ornstein-Uhlenbeck process to approximate a stochastic process. First, the drift is optimized to approximate the time series as if it were deterministic; then, the diffusion is chosen so that the variances of both systems are identical. LIM is widely used in climate research and stands as a simple approach giving relatively good results [Penland, 1996, Hawkins et al., 2011, Zanna, 2012, Barnston et al., 2012, Newman, 2013].

To compare two stochastic processes, and thus to define what it means to approximate a stochastic process, we use the relative entropy (also known as Kullback-Leibler divergence) [Kullback and Leibler, 195 Although not a true distance, it displays many interesting properties, interpretations and relationships with other quantities such as the mutual information [Cover and Thomas, 2012] or the rate function in large deviations theory [Ellis, 2005]. It also is computationally convenient (as opposed to the Wasserstein distance for instance), and has been widely used used in machine learning [Ackley et al., 1985, Hinton et al., 2006]. Usually, this measure is used to compare the laws of two discrete or continuous random variables, but it can also be used to compare the laws of two stochastic processes in the path space, which is at the basis of this paper. This way of measuring the difference in law between two stochastic processes amounts in performing a change of probabilities thanks to Girsanov Theorem [Karatzas and Shreve, 1991], whose applications range from mathematical finance [Avellaneda et al., 1997] to simulation methods for rare events [Wainrib, 2013]. In the context of recurrent neural networks, we have already shown that the learning rule deriving from the minimization of relative entropy has interesting biological features since it combines two biologically plausible learning mechanisms [Galtier and Wainrib, 2013]

In this paper, we show how to train a noise-driven RNN to minimize its relative entropy with respect to a target process. The method consists two steps. First, the drift of the neural network is trained by minimizing its relative entropy with respect to the target (Section 3). Second, the noise matrix of the network is determined based on a conservation principle similarly to LIM (Section 4). We show how this approach extends the existing ESN and LIM theory in Section 5. Numerical approximations to the double well potential and to the El Niño phenomenon studied in climate research are presented in Section 6.

2 Model

We define here two mathematical objects that are, a priori, unrelated: a stochastic time series and an autonomous RNN made of two layers. The time series is assumed to be a sample path of an

underlying stochastic process which is the modelling target. The objective is to make the RNN approximate the target process.

The target time series $\mathbf{u}$ is assumed to be the discretization of an n-dimensional ergodic continuous process defined on the time interval $[0, T]$. The discretization step is chosen to be $dt = 1$ which corresponds to fixing the timescale. Imposing $T \in \mathbb{N}$, $\mathbf{u}$ can be seen as a matrix in $\mathbb{R}^{n\times T}$. For each $t \in \{1, .., T\}$, we use the notation $\mathbf{u}_t$ for the n-dimensional vector corresponding to the value of the continuous target time series at time t. Similarly, we write $\delta\mathbf{u}_t \overset{def}{=} \mathbf{u}_{t+1} - \mathbf{u}_t$ and $\delta\mathbf{u} \in \mathbb{R}^{n\times T}$ corresponding to the previous definition (with the convention that $\delta\mathbf{u}_T = 0$).

The two-layer neural network is defined as follows. The first layer, also called retina, has n neurons, as many as the target time series dimension. We take $\mathbf{v}_t^0 \in \mathbb{R}^n$ to be the activity of the retina at time t which will eventually approximate the target time series. The second layer, also called reservoir, has m neurons. Because each reservoir neuron does not directly correspond to a variable of the target, they are said to be hidden neurons. We denote the activity of the reservoir at time t by $\mathbf{v}_t^1 \in \mathbb{R}^m$. Each layer has a complete internal connectivity, recurrent connections, that is, all neurons within a layer are interconnected. The two layers are interconnected with feedforward, i.e. retina to reservoir, connections and feedback, i.e. reservoir to retina, connections, as shown in Figure 1.a. In this paper, according to a guiding principle in reservoir computing [Lukoševičius and Jaeger, 2009], the feedforward and reservoir matrices $\mathbf{W}_{10}$ and $\mathbf{W}_{11}$ are drawn randomly and remain unchanged. Only connections leading to retina neurons will be adapted. These are collected in $\mathbf{W} = (\mathbf{W}_{00}\ \mathbf{W}_{01}) \in \mathbb{R}^{n\times(n+m)}$.

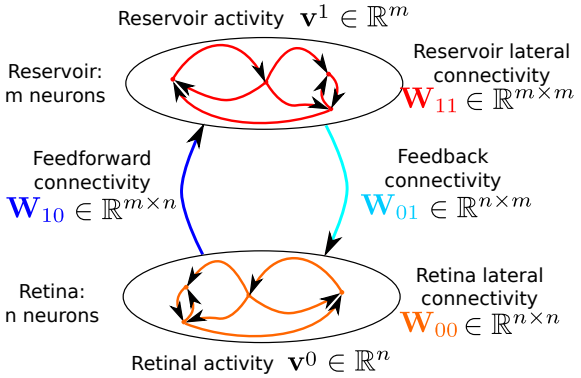


Figure 1: Structure and main notations of the neural network described in Section 2.

The activity of each layer is governed by the following differential law:

$$\begin{cases} d\mathbf{v}_t^0 = & \big(-\mathbf{v}_t^0 + \mathbf{W}_{00}\mathbf{v}_t^0 + \mathbf{W}_{01}\mathbf{v}_t^1\big)dt + \Sigma dB_t \\ d\mathbf{v}_t^1 = & \epsilon\big(-l\mathbf{v}_t^1 + s(\mathbf{W}_{10}\mathbf{v}_t^0 + \mathbf{W}_{11}\mathbf{v}_t^1)\big)dt \end{cases} \tag{1}$$

where $\epsilon, l \in \mathbb{R}_+$, s is a sigmoid function, e.g. $\tanh$, that is applied elementwise, i.e. $s(\mathbf{x})_i = s(\mathbf{x}_i)$, $\Sigma \in \mathbb{R}^{n\times n}$ is the noise matrix and B_t is an n-dimensional Brownian motion.

In order to unify LIM and ESNs, we submit this architecture to certain restrictions. In particular, we choose the first layer to be linear and we choose a $\tanh$ nonlinearity in the reservoir. Later, we will make a simple choice for a numerical differentiation scheme for the same reason.

3 Training to minimize relative entropy

This section explains the training of the connection matrices $\mathbf{W}_{00}$ and $\mathbf{W}_{01}$ so that the distance between the neural network and the target time series is minimized. In other words, the drift of the neural network will be designed to match that of the target stochastic process.

3.1 Relative entropy between target and retina

We now define a quantity measuring the dynamical distance between the target time series and the retinal activity. At first sight, these two mathematical objects have a different nature: the first is a time series and the second is a dynamical system. However, we assume that there exists a dynamical system (possibly very complicated and/or with hidden variables) which has generated the target time series. Thus, we want to compute the distance between this system and the neural network. A natural measure of similarity between stochastic processes is the relative entropy or Kullback-Leibler divergence. First, we show how to compute the relative entropy between two diffusion processes with the same diffusion. Second, we apply it to computing the "distance" between the neural network and the target time series.

3.1.1 Relative entropy between two diffusion processes sharing the same diffusion coefficient

We now introduce the computation of the relative entropy, or Kullback-Leibler divergence, between two n-dimensional diffusion processes $\mathbf{x}$ and $\mathbf{y}$ sharing the same diffusion coefficient $\Sigma \in \mathbb{R}^{n\times n}$. Let $\{\Omega, \mathcal{F}, P, \{\mathcal{F}_t\}_{0\leq t\leq T}\}$ be a probability space, equipped with the natural filtration of the standard Brownian motion. Consider two diffusion processes $(\mathbf{x}_t)$ and $(\mathbf{y}_t)$ in $\mathbb{R}^n$ under P, solutions of the following stochastic differential equation:

$$d\mathbf{x}_t = f(\mathbf{x}_t)dt + \Sigma dB_t \tag{2}$$

$$d\mathbf{y}_t = g(\mathbf{y}_t)dt + \Sigma dB_t \tag{3}$$

where B_t is a n-dimensional P-Brownian motion.

Defining and computing the relative entropy naturally follows from the application of the Girsanov theorem [Girsanov, 1960] described in chapter 3.5 of Karatzas and Shreve's textbook [Karatzas and Shreve, 1991][1]. It provides a stochastic change of variable which makes it possible to change the drift of a diffusion process provided the underlying measure of the Brownian motion is changed accordingly. Indeed, there exists a probability measure Q and $\tilde{B}_t$ a n-dimensional Q-brownian motion, such that the stochastic process $(\mathbf{x}_t)$ is also solution of

$$d\mathbf{x}_t = g(\mathbf{x}_t)dt + \Sigma d\tilde{B}_t \tag{4}$$

The probability measure has to be coherent with this change of drift, which is enforced through the Radon-Nikodym derivative of Q with respect to P

$$\frac{dQ}{dP} = \exp\left(\sum_{i=1}^{n}\int_0^T \left[\Sigma^{-1}\big(g(\mathbf{x}_t) - f(\mathbf{x}_t)\big)\right]_i dB_{t,i} - \frac{1}{2}\int_0^T \left\|\Sigma^{-1}\big(f(\mathbf{x}_t) - g(\mathbf{x}_t)\big)\right\|^2 dt\right) \tag{5}$$

where $\left[\Sigma^{-1}\big(g(\mathbf{x}_t) - f(\mathbf{x}_t)\big)\right]_i$ and $dB_{t,i}$ are the i^{th} components of $\Sigma^{-1}\big(g(\mathbf{x}_t) - f(\mathbf{x}_t)\big)$ and dB_t respectively.

For this quantity to be well-defined, the probability measure P has to be absolutely continuous with respect to Q. This is a consequence of the technical condition $\int_0^T \mathbb{E}[\|\Sigma^{-1}\big(f(\mathbf{x}_t) - g(\mathbf{x}_t)\big)\|^2]dt < \infty$.

Given that both processes $(\mathbf{x}_t)$ and $(\mathbf{y}_t)$ are written with the same drift (3), (4), it is natural to consider the relative entropy between the processes as the relative entropy between the measures P and Q which reads

$$H(\mathbf{y}|\mathbf{x}) \overset{def}{=} H(Q|P) \overset{def}{=} \mathbb{E}_Q\left[\ln\left(\frac{dQ}{dP}\right)\right] \tag{6}$$

Using (5) leads to

$$H(\mathbf{y}|\mathbf{x}) = \frac{1}{2}\mathbb{E}\left[\int_0^T \left\|\Sigma^{-1}\big(f(\mathbf{x}_t) - g(\mathbf{x}_t)\big)\right\|^2 dt\right] \tag{7}$$

[1] we apply the theorem 5.1 with, in their notations, $X_t = \Sigma^{-1}\big(g(\mathbf{x}_t) - f(\mathbf{x}_t)\big)$, $W_t = B_t$, $\tilde{W}_t = \tilde{B}_t$ and Z_T is the Radon-Nikodym derivative of Q with respect to P according to equation 5.4.

Strictly speaking, this quantity is not a distance since it is not symmetric. Yet it is always positive and zero only when the two drifts f and g are equal. This makes it a natural and useful measure of the similarity of two stochastic processes.

Note that although the drift g initially corresponds to the stochastic process $(\mathbf{y}_t)$, it is evaluated at the value $\mathbf{x}_t$. This is one of the main feature of relative entropy: it measures the difference of the drifts along the trajectory of one process.

It is crucial that the diffusion matrix Σ is the same in both equations (2) and (3). If it were not the case, the two measures P and Q would not be absolutely continuous and the Radon-Nikodym derivative, and a fortiori the relative entropy, would not be defined.

3.1.2 Application to our case

The main conceptual problem to apply the previous result is that, in practice, we do not know the continuous-time stochastic process which we assume has generated the discrete target time series. The time series is the only piece of information we have. Actually, many stochastic processes could have generated this discrete time series. We want to find the stochastic processes of the form (2), which are the more likely to have produced the time series. Let us call f the smooth function that defines the diffusion process (2) which was most likely to produce the target time series. Although we do not and will not know the explicit formula for f, we formally define the distance between the neural network (1) and the target time series as the relative entropy between the neural network and the diffusion process (2) with this particular f. We will eventually make this quantity computable.

We also need to bridge the gap between the discrete definition of the target time series and the continuous formulation of the relative entropy in (7). This can be done by discretizing equation (7) on the partition adapted to the definition of $\mathbf{u}$. Assuming that the sampling of $\mathbf{u}$ is fine enough, we can reasonably replace the integral by a discrete sum. At this step we can also use the ergodicity property of the target time series to drop the expectation in this equation. Therefore, a first tentative of definition of the relative entropy between the target time series and the neural network is

$$H = \frac{1}{2T} \sum_{t=0}^{T} \left\| \Sigma^{-1} \big(f(\mathbf{u}_t) - g(\mathbf{u}_t) \big) \right\|^2 + O(T^{-1/2}) \tag{8}$$

The term $O(T^{-1/2})$ accounts for the ergodicity approximation when the total time window T is not infinite.

However, in practice, it is not possible to have a direct access to $f(\mathbf{u})$. Estimating the drift $f(\mathbf{u})$ from the observation of the time-series $\mathbf{u}$ belongs to the class of problems called *numerical differentiation*. Since the seminal work of Savitzky and Golay [Savitzky and Golay, 1964], in which the signal is first approximated by moving polynomials before differentiation, a large number of numerical methods have been introduced, from finite difference methods to regularization or algebraic methods (see [Liu et al., 2011] and references therein). However, for simplicity and to rigorously relate to ESNs and LIM, we will simply approximate $f(\mathbf{u})$ by a *temporal difference* approximation which we called $\delta\mathbf{u}$. Recall

$$\delta\mathbf{u}_t = \mathbf{u}_{t+1} - \mathbf{u}_t = \left(\int_t^{t+1} f(\mathbf{x}_s) ds + \Sigma \xi_t \right) \tag{9}$$

where $\mathbf{x}_t$ is the realization of the process from which $\mathbf{u}$ was sampled and the ξ_t are i.i.d standard Gaussian random variables. We now assume that f is smooth enough and that the sampling of the stochastic process realization, which leads to defining the time series, is fine enough, so that the following approximation (of order 0) holds:

$$\int_t^{t+1} f(\mathbf{x}_s) ds \simeq f(\mathbf{u}_t) \tag{10}$$

We now form the difference $\|\Sigma^{-1}\big(f(\mathbf{u}_t) - g(\mathbf{u}_t)\big)\|^2$ and introduce $\Sigma^{-1}\delta\mathbf{u}_t$ in the computation. This leads to

$$\begin{aligned}
\|\Sigma^{-1}\big(f(\mathbf{u}_t) - g(\mathbf{u}_t)\big)\|^2 &\simeq \|\Sigma^{-1}\big(f(\mathbf{u}_t) - \delta\mathbf{u}_t\big)\|^2 \\
&+ 2\langle\Sigma^{-1}\big(f(\mathbf{u}_t) - \delta\mathbf{u}_t\big), \Sigma^{-1}\big(\delta\mathbf{u}_t - g(\mathbf{u}_t)\big)\rangle \\
&+ \|\Sigma^{-1}\big(\delta\mathbf{u}_t - g(\mathbf{u}_t)\big)\|^2 \\
&\simeq \|\xi_t\|^2 \\
&+ 2\Big\langle\xi_t, \Sigma^{-1}f(\mathbf{u}_t) + \xi_t\Big\rangle \\
&- 2\langle\xi_t, \Sigma^{-1}g(\mathbf{u}_t)\rangle \\
&+ \|\Sigma^{-1}\big(\delta\mathbf{u}_t - g(\mathbf{u}_t)\big)\|^2
\end{aligned}$$

Let us look at what becomes each of the four lines above when taking the empirical average (or equivalently the expectation by ergodicity):

- The first line becomes 1.
- In the second line, the first term $2\langle\xi_t, \Sigma^{-1}f(\mathbf{u}_t)\rangle$ has zero mean and thus vanishes. The second term becomes 2.
- The third line is centered and thus vanishes.
- Finally, the last term is the one we want to keep in the algorithm.

To summarize, it remains

$$H \simeq \frac{1}{2T}\Big(\sum_{t=0}^{T}\big\|\Sigma^{-1}\big(\delta\mathbf{u}_t - g(\mathbf{u}_t)\big)\big\|^2\Big) + O(T^{-1/2}) \tag{11}$$

One could also use a Taylor expansion for equation (10) which would add some corrective terms to (11). However, provided that the sampling of the target is fine enough, these terms can be neglected.

To be fully exhaustive, one should also look at the term $O(T^{-1/2})$ coming from the central limit theorem for ergodic convergence, and this term may also have some contribution which depends on the connectivity, in particular from the second term in line 3: $2\langle\xi_t, \Sigma^{-1}g(\mathbf{u}_t)\rangle$. It is zero-mean, so in the limit $T \to \infty$ it will disappear when we take the empirical average, but its variance is in fact $4\mathbb{E}[\Sigma^{-1}g(\mathbf{u})^2]$ which is not necessarily zero.

One last modification of the formal definition in equation (7) stems from its problematic dependence on the noise matrix Σ. Recall we intend to tune the noise to minimize the relative entropy. Given the definition (7) a trivial choice is to have an extremely strong noise: in this case the two stochastic processes are so random that they can be said to be identical. However, this is a situation which we would like to avoid: we want to find a reasonable noise matrix Σ. To correct this pathologic behavior, it seems natural to divide the formal definition (7) by the square of the operator norm of Σ^{-1} which we write $|||\Sigma^{-1}|||^2$ (or equivalently multiplying by the square of the smallest eigenvalue of Σ). The resulting quantity simply is proportional to the original definition, but without this problem. Note that this does not mean that we restrict our approach to unit norm noises matrices as is shown later.

This finally leads to a definition corresponding to equation (11) without the various error terms. Besides, we also want to take into account a regularization term. This leads to defining an analog to regularized relative entropy between the target time series $\mathbf{u}$ and our neural network (1) (respectively corresponding to $\mathbf{x}$ and $\mathbf{y}$ in (2) and (3)) as

$$H_{\mathbf{u}}(\mathbf{W}) \stackrel{def}{=} \frac{1}{2T}\sum_{t=0}^{T}\Big\|\mathbf{S}^{-1}\big(-\mathbf{u}_t + \mathbf{W}_{00}\mathbf{u}_t + \mathbf{W}_{01}\mathbf{u}_t^1 - \delta\mathbf{u}_t\big)\Big\|^2 + \frac{\alpha^2}{2}\|\mathbf{S}^{-1}\mathbf{W}\|^2 \tag{12}$$

where $\mathbf{S}^{-1} = \frac{\Sigma^{-1}}{|||\Sigma^{-1}|||}$ and $\mathbf{u}^1 \in \mathbb{R}^{m \times T}$ is the solution of

$$\delta \mathbf{u}_t^1 = \epsilon\big(- l\mathbf{u}_t^1 + s(\mathbf{W}_{10}\mathbf{u}_t + \mathbf{W}_{11}\mathbf{u}_t^1)\big)$$

governing the reservoir when it is fed by the target time series (as opposed to the retinal activity). The definition of $\delta\mathbf{u}^1 \in \mathbb{R}^{m \times T}$ is similar to that of $\delta\mathbf{u}$. In the following, we abusively call $H_{\mathbf{u}}(\mathbf{W})$ the relative entropy between the target time series and the network.

The second term in (12) is a regularization term which ensures decent generalization properties of the following algorithm. It does not change the qualitative meaning of the notion of relative entropy but penalizes the networks with high connection strength. The reason why it contains a multiplication by Σ^{-1} will become clear later. In short, it will make the system decoupled and we will be able to compute subsequently the connections first and the noise matrix second.

One can also understand the definition (12) without referring to the analogy with relative entropy, in the special case $\Sigma = I_d$ and $\alpha = 0$. Indeed, as shown in Figure 2, it corresponds to integrating the distance between the derivative of the target time series and the flow of the retinal activity along the trajectories of the target time series. Therefore, it is obvious that if this quantity is null then the vector field of the retinal activity will be tangent to the target time series derivative. Therefore, initializing the network with the value of the target time series at time $t = 0$, will lead the network to reproduce precisely the target time series.

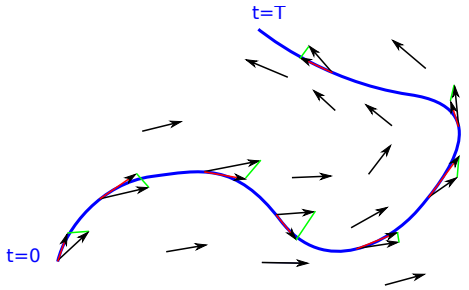


Figure 2: Illustration of the computation of the relative entropy in the case $\Sigma = I_d$. It corresponds to integrating all the green bars along the trajectory of the target time series. The solid blue line corresponds to the target time series. The red arrows correspond to the speed vectors of the target time series. The black arrows correspond to the vector field of the retinal neural network.

Observe that the relative entropy (12) (even more the rigorous definition (7)) are quite similar to the quantity usually minimized for prediction [Williams and Zipser, 1995, Pearlmutter, 1995, Bishop, 2006]. The main difference is that this is the integral of the distance between the *derivative* of the target time series and activity variable, instead of the mere distance between target and activity.

Note that the relative entropy (12) can easily be shown to be proportional to the negative of the log-likelihood of the target given the neural network. This close relationship shows that, in this case where the stochastic processes are diffusion processes, minimizing the relative entropy is rigorously equivalent to maximizing the log-likelihood.

3.2 Gradient of the relative entropy

Now, the idea is to compute the gradient of $H_{\mathbf{u}}(\mathbf{W})$ with respect to retinal and feedback connectivities. This will be useful because the gradient cancels out at the minimum of the relative entropy. This gives us a useful way to compute the connectivity $\mathbf{W}^* = (\mathbf{W}_{00}^* \ \mathbf{W}_{01}^*) \in \mathbb{R}^{n \times (n+m)}$ such that the network best approximates the target stochastic process. Because the relative entropy (12) is quadratic in $\mathbf{W}$, it is convex. Thus, it has a single critical point which is a global minimum.

For readability, we introduce the continuous nn-dimensional function defined on $[0, T]$ by

$$\zeta(\mathbf{W}) = \mathbf{W}_{00}\mathbf{u} + \mathbf{W}_{01}\mathbf{u}^1 - \delta\mathbf{u} - \mathbf{u} \tag{13}$$

such that

$$H_{\mathbf{u}}(\mathbf{W}) = \frac{1}{2}\sum_{t=0}^{T}\left\|\mathbf{S}^{-1}\zeta_t(\mathbf{W})\right\|^2 + \frac{\alpha^2}{2}\|\mathbf{S}^{-1}\mathbf{W}\|^2 \tag{14}$$

The gradient will be computed in an operator framework, i.e. as the differential of the relative entropy. Observe that the differential is a linear operator such that

$$\begin{aligned} d_{\mathbf{W}}H_{\mathbf{u}}(\mathbf{J}) &= \frac{1}{2}d_{\mathbf{W}}\Big(\sum_{t=0}^{T}\langle\mathbf{S}^{-1}\zeta_t,\mathbf{S}^{-1}\zeta_t\rangle + \alpha^2\langle\mathbf{S}^{-1}\mathbf{W},\mathbf{S}^{-1}\mathbf{W}\rangle\Big)(\mathbf{J}) \\ &= \sum_{t=0}^{T}\frac{1}{2}d_{\mathbf{W}}\big\langle\mathbf{S}^{-1}\zeta_t,\mathbf{S}^{-1}\zeta_t\big\rangle(\mathbf{J}) + \alpha^2\langle\mathbf{S}^{-1}\mathbf{W},\mathbf{S}^{-1}\mathbf{J}\rangle \\ &= \sum_{t=0}^{T}\langle\mathbf{S}^{-1}\zeta_t(\mathbf{W}),\mathbf{S}^{-1}d_{\mathbf{W}}\zeta_t(\mathbf{J})\rangle + \alpha^2\langle\mathbf{S}^{-1}\mathbf{W},\mathbf{S}^{-1}\mathbf{J}\rangle \\ &= \sum_{t=0}^{T}\langle\big(\mathbf{S}\mathbf{S}'\big)^{-1}\zeta_t(\mathbf{W}),d_{\mathbf{W}}\zeta_t(\mathbf{J})\rangle + \alpha^2\langle(\mathbf{S}\mathbf{S}')^{-1}\mathbf{W},\mathbf{J}\rangle \end{aligned} \tag{15}$$

Because $\zeta(\mathbf{W})$ is affine in $\mathbf{W}_{00}$ and $\mathbf{W}_{01}$, it appears that

$$d_{\mathbf{W}_{00}}\zeta(\mathbf{J}) = \mathbf{J}\mathbf{u} \quad \text{and} \quad d_{\mathbf{W}_{01}}\zeta(\mathbf{J}) = \mathbf{J}\mathbf{u}^1 \tag{16}$$

In both cases, we have a differential of the form $d_{\mathbf{W}}\zeta(\mathbf{J}) = \mathbf{J}\mathbf{b}$, with $\mathbf{b} = \mathbf{u}$ (resp. $\mathbf{b} = \mathbf{u}^1$) is a matrix of $\mathbb{R}^{n\times T}$ (resp. $\mathbb{R}^{m\times T}$).
Observe that $\nabla_{\mathbf{W}}H_{\mathbf{u}ij} = d_{\mathbf{W}}H_{\mathbf{u}}(\mathbf{E}^{ij})$ where $\mathbf{E}^{ij}$ is the canonical matrix made of zeros except at position ij where it is one. Then,

$$\begin{aligned} d_{\mathbf{W}}H_{\mathbf{u}}(\mathbf{E}^{ij}) &= \sum_{t=0}^{T}\langle(\mathbf{S}\mathbf{S}')^{-1}\zeta_t(\mathbf{W}),\mathbf{E}^{ij}\mathbf{b}_t\rangle + \alpha^2\langle(\mathbf{S}\mathbf{S}')^{-1}\mathbf{W},\mathbf{E}^{ij}\rangle \\ &= \sum_{t=0}^{T}\big\{(\mathbf{S}\mathbf{S}')^{-1}\zeta_t(\mathbf{W})\big\}_i\mathbf{b}_{jt} + \alpha^2\{(\mathbf{S}\mathbf{S}')^{-1}\mathbf{W}\}_{ij} \\ &= \big\{(\mathbf{S}\mathbf{S}')^{-1}\zeta(\mathbf{W})\mathbf{b}'\big\}_{ij} + \alpha^2\{(\mathbf{S}\mathbf{S}')^{-1}\mathbf{W}\}_{ij} \end{aligned}$$

which leads to

$$\nabla_{\mathbf{W}}H_{\mathbf{u}} = (\mathbf{S}\mathbf{S}')^{-1}\zeta(\mathbf{W})\mathbf{b}' + \alpha^2(\mathbf{S}\mathbf{S}')^{-1}\mathbf{W}$$

Using the definition of ζ leads to

$$(\mathbf{S}\mathbf{S}')\nabla_{\mathbf{W}}H_{\mathbf{u}} = -\Big[(\delta\mathbf{u}+\mathbf{u})\mathbf{u}' \quad (\delta\mathbf{u}+\mathbf{u})\mathbf{u}^{1\prime}\Big] + (\mathbf{W}_{00}\,\mathbf{W}_{01})\left(\alpha^2 I_d + \begin{bmatrix} \mathbf{u}\mathbf{u}' & \mathbf{u}\mathbf{u}^{1\prime} \\ \mathbf{u}^1\mathbf{u}' & \mathbf{u}^1\mathbf{u}^{1\prime} \end{bmatrix}\right) \tag{17}$$

The global minimum can be computed as $\nabla_{\mathbf{W}^*}H_{\mathbf{u}} = 0$. Since we have assumed that $\mathbf{S}$ is a full rank matrix, this leads to the following formula

$$\begin{aligned} \mathbf{W}^* &= \Big[(\delta\mathbf{u}+\mathbf{u})\mathbf{u}' \quad (\delta\mathbf{u}+\mathbf{u})\mathbf{u}^{1\prime}\Big]\left(\alpha^2 I_d + \begin{bmatrix} \mathbf{u}\mathbf{u}' & \mathbf{u}\mathbf{u}^{1\prime} \\ \mathbf{u}^1\mathbf{u}' & \mathbf{u}^1\mathbf{u}^{1\prime} \end{bmatrix}\right)^{-1} \\ &= (\delta\mathbf{u}+\mathbf{u})\begin{pmatrix}\mathbf{u}\\ \mathbf{u}^1\end{pmatrix}'\left(\alpha^2 I_d + \begin{pmatrix}\mathbf{u}\\ \mathbf{u}^1\end{pmatrix}\begin{pmatrix}\mathbf{u}\\ \mathbf{u}^1\end{pmatrix}'\right)^{-1} \end{aligned} \tag{18}$$

It is interesting to observe that the solution does not depend on the noise matrix Σ. It is this property which makes it possible for the problem to be decoupled in two parts: (i) computation of $\mathbf{W}^*$ and (ii) computation of Σ.

4 Computing the noise with a conservation principle

This section is devoted to computing the noise matrix Σ such that the neural network matches the statistics of the target time series. Obviously the choice of a simple additive noise (i.e. Σ does not depend on $\mathbf{v}$ nor t) in system (1), restricts the class of target system the neural network can approximate accurately. Although the match will not be perfect, we will see the method provides a reasonable and (more crucially) coherent noisy neural network approximating the statistics of the target.

At first sight, it may seem appropriate to choose Σ so that the covariance of the retinal activity matches that of the target. However, the non-linearity in the reservoir makes it seemingly impossible to compute analytically the covariance of system (1). Therefore, we have not been able to use this idea to fix Σ.

Another method consists in using a generalized fluctuation dissipation relation. Penland and Matrosova [Penland and Matrosova, 1994] have detailed a method to use a conservation principle to link the correlation of the activity, the flow of the retina and the matrix $\Sigma\Sigma'$. Following the lines of their derivation, we generalize their approach to the case of reservoirs.

The generalized fluctuation dissipation relation is based on the Fokker-Planck equation [Risken, 1996] of the neural network (1). Any stochastic differential system can be described equivalently by a sample path dynamics governed by (1) or a Fokker-Planck equation which governs the evolution of the probability density function. It corresponds to the Eulerian description of the original stochastic differential equation. It can intuitively be understood as a balance of how much goes in and out of a small box centered on $\mathbf{v}$, taking into account both a drift and a diffusion mechanism. In our case, it takes the form of the following partial differential equation.

$$\frac{\partial p(\mathbf{v}^0,\mathbf{v}^1,t)}{\partial t} = -\text{div}\left[\begin{pmatrix} -\mathbf{v}^0 + \mathbf{W}_{00}\mathbf{v}^0 + \mathbf{W}_{01}\mathbf{v}^1 \\ \epsilon\big(-l\mathbf{v}^1 + s(\mathbf{W}_{10}\mathbf{v}^0 + \mathbf{W}_{11}\mathbf{v}^1)\big)\end{pmatrix} p(\mathbf{v}^0,\mathbf{v}^1,t)\right] + \frac{1}{2}\Delta\left[\begin{pmatrix}\Sigma\Sigma' & 0\\ 0 & 0\end{pmatrix} p(\mathbf{v}^0,\mathbf{v}^1,t)\right] \quad (19)$$

where div is the divergence operator, i.e. $\text{div}(\mathbf{x}) = \sum_i \frac{\partial \mathbf{x}_i}{\partial \mathbf{v}_i}$ which corresponds to the drift, and Δ is the Laplacian operator, i.e. $\Delta\mathbf{J} = \sum_{i,j} \frac{\partial^2 \mathbf{J}_{ij}}{\partial \mathbf{v}_i \partial \mathbf{v}_j}$, which corresponds to the diffusion. Note that this Fokker-Planck equation is independent of the underlying choice between Itô or Stratonovich noise in the initial system (1), because we assumed Σ does not depend on the activity of the network.

A conservation principle about the moments of the stochastic process $\mathbf{v}$ can easily be derived from equation (19). Indeed, multiply it by $\{\mathbf{v}_t^0\}_p\{\mathbf{v}_t^0\}_q$ (the components p and q of the activity $\mathbf{v}_t^0$) and integrate over the entire domain. We can use the integration by part formula several times to get

$$\frac{\partial \mathbb{E}[\{\mathbf{v}_t^0\}_p\{\mathbf{v}_t^0\}_q]}{\partial t} = \mathbb{E}\left[\big(-\{\mathbf{v}_t^0\}_p + \{\mathbf{W}_{00}\mathbf{v}_t^0\}_p + \{\mathbf{W}_{01}\mathbf{v}_t^1\}_p\big)\{\mathbf{v}_t^0\}_q\right] + \mathbb{E}\left[\big(-\{\mathbf{v}_t^0\}_q + \{\mathbf{W}_{00}\mathbf{v}_t^0\}_q + \{\mathbf{W}_{01}\mathbf{v}_t^1\}_q\big)\{\mathbf{v}_t^0\}_p\right] + \Sigma\Sigma' \quad (20)$$

Given that the target is ergodic, it is natural to assume that its approximation also has this property. Therefore, it is legitimate to replace the expectations by time integrals over $[0,T]$ divided by T in the previous equation. In a matrix formalism, this reads

$$\frac{\mathbf{v}_T^0\mathbf{v}_T^{0\prime} - \mathbf{v}_0^0\mathbf{v}_0^{0\prime}}{T} = \big(-\mathbf{v}^0 + \mathbf{W}_{00}\mathbf{v}^0 + \mathbf{W}_{01}\mathbf{v}^1\big)\mathbf{v}^{0\prime} + \mathbf{v}^0\big(-\mathbf{v}^0 + \mathbf{W}_{00}\mathbf{v}^0 + \mathbf{W}_{01}\mathbf{v}^1\big)' + \Sigma\Sigma'$$

A careful inspection of the terms shows that the left hand side is negligible when T is large enough (which will always be the case in practice). Thus, we will drop this term for simplicity (although it would not pose any problem to take it into account).

Notice the correlations terms $\mathbf{v}^0\mathbf{v}^{0\prime}$ and $\mathbf{v}^0\mathbf{v}^{1\prime}$ in the previous equation. Recall our initial wish to choose Σ so that both neural network and target second order moments are matched. Although

we could not directly implement this wish, we are now able to replace the network correlation terms by the observed moments of the target in the present formulation. This ansatz leads to the following generalized fluctuation dissipation relation [Penland and Matrosova, 1994]:

$$\Sigma\Sigma' = \big(I_d - \mathbf{W}_{00}\big)\frac{\mathbf{u}\mathbf{u}'}{T} - \mathbf{W}_{01}\frac{\mathbf{u}^1\mathbf{u}'}{T} + \frac{\mathbf{u}\mathbf{u}'}{T}\big(I_d - \mathbf{W}_{00}\big) - \frac{\mathbf{u}\mathbf{u}^{1\prime}}{T}\mathbf{W}_{01} \tag{21}$$

This equation can be seen as a coherency requirement between drift and diffusion of the network and second order moments of the target. Fortunately, the derivation of the drift leads to an explicit equation (18) independent of the matrix Σ. Therefore, the previous equation can be used to characterize Σ based on the knowledge of $\mathbf{W}$. Given that the square root of matrix is not injective, there are several choices for the matrix Σ. They all correspond to an ambiguity on the sign of its eigenvalues. We arbitrarily pick one of them and have thus found a coherent noise matrix.

5 Comparison with existing methods

Our approach takes selected features of two existing methods for approximation, ESN and LIM, and unifies them in a mathematical framework. This section is devoted to clarifying the links with these two methods.

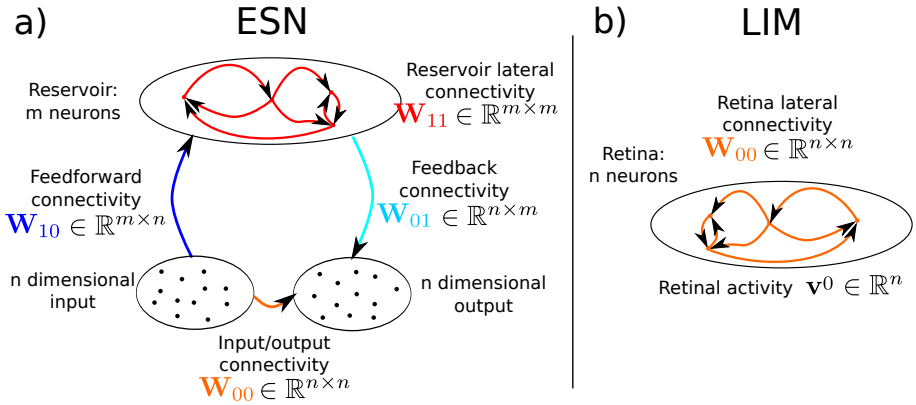


Figure 3: Architecture of related existing models. To be compared with Fig.1. a) Architecture of an ESN (see section 5.2): the inputs and outputs are different layers. However, when the network is set to the prediction mode the value of the inputs is copied from the outputs, which corresponds to closing up the loop as in Fig.1. b) Architecture of LIM (see section 5.1): there is no reservoir and only a retina.

5.1 Adding nonlinearities to Linear Inverse Modeling

The proposed method rigorously extends LIM by adding non-linearities to the model dynamics. Linear inverse modeling consists in designing a multidimensional linear stochastic (also called Ornstein-Uhlenbeck) process which reproduces a target multivariate time series [Penland and Sardeshmukh, 1995, Penland, 1996]. Naturally, the dimension of the approximating process is identical to the dimension of the target. A pervasive idea in machine learning is to consider additional (often called hidden) variables which will help the reconstruction of the target multivariate time series. In the framework presented in this paper, they correspond to the neurons in the reservoir, while the linear dynamical system analogous to LIM corresponds to the neurons in the retina. Due to the non-linearity of the dynamics of these additional variables or neurons, the present framework is as a non-linear extension of LIM. Actually, it turns out to be a surprisingly simple extension since the same formula can be used to calculate the linear matrix in LIM and the rectangular matrix which combines the additional variables to improve the retina's predictions.

More precisely, LIM consists in finding the matrices $\mathbf{B}, \mathbf{Q} \in \mathbb{R}^{n \times n}$ such that the following dynamical system reproduces the target time series $\mathbf{u}$.

$$\dot{\mathbf{v}^0}_t = \mathbf{B}\mathbf{v}^0_t + \sqrt{\mathbf{Q}}\dot{B}_t$$

where $\dot{B}_t$ is a white noise and $\sqrt{}$ is the matrix square root (i.e. $\mathbf{Q}$ is the covariance of $\sqrt{\mathbf{Q}}\dot{B}_t$). Note that we have intentionally used the same variable $\mathbf{v}^0$ for the LIM and the activity in the retina.

Based on the explicit expression of the Green function of a linear system [Risken, 1996], $\mathbf{B}$ can be caracterized as follow

$$\mathbf{B} = \frac{1}{\tau} \ln(\mathbf{u}_{.+\tau}\mathbf{u}'(\mathbf{u}\mathbf{u}')^{-1}) \tag{22}$$

where $\ln$ is the logarithm for matrices and $\mathbf{u}_{.+\tau}$ is the matrixes whose column number t is $\mathbf{u}_{t+\tau}$ and τ is an integer usually equal to 1 (depending on the intrinsic timescales of the target). Assuming $\tau = 1$ in the following, $\mathbf{B}$ can be written:

$$\mathbf{B} = \ln(I_d + (\mathbf{u}_{.+1} - \mathbf{u})\mathbf{u}'(\mathbf{u}\mathbf{u}')^{-1}) = \sum_{k=1}^{+\infty} \frac{(-1)^{k+1}}{k} \left[(\mathbf{u}_{.+1} - \mathbf{u})\mathbf{u}'(\mathbf{u}\mathbf{u}')^{-1}\right]^k$$

Assuming appropriate sampling of $\mathbf{u}$ such that $\|\delta\mathbf{u}\| \ll 1$, we can reasonably truncate $\mathbf{B}$ at first order. This leads to the following approximation: $\mathbf{B} \simeq \delta\mathbf{u}\mathbf{u}'(\mathbf{u}\mathbf{u}')^{-1}$.

To see the link with ESNsto, we must consider the case $m = 0$ in eq.1, so that $\mathbf{W} = I_d + \mathbf{B}$. From the above approximation, it follows $\mathbf{W} \simeq I_d + \delta\mathbf{u}\mathbf{u}'(\mathbf{u}\mathbf{u}')^{-1} \simeq (\delta\mathbf{u} + \mathbf{u})\mathbf{u}'(\mathbf{u}\mathbf{u}')^{-1}$, which corresponds to $\mathbf{W}^*$ in (18) with $m = 0$ and $\alpha = 0$. Given the definition of both systems, it is clear that the two methods are identical in this restricted case. However, when the sampling is not fine enough, equation (22) may lead to solutions than differ from the log-free equation (18). One then could also imagine a variation of ESNs using a matrix log when dealing with badly sampled data, but this is beyond the scope of this paper.

Concerning the treatment of noise, we can observe that it is strictly the same with or without additional reservoir neurons.

5.2 Adding noise to Echo States Networks

The proposed algorithm rigorously generalizes ESNs to a stochastic framework. Indeed, the formula for the connectivity in equation (18) is identical to the solution given by applying the classical deterministic ESN method [Jaeger and Haas, 2004, Lukoševičius and Jaeger, 2009].

To explain further the equivalence between this formalism and classical ESNs, we now introduce the ESN formalism in its original form as summarized in Figure 3.a. Let us setup an ESN of reservoir size m as a one time step predictor of the input $\mathbf{u}$, where the input and the output have, naturally, the same dimension n, and the model includes direct connections from the input to the output. In fact, the initial setup of echo state network makes a distinction between input and output, see Figure 3.a; whereas the model introduced in section 2 only has a retina, see Fig.1. However, ESNs can also be run in a generative mode, where the current output becomes the next input, closing the loop between the two. This closed loop system is precisely the same as the two layer network of Section 2, where the joined input/output nodes become the retina with activations $\mathbf{v}^0$, and the reservoir remains with activations $\mathbf{v}^1$. As summarized in Figure 3.a, connections from the input to the reservoir correspond to $\mathbf{W}_{10}$, internal reservoir connections to $\mathbf{W}_{11}$, output connections from the reservoir to $\mathbf{W}_{01}$, and connections from input directly to the output after closing the loop become the recurrent connections in the retina $\mathbf{W}_{00}$.

Classical ESNs are discrete-time systems, as opposed to our continuous-time approach. Yet the two methods are closely linked: ESNs correspond to a time-discretized version of (1).

$$\left\{ \begin{array}{rcl} \mathbf{v}^0_{t+1} = & \mathbf{W}_{00}\mathbf{v}^0_t + \mathbf{W}_{01}\mathbf{v}^1_t \\ \mathbf{v}^1_{t+1} = & s(\mathbf{W}_{10}\mathbf{v}^0_t + \mathbf{W}_{11}\mathbf{v}^1_t) \end{array} \right. , \tag{23}$$

where (1) is discretized using Euler's approximation and the discretization step is taken to be equal to 1 and $\epsilon = l = 1$.

Training such a setup to minimize a squared error on the input prediction $\sum_{t=0}^{T} \left(\mathbf{u}_{t+1} - \mathbf{v}_t^0\right)^2$ precisely corresponds to learning connections $\mathbf{W}_{00}$ and $\mathbf{W}_{01}$ according to the equation (18). Indeed, observe that $\{\delta\mathbf{u}+\mathbf{u}\}_t = \mathbf{u}_{t+1}$ is the prediction of the input (which corresponds to the target signal in this setup). Taking $\mathbf{u}$ as input and $\mathbf{u}^1$ as teacher-forced reservoir activations, the equation (18) turns out to be the ridge regression equation generally used in [Lukoševičius and Jaeger, 2009].

The treatment of noise proposed in this paper is a new contribution to the ESN theory. In that sense, this paper consists in an extension of ESNs as time series approximators to stochastic ESNs as stochastic process approximators.

6 Numerical simulations

This section shows two examples of application of the proposed ESNsto algorithm described in algorithm 1.

Algorithm 1 ESNsto learning

Initialization:
Components of $\mathbf{W}_{01}$ and $\mathbf{W}_{11}$ are drawn randomly following a normal law. The two matrices are then rescaled get desired spectral radii.
$\mathbf{u}_0^1, \mathbf{v}_0^0, \mathbf{v}_0^1 \leftarrow 0$
# Collect reservoir states:
while $t < T$: **do**
$\mathbf{u}_{t+1}^1 = (1 - \epsilon l)\mathbf{u}_t^1 + \epsilon l \; s(\mathbf{W}_{10}\mathbf{u}_t + \mathbf{W}_{11}\mathbf{u}_t^1)$
end while
# Learn connections:

$$(\mathbf{W}_{00}, \mathbf{W}_{01}) \leftarrow (\delta\mathbf{u} + \mathbf{u}) \begin{pmatrix} \mathbf{u} \\ \mathbf{u}^1 \end{pmatrix}' \left(\alpha^2 I_d + \begin{pmatrix} \mathbf{u} \\ \mathbf{u}^1 \end{pmatrix} \begin{pmatrix} \mathbf{u} \\ \mathbf{u}^1 \end{pmatrix}' \right)^{-1}$$

Compute noise connections:

$$\Sigma = \sqrt{\left(I_d - \mathbf{W}_{00}\right)\frac{\mathbf{u}\mathbf{u}'}{T} - \mathbf{W}_{01}\frac{\mathbf{u}^1\mathbf{u}'}{T} + \frac{\mathbf{u}\mathbf{u}'}{T}\left(I_d - \mathbf{W}_{00}\right) - \frac{\mathbf{u}\mathbf{u}^{1\prime}}{T}\mathbf{W}_{01}}$$

# Simulate ESNsto:
while True **do**

$$\begin{aligned} \mathbf{v}_{t+1}^0 &= \mathbf{W}_{00}\mathbf{v}_t^0 + \mathbf{W}_{01}\mathbf{v}_t^1 + \Sigma \,\text{randn}(n) \\ \mathbf{v}_{t+1}^1 &= (1 - \epsilon l\mathbf{v}_t^1) + \epsilon s(\mathbf{W}_{10}\mathbf{v}_t^0 + \mathbf{W}_{11}\mathbf{v}_t^1) \end{aligned}$$

end while

It is important to realize that the goal here is not to approximate or predict a time series, but rather to approximate a stochastic process. Because a single stochastic process can have different realizations, an approximation of such a process should not aim at reproducing a given path. In this sense, we are not dealing with classical prediction tasks and we should exlusively focus on building a system that reproduces the law of the target stochastic process. As a consequence, we can not compare ESNsto with classical ESN since they do not approximate the same mathematical objects.

We are going to compare the performances of LIM with that of ESNsto. LIM belongs to several of the different classes of approximators that we mentioned in the introduction: it is a Gaussian process, a multivariate autoregressive process of order one and an Ornstein-Uhlenbeck process. Besides it can be easily compared to ESNsto, since the latter generalizes the former and LIM simply is an ESNsto with 0 neurons in the reservoir. Establishing a complete benchmark of the different methods for stochastic processes approximation is beyond the scope of this paper. However, we point out the low computational complexity of learning, which is independent of the length of the time series and mainly governed by the inversion of a positive semi definite square matrix of size n. This is lower than the complexity of Hidden Markov Models or Gaussian Processes for long time series.

To compute the relative entropy we have used a classical cross-validation framework. This

means that we have divided the target time series in k blocks. Then for each block, we have computed the connectivity and noise matrices on the remaining $k-1$ blocks, and evaluated the value of the relative entropy on the selected block. This provides a robust way to prevent over-fitting.

This numerical section is only a proof of concept. The (hyper)parameters of the networks (such as the spectral radius of $\mathbf{W}_{11}$) have been coarsely tuned, although they could significantly improve the approximations if they were set carefully. The main reason for our negligence is that we want to show that an off-the-shelf ESNsto model is better than LIM and does not require deep knowledge or large effort of neural networks tuning.

The first example we consider is a widely considered toy model: the target is generated by a noisy particle living in a double well. The second example, devoted to climate modeling, will show how to approximate the El Niño phenomenon in the tropical Pacific ocean.

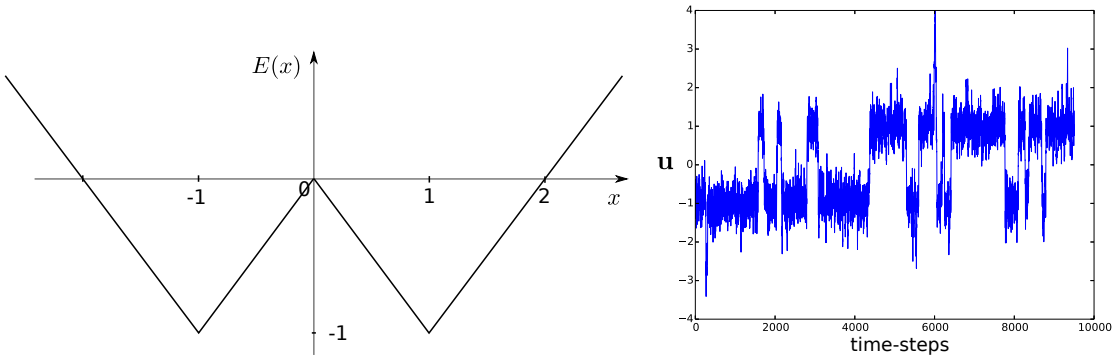
6.1 The noisy double-well

The double well example explores a basic form of non-linearity. It illustrates the significant improvement brought by reservoir neurons in dealing with non-linearities.

We consider a synthetic example where the data are generated as the solution of the stochastic differential equation corresponding to a particle in an energy landscape made of two different wells, as shown in Fig.4(a). More precisely, the target is a one-dimensional process described by

$$d\mathbf{u} = -\nabla_{\mathbf{u}} E dt + \sigma dB_t \tag{24}$$

where $\nabla_{\mathbf{u}} E = \begin{cases} 1 & \text{if } -1 < \mathbf{u} < 0 \text{ or } \mathbf{u} > 1 \\ -1 & \text{else} \end{cases}$ is the gradient of the function described in Fig.4(a) at point $\mathbf{u}$. The typical behavior of such a system is illustrated in Fig.4(b). Roughly speaking, the particle jumps from one well to the other after random durations. Informally, each well can be said to be an attractor.



(a) Double well energy landscape.

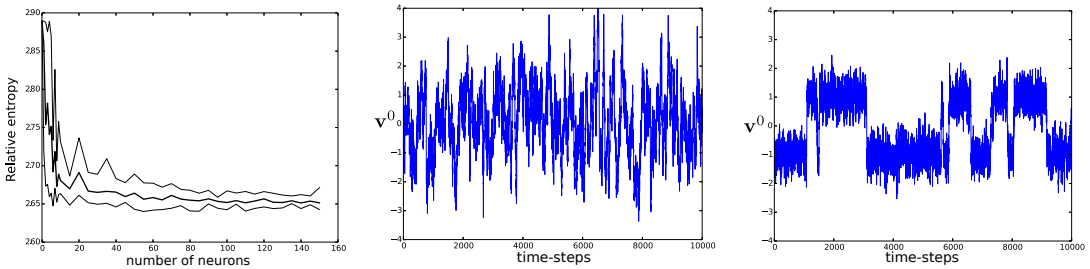
(b) Trajectory of process (24).

Figure 4: Illustrations of the double well process (24). (left) Energy landscape. (right) Trajectory of the process. The simulation was done using Euler-Maruyam algorithm with $\sigma = 0.7$, $dt = 0.1$ during 10000 time steps. These parameters are kept constant for the next simulations.

We now compare the approximations of this time series based on LIM and the ESNsto. We see in Fig.5(a) that increasing the number of neurons in the reservoir improves the relative entropy defined by (12). With no neurons in the reservoir, which corresponds to LIM, the relative entropy is approximately 2.9. After a sharp decrease for the first dozens of additional neurons, the relative entropy slowly decreases when the number of neurons increases to finally reach a value close to 2.65 for $m = 150$ neurons. Note that the simple numerical differentiation method that we have used imposes a lower bound on the relative entropy shown here. Indeed, it is easy to observe that the relative entropy between system (24) and itself using definition (12) is 2.45. We believe that the gap between the ESNsto and the optimal value will decrease with additional neurons

in the network, but may not vanish due to the over-fitting issue mentioned later. This suggests that appropriately choosing the regularization parameter is crucial for optimal accuracy. We can also observe that the variance of the relative entropy value, corresponding to different random realizations for the connections in the reservoir $\mathbf{W}_{11}$ and $\mathbf{W}_{10}$, decreases with the number of neurons: due to better averaging, large reservoirs are less dependent on the realization defining their weights.

Running the LIM and ESNsto networks post-learning shows different qualitative behaviors as displayed in Fig. 5(b) and 5(c). As opposed to LIM, the ESNsto reproduces patterns of noise-induced jumps between two attractors.



(a) Relative entropy vs number of neurons.

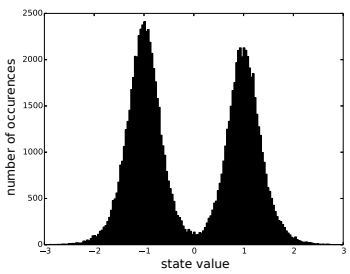
(b) LIM trajectory.

(c) ESNsto trajectory.

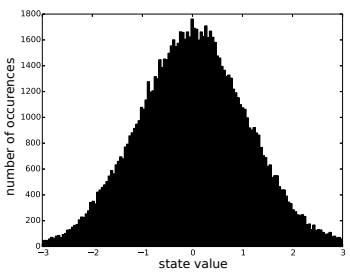
Figure 5: (a) Relative entropy as a function of the number of neurons in the reservoir. For any number of neurons, we launched 10 simulations, to take into account the variability of reservoir connections, and measured the relative entropy according to equation (12). The top (resp. bottom) curve is the maximum (resp. minimum) value of the entropy among the different trials. The middle curve is the mean. The relative entropy roughly decreases from 2.9 to 2.65 which is enough to qualitatively change the behavior of the neural networks as shown in the left and middle picture. (b) Result of the LIM simulation (i.e. $m = 0$) learned from the data in Fig.4(b). (c) Result of the ESNsto simulation with $m = 100$ neurons. Parameters used: $\epsilon = 1$, $l = 1$, $s(x) = \tanh(2x)$, $\mathbf{W}_{11}$ is drawn according to a normal law and rescaled so that $\|\mathbf{W}_{11}\| = 1$ (operator norm), $\mathbf{W}_{10}$ is drawn uniformly in $[-1.1]$, $\alpha = 1$. The network simulations and learning were done during $T = 10^5$ time steps. Note that there is a washout time interval of 500 at the beginning of the learning (and reconstruction) in order to remove transient effects.

Significant qualitative improvements made possible by having neurons in the reservoir can also be seen by empirically measuring some statistical quantities of the target, LIM and ESNsto runs, as shown in Fig.6. The first three figures 6(a), 6(b) and 6(c), show that LIM is failing to reproduce the bimodal distribution of the target corresponding to the two attractors: LIM has a unimodal Gaussian-like distribution, whereas ESNsto is able to reproduce the bimodality thanks to the non-linearities in the reservoir.

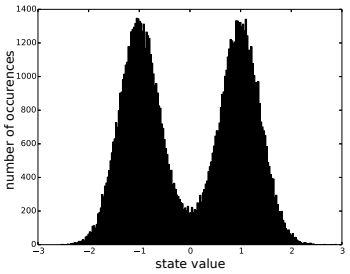
In Fig. 6(d), it is shown that the distribution of the times spent in each attractors between two jumps is irrelevant for LIM whereas it is similar between target and ESNsto. This is also reflected in the transition rates which is approximately 0.0019 for the data, 0.0016 for the ESNsto and 0.0075 for the LIM. However, it is to be noticed that an increase in the number of neurons beyond 150 neurons leads to a decrease of the transition rate (not shown). This underlines a drawback of the method for non-ergodic time series which exhibit significant noisy flucutuations: the network tries to put in the connectivity as much variability as possible; the noise term is simply taking care of the left-overs. Therefore, the noise induced transitions in the target are not only modeled by the diffusion term in the neural network but also by the drift. This effect will vanish if the learning time series has enough ergodicity so that the noisy behavior is averaged out when computing the drift. When there is a limited amount of time steps available, a better numerical differentiation scheme may improve the approximation accuracy since it would filter out noise before asking the drift to approximate it.



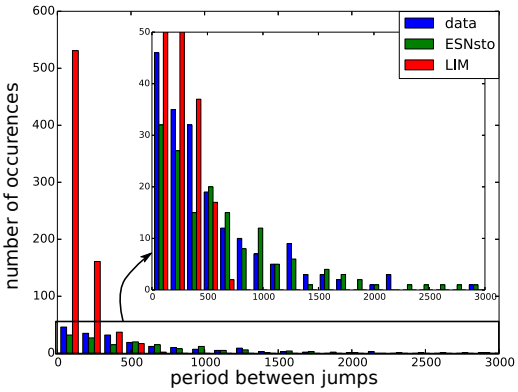
(a) State distribution for the data.



(b) State distribution for the LIM.



(c) State distribution for the ESNsto.



(d) Distributions of the durations between jumps.

Figure 6: Distribution of the states of the data (a), LIM (b) and ESNsto (c). (d): Compared histograms of the duration spent in a well between two jumps. The parameters used are the same as for Fig.5.

6.2 El Niño phenomenon

In this section, we focus on approximating the geophysical process El Niño. It corresponds to a large warming of Sea Surface Temperature (SST) in the eastern equatorial Pacific, occurring irregularly every 2 to 7 years, and having a broad impact of the global climate [Trenberth, 1997, Deser et al., 2010]. As many other geophysical processes, its dynamics evolves on different interacting timescales. It is in particular strongly linked to atmospheric processes evolving at shorter time scales with a nonlinear behaviour. Penland showed that the evolution of 3-month running mean SST anomalies in the tropical Indo-Pacific, and thus of El Niño, are well approximated by a LIM, where the rapidly varying nonlinear processes are parameterized as a stochastic forcing of the slower system [Penland, 1996]. One commonly used index of the El Niño phenomenon is the Niño 3.4 index (N34 index), defined as the averaged of SST anomalies between 5°S- 5°N and 170°W-120°W. Using the definition of [Trenberth, 1997], an El Niño event is said to occur when the N34 index, smoothed with a 5-month running mean, exceed 0.4°C for 6 months or more. Fig. 7 shows the N34 index (top) and the regression of SST anomalies onto this index (bottom), indicating the warming in the eastern equatorial Pacific associated with a positive N34 index.

The target time series considered here are the N34 index, smoothed with a 3-month running mean, and the 10 first principal components (PCs) of the empirical orthogonal function of 3-month running mean SST anomalies in the IndoPacific region (30°S-30°N, 40°E-70°W). These 10 PCs represent 80% of the total variance of monthly IndoPacific SST anomalies, and they are used instead of considering directly the SST anomalies at each grid point in the IndoPacific to reduce the dimensionality of the system. The data come from the HadISST1 SST dataset [Rayner et al., 2003], constructed from in situ SST observations and satellite derived estimates and available from 1870. We used data from 1870 to 2011. Our target times series contain thus 1704 time steps (corresponding to 1704 months or 142 years) and a washout period of 240 time steps is used at the beginning of the learning to remove transient effects.

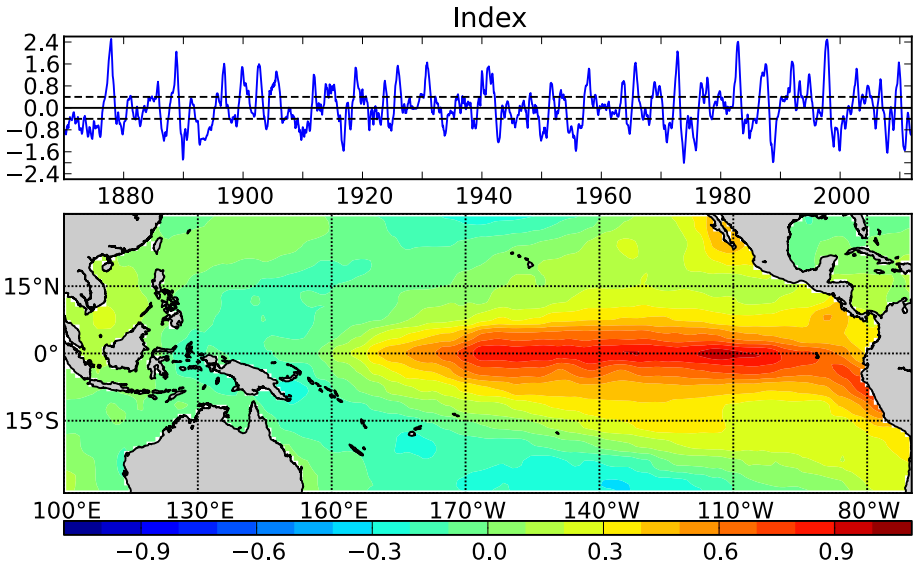
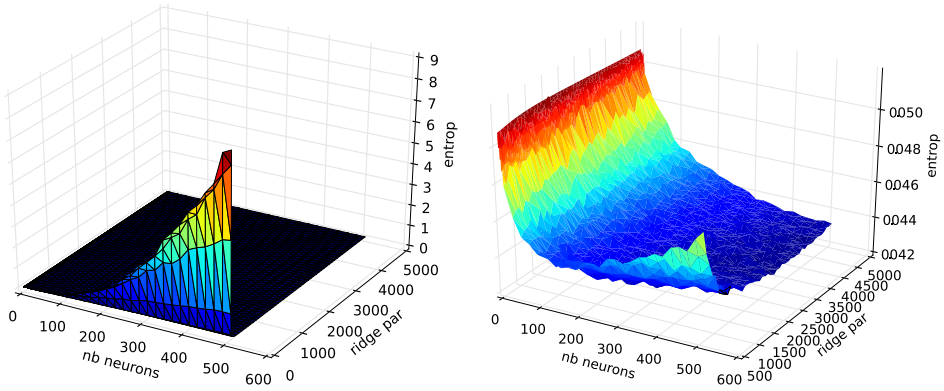


Figure 7: Niño 3.4 index, smoothed with a 3-month running mean (top) and associated SST anomalies (bottom).

Here, we compare the approximations of the N34 index based on LIM and ESNsto. A crucial parameter for the success of ESNsto was the choice of $\epsilon = 0.1$ in equation (1). It is known that this parameter controls the speed of the reservoir [Jaeger et al., 2007]. In our case, the reservoir needed to evolve, not at the scale of months (which would have corresponded to $\epsilon = 1$), but rather at longer time scales to be helpful in reconstructing the dynamics. Fig. 8 shows the relative entropy of the system as a function of the number of neurons and the ridge regularization parameter α. Without any regularization, the relative entropy increases with the number of neurons. However, for $\alpha > 10$, the relative entropy decreases with the number of neurons and with α, suggesting that ESNsto leads to a better approximation than LIM when using strong regularization. This can be interpreted as overfitting in the case of weak regularization. Adding regularization penalizes the accuracy on the training dataset (not shown) but significantly improve the generalization on the test dataset, as observed in Fig. 8(b).



(a) Ridge regularization parameter ranging from 0 to 5000

(b) Zoom for ridge regularization parameter ranging from 500 to 5000

Figure 8: Relative Entropy as a function of the number of neurons m and the ridge regularisation parameter α. Parameters used: $\epsilon = 0.1$, $l = 1/2$, $s(x) = \tanh(2x)$, $\mathbf{W}_{11}$ is drawn according to a normal law and rescaled so that $\|\mathbf{W}_{11}\| = 1$ (operator norm), $\mathbf{W}_{10}$ is drawn uniformly in $[-1.1]$, $\alpha = 1$. Washout $= 240$. The cross-validation was done with $k = 10$ blocks. Each value of the relative entropy corresponds to the average over 10 realizations of the weight matrices.

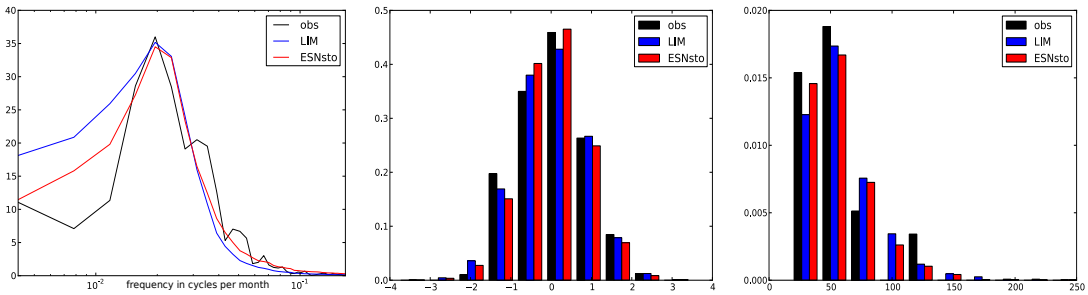
We now compare the simulations based on LIM and ESNsto with $m = 500$ and $\alpha = 3000$.

This choice of m and α is motivated by Fig. 8. The system is simulated forward for 10^5 time steps, corresponding to more than 8000 years.

As shown in Fig. 9(a), the spectrum of the N34 index is closer to the target when approximated by ESNsto than by LIM. At time scales longer than 4-5 years, the latter shows too much variability. The spectrum obtained with ESNsto also shows a too high variability, but less than the LIM and only at time scales longer than 7 years.

The distributions of the N34 index based on the LIM and ESNsto simulations are compared with the targeted distribution in Fig. 9(b). Due to the low number of target samples, it is hard to determine from the figure which distribution is closer to the target. A Kolmogorov-Smirnov test is used to determine whether the simulated distributions differ from the targeted one. The p-values in the case of LIM and ESNsto are respectively 0.45 and 0.62, meaning that, in both cases, we cannot reject the null hypothesis that the simulated and targetted N34 indices are drawn from the same distribution. The larger p-value in the case of ESNsto indicates a stronger evidence against the null hypothesis.

Fig.9(c) shows the distributions of the time interval between two El Niño events. Again, a Kolmogorov-Smirnov test is used to estimate if the simulated distributions differ from the targeted one, and gives a p-value of 0.76 for LIM and 0.98 for ESNsto. The p-value almost equals 1 in the case of ESNsto, suggesting a very good accuracy of our model to reproduce the some aspects of the dynamics governing El Niño events.



(a) Spectrum of the N34 index. Frequency in cycle per month.

(b) Distribution of the N34 index

(c) Distribution of the time interval between two El Niño events

Figure 9: Spectrum (left) and distribution (middle) of the N34 index, smoothed with a 3-month running mean. Distribution of the time interval between two El Niño events (right). Black corresponds to the targeted time series, blue to LIM, and red to ESNsto. Parameters are identical to those of Fig. (8).

7 Discussion

We have shown how to design a recurrent neural network to reproduce a target stochastic process. By doing so, we have introduced a rigorous mathematical derivation which unifies ESNs and LIM under the general principle of relative entropy minimization. Finally, we have shown how the proposed system outperforms LIM, even when the parameters are only coarsely tuned, on a simple synthetic task and on a climate example of well-known importance.

We have observed that the system is prone to over-fitting, which forced us to use large regularization parameters. Indeed, the sequential computation of connectivity matrix followed by noise matrix, implies that noise only takes care of left-overs. The connectivity matrix will try to encode as much of the signal as possible, even some part of the inherent noise. This is problematic in applications where the target process is significantly noise driven. However, when the number of time steps of the target time series is large enough to have a good ergodic approximation or if we improve the numerical differentiation scheme (e.g. using the Savitzky-Golay algorithm), we believe this drawback will vanish.

Possible extensions of this theory could include a proper treatment of the case of badly sampled data as well as the generalization of the method to space dependent diffusion coefficients. Finally, an important step in increasing accuracy of these networks would be to identify an appropriate automatic tuning of the hyper parameters of the network (e.g. the spectral radius of the reservoir connections).

8 Acknowledgment

The authors would like to thank Mantas Lukosevicius for helpful discussions. MNG was funded by the Amarsi European Project.

References

[Ackley et al., 1985] Ackley, D. H., Hinton, G. E., and Sejnowski, T. J. (1985). A learning algorithm for boltzmann machines. Cognitive science, 9(1):147–169.

[Avellaneda et al., 1997] Avellaneda, M., Friedman, C., Holmes, R., and Samperi, D. (1997). Calibrating volatility surfaces via relative-entropy minimization. Applied Mathematical Finance, 4(1):37–64.

[Barnston et al., 2012] Barnston, A. G., Tippett, M. K., L'Heureux, M. L., Li, S., and DeWitt, D. G. (2012). Skill of real-time seasonal enso model predictions during 2002-11: Is our capability increasing? Bulletin of the American Meteorological Society, 93(5):631–651.

[Baum and Petrie, 1966] Baum, L. E. and Petrie, T. (1966). Statistical inference for probabilistic functions of finite state markov chains. The annals of mathematical statistics, 37(6):1554–1563.

[Bishop, 2006] Bishop, C. (2006). Pattern recognition and machine learning.

[Box et al., 2013] Box, G. E., Jenkins, G. M., and Reinsel, G. C. (2013). Time series analysis: forecasting and control. Wiley. com.

[Buesing et al., 2011] Buesing, L., Bill, J., Nessler, B., and Maass, W. (2011). Neural dynamics as sampling: A model for stochastic computation in recurrent networks of spiking neurons. PLoS computational biology, 7(11):e1002211.

[Chatzis and Demiris, 2011] Chatzis, S. P. and Demiris, Y. (2011). Echo state gaussian process. Neural Networks, IEEE Transactions on, 22(9):1435–1445.

[Cover and Thomas, 2012] Cover, T. M. and Thomas, J. A. (2012). Elements of information theory. John Wiley & Sons.

[Dempster et al., 1977] Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm. Journal of the Royal Statistical Society. Series B (Methodological), pages 1–38.

[Deser et al., 2010] Deser, C., Alexander, M. A., Xie, S.-P., and Phillips, A. S. (2010). Sea surface temperature variability: Patterns and mechanisms. Annual Review of Marine Science, 2:115–143.

[Ellis, 2005] Ellis, R. (2005). Entropy, large deviations, and statistical mechanics, volume 1431. Taylor & Francis US.

[Funahashi and Nakamura, 1993] Funahashi, K.-i. and Nakamura, Y. (1993). Approximation of dynamical systems by continuous time recurrent neural networks. Neural networks, 6(6):801–806.

[Galtier and Wainrib, 2013] Galtier, M. and Wainrib, G. (2013). A biological gradient descent for prediction through a combination of stdp and homeostatic plasticity. Neural Computation, 25(11):2815–2832.

[Girsanov, 1960] Girsanov, I. (1960). On transforming a certain class of stochastic processes by absolutely continuous substitution of measures. Theory of Probability & Its Applications, 5(3):285–301.

[Hawkins et al., 2011] Hawkins, E., Robson, J., Sutton, R., Smith, D., and Keenlyside, N. (2011). Evaluating the potential for statistical decadal predictions of sea surface temperatures with a perfect model approach. Climate dynamics, 37(11-12):2495–2509.

[Haykin, 2005] Haykin, S. S. (2005). Adaptive Filter Theory, 4/e. Pearson Education India.

[Hinton et al., 2006] Hinton, G. E., Osindero, S., and Teh, Y.-W. (2006). A fast learning algorithm for deep belief nets. Neural computation, 18(7):1527–1554.

[Husmaier and Taylor, 1997] Husmaier, D. and Taylor, J. (1997). Predicting conditional probability densities of stationary stochastic time series. Neural Networks, 10(3):479–498.

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

[Jaeger et al., 2007] Jaeger, H., Lukoševičius, M., Popovici, D., and Siewert, U. (2007). Optimization and applications of echo state networks with leaky-integrator neurons. Neural Networks, 20(3):335–352.

[Karatzas and Shreve, 1991] Karatzas, I. and Shreve, S. (1991). Brownian motion and stochastic calculus, volume 113. Springer Verlag.

[Krogh and Riis, 1999] Krogh, A. and Riis, S. (1999). Hidden neural networks. Neural Computation, 11:541–563.

[Kullback and Leibler, 1951] Kullback, S. and Leibler, R. A. (1951). On information and sufficiency. The Annals of Mathematical Statistics, 22(1):79–86.

[Liu et al., 2011] Liu, D.-Y., Gibaru, O., and Perruquetti, W. (2011). Error analysis of jacobi derivative estimators for noisy signals. Numerical Algorithms, 58(1):53–83.

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

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

[Moon, 1996] Moon, T. K. (1996). The expectation-maximization algorithm. Signal processing magazine, IEEE, 13(6):47–60.

[Murphy, 2002] Murphy, K. P. (2002). Dynamic bayesian networks: representation, inference and learning. PhD thesis, University of California.

[Newman, 2013] Newman, M. (2013). An empirical benchmark for decadal forecasts of global surface temperature anomalies. Journal of Climate, (2013).

[Pearlmutter, 1995] Pearlmutter, B. (1995). Gradient calculations for dynamic recurrent neural networks: A survey. Neural Networks, IEEE Transactions on, 6(5):1212–1228.

[Penland, 1996] Penland, C. (1996). A stochastic model of indopacific sea surface temperature anomalies. Physica D: Nonlinear Phenomena, 98(2):534–558.

[Penland and Magorian, 1993] Penland, C. and Magorian, T. (1993). Prediction of niño 3 sea surface temperatures using linear inverse modeling. Journal of Climate, 6(6):1067–1076.

[Penland and Matrosova, 1994] Penland, C. and Matrosova, L. (1994). A balance condition for stochastic numerical models with application to the el nino-southern oscillation. Journal of climate, 7(9):1352–1372.

[Penland and Sardeshmukh, 1995] Penland, C. and Sardeshmukh, P. D. (1995). The optimal growth of tropical sea surface temperature anomalies. Journal of climate, 8(8):1999–2024.

[Rabiner, 1989] Rabiner, L. R. (1989). A tutorial on hidden markov models and selected applications in speech recognition. Proceedings of the IEEE, 77(2):257–286.

[Rasmussen, 2006] Rasmussen, C. E. (2006). Gaussian processes for machine learning.

[Rayner et al., 2003] Rayner, N., Parker, D., Horton, E., Folland, C., Alexander, L., Rowell, D., Kent, E., and Kaplan, A. (2003). Global analyses of sea surface temperature, sea ice, and night marine air temperature since the late nineteenth century. Journal of Geophysical Research: Atmospheres (1984–2012), 108(D14).

[Risken, 1996] Risken, H. (1996). The Fokker-Planck equation: Methods of solution and applications, volume 18. Springer Verlag.

[Savitzky and Golay, 1964] Savitzky, A. and Golay, M. J. (1964). Smoothing and differentiation of data by simplified least squares procedures. Analytical chemistry, 36(8):1627–1639.

[Sontag, 1997] Sontag, E. (1997). Recurrent neural networks: Some systems-theoretic aspects. Dealing with complexity: A neural network approach, pages 1–12.

[Sutskever and Hinton, 2006] Sutskever, I. and Hinton, G. (2006). Learning multilevel distributed representations for high-dimensional sequences. Technical Report UTML TR 2006-003, Department of Computer Science, University of Toronto.

[Trenberth, 1997] Trenberth, K. E. (1997). The definition of el nino. Bulletin of the American Meteorological Society, 78(12):2771–2777.

[Wainrib, 2013] Wainrib, G. (2013). Some numerical methods for rare events simulation and analysis. In Stochastic Biomathematical Models, pages 73–95. Springer.

[Williams and Zipser, 1995] Williams, R. J. and Zipser, D. (1995). Gradient-based learning algorithms for recurrent networks and their computational complexity. Back-propagation: Theory, architectures and applications, pages 433–486.

[Zanna, 2012] Zanna, L. (2012). Forecast skill and predictability of observed atlantic sea surface temperatures. Journal of Climate, 25(14):5047–5056.