System Identification of a Vertical Riser Model with Echo State Networks [⋆]

Eric A. Antonelo, Eduardo Camponogara, Agustinho Plucenio

Department of Automation and Systems Engineering, Federal University of Santa Catarina, Florianópolis, Brazil (e-mail: erantone@elis.ugent.be).

Abstract: System identification of highly nonlinear dynamical systems, important for reducing time complexity in long simulations, is not trivial using more traditional methods such as recurrent neural networks (RNNs) trained with back-propagation through time. The recently introduced Reservoir Computing (RC) [*] approach to training RNNs is a viable and powerful alternative which renders fast training and high performance. In this work, a single Echo State Network (ESN), a flavor of RC, is employed for system identification of a vertical riser model which has stationary and oscillatory signal behaviors depending of the production choke opening input variable. It is shown experimentally that these different behaviors are learned by constraining the high-dimensional reservoir states to attractor subspaces in which the specific behavior is represented. Further experiments show the stability of the identified system.

[*] The term *reservoir* used here is not related to reservoirs in oil and gas industry.

Keywords: Echo State Networks, system identification, vertical riser, reservoir computing.

1. INTRODUCTION

The simulation of nonlinear process models in optimization tasks and (nonlinear) model predictive control (MPC) usually requires a significant computational effort, especially when the model is composed of many inter-related higher-order components. Sometimes, the nonlinear dynamic process equations are not known in advance, or only the linearized version is modeled, which may not always account for the right process nonlinear behavior. In these cases, a set of process samples, if available, may be used to identify the nonlinear dynamic system, for instance, using artificial neural networks (Cessac, 2010).

The system identification task, which seeks to model the process dynamics, can be based on a *grey* or *black* box model. The former assumes a priori knowledge, for instance, in the form of a model structure. Parameter estimation is relatively easy if the model structure is known, which is rarely the case. The latter, or black box model, is more commonly used, and do not assume a priori knowledge, using only the set of samples (input-output mappings) to find the model dynamics.

The most commonly used approaches for nonlinear system identification are Volterra series models (Rugh, 1981), NARMAX models (Billings, 2013), and neural network models (Nelles, 2001). Feedforward neural networks can only model static input-output mappings. So, in order to model the dynamic properties of nonlinear systems, a tapped-delay line can be used with these networks (or with other methods such as nonlinear regression), providing a finite window of past inputs. This type of approach does not really model a dynamic system (i.e., there is no internal state). On the other hand, recurrent neural networks (RNNs) do possesses an internal state, and may be used as an universal approximation method for dynamical systems. Training an RNN was traditionally done using backpropagation through time (Werbos, 1990). The drawbacks of this method are that the training is an iterative and slow process without global convergence guarantee, which can undergo bifurcations. It also requires substantial expert practice to do it correctly.

A very efficient method for training RNNs was recently coined as Reservoir Computing (RC) (Verstraeten et al., 2007) to designate a research area sharing common characteristics in learning RNNs: the network should be composed of two main parts, a recurrent pool of neurons, with randomly generated and fixed synaptic weights, called reservoir, and a linear adaptive readout output layer which transforms the reservoir states to the actual system's output. As only the output layer needs to be trained (usually via linear regression methods), the training is simplified and global convergence guaranteed. RC originated in the works of Jaeger and Haas (2004) as Echo State Networks (ESNs) when using analog neurons and of (Maass et al., 2002) as Liquid State Machines (LSMs) when using spiking neurons.

The reservoir has its weights chosen such that its dynamic regime is usually situated at the edge of stability. This is done in order to provide rich nonlinear transformations of its input. In this way, it can be viewed as a dynamic nonlinear kernel, projecting the input to a high-dimensional dynamic space, in which linear regression or classification can

[⋆] This work was partially funded by CNPq under grants 471978/2013-2 and 501507/2013-2.

be more easily performed. Numerous applications, relying on the powerful temporal processing capabilities of RC, have been derived: navigation and localization of mobile robots in partially observable environments (Antonelo and Schrauwen, 2014, 2012), periodic signal generation with nanophotonic reservoir computing (Fiers et al., 2014), hierarchical control of robotic arms (Waegeman et al., 2013), speech recognition (Triefenbach et al., 2013), etc.

In this work, the powerful capabilities of RC are employed for identifying a relatively complex vertical riser model which possess qualitatively different dynamic behaviors depending on the values of the input signal (actuator). The input-output mapping to be modeled corresponds to estimating the bottom hole pressure (output) based solely on the production choke opening input variable.

The contribution of this paper is two-fold: first, as far as the authors know, this is the first use of RC for vertical riser modeling; secondly, by identifying a relatively complex vertical riser model which has two main distinct dynamic behaviors, we show that a single RC network can simultaneously learn qualitatively distinct signal behaviors based on the value of the input signal. These signals correspond to hard-shaped and oscillatory curves, whose nature differ significantly from each other. After training, the RC network can generate stably both of these signals, in accordance with the input-output relationship present in the training samples and without feedback from the original model measurements.

The idea of learning to generate different *behaviors* or patterns can be compared to generation of robotic behaviors in Antonelo and Schrauwen (2014), in which different behaviors are projected to different regions in the high-dimensional space of the reservoir, by shifting the operating point of the reservoir with binary inputs. This makes it possible to learn qualitatively different behaviors, as well as to smoothly switch between them. In the current work, instead of having a binary input shifting mechanism, the modeling of different nonlinear dynamic patterns is a direct implicit result of the relationship between the input-output mapping obtained from the variables of the vertical riser model. Particularly, the square-shaped changes in the production choke opening variable induces one of the following signal behaviors in the bottom hole pressure variable: a shift to a new constant value or an oscillatory behavior. Related recent research which deals with long term memory mechanisms in RC for learning multiple dynamic pattern generation can be found in Jaeger (2014).

This paper is organized as follows. Section 2 presents the vertical riser model used for generating training samples. The following section elaborates on the methods used in this work, presenting a more detailed explanation of Reservoir Computing, and its application to the particular problem of identifying the vertical riser model. The experimental results are given in Section 4 and the conclusion in Section 5.

2. VERTICAL RISER MODEL

The model from Di Meglio et al. (2009) was selected for representing the complex phenomena involved in multiphase flow dynamics observed in vertical risers. Besides, it has also been used in other studies (Petit, 2011; Di Meglio et al., 2010).

The model is based on first principles of fluid dynamics to represent the oscillatory flow behavior in risers, typically referred to as slugging flow. The oscillations arise from the accumulation of gas in an elongated bubble that is formed below the bottom of the riser, as a consequence of an obstruction to the gas flow. The pressure in the bubble builds up with incoming of gas until reaching a critical pressure, a condition that causes discharge of gas to the riser which causes a turbulence in the multiphase flow.

The model has three states which are the mass of liquid in the riser ($m_{l,r}$), the mass of gas flowing with liquid phase ($m_{g,r}$), and the mass of gas stuck in the bubble ($m_{g,eb}$). These state variables are related by the following equations of mass balance:

$$\dot{m}_{g,eb}(t) = (1-\epsilon)w_{g,in} - w_g(t) \tag{1a}$$
$$\dot{m}_{g,r}(t) = \epsilon w_{g,in} + w_g(t) - w_{g,out}(t) \tag{1b}$$
$$\dot{m}_{l,r}(t) = w_{l,in} - w_{l,out}(t) \tag{1c}$$

where $w_{g,in}$ and $w_{g,out}$ (resp. $w_{l,in}$ and $w_{l,out}$) are the mass flow rates of gas (resp. liquid) entering (in) and leaving (out) the riser, $w_g(t)$ is the flow from the bubble to the riser, and $\epsilon \in (0,1)$ is the fraction of the gas that flows straight to the riser, whereas $(1-\epsilon)$ is the fraction that accumulates in the bubble.

A virtual valve is introduced at the bottom point of the riser to represent the obstruction to gas flow: the pressure in the gas bubble rises when this valve is closed; the valve opens when the pressure in the gas bubble exceeds the pressure at bottom of the riser, allowing the gas of the bubble to flow into the riser that, in turn, reduces the bubble pressure until it gets below the pressure at bottom of the riser which forces the valve to close. When flowing from the bubble into the riser, the gas expels the liquid stored inside the riser and later causes a burst in oil production.

A choke at the top of the riser enables the control of the outlet flow. The control action consists of the opening $u \in [0,1]$ of this choke. The model of Di Meglio et al. (2009) assumes a constant flow of gas and liquid into the riser. Under this assumption, the riser outflows are given by the choke equations that follow:

$$w_{l,out} \approx C_c \max(p_{r,top} - p_s, 0)u \tag{2a}$$
$$w_{g,out} \approx \frac{m_{g,r}}{m_{l,r}} w_{l,out} \tag{2b}$$

where $p_{r,top}$ is the pressure at the top of the riser, upstream of the production choke, p_s is the pressure at the separator, and C_c is a positive choke constant.

The flow through the virtual valve is defined by:

$$w_g = C_g \max(p_{eb} - p_{r,bh}, 0) \tag{3a}$$

where p_{eb} is the pressure of the gas in the elongated bubble, $p_{r,bh}$ is the pressure downstream the virtual choke, and C_g is a positive choke constant.

The pressures that appear in the equations above are derived from the ideal gas law and the gravitational pressure caused by the masses, being defined as follows:

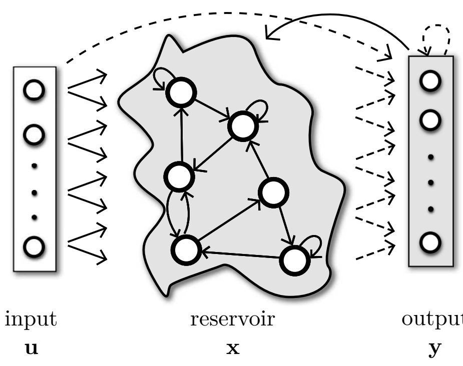


Fig. 1. Reservoir Computing (RC) network. The reservoir is a non-linear dynamical system usually composed of recurrent sigmoid units. Solid lines represent fixed, randomly generated connections, while dashed lines represent trainable or adaptive weights. The output feedback from the output layer to the reservoir is important for more complex tasks requiring, for instance, self-sustaining oscillations.

$$p_{eb} = \frac{RT}{MV_{eb}} m_{g,eb} \tag{4a}$$

$$p_{r,top} = \frac{RT}{M\left(V_r - \frac{m_{l,r}}{\rho_l}\right)} m_{g,r} \tag{4b}$$

$$p_{r,bh} = p_{r,top} + \frac{g, \sin\theta}{A} m_{l,r} \tag{4c}$$

where V_{eb} is the volume of the elongated bubble which is assumed constant, M is the gas molar mass, R is the constant of the ideal gases, T is the temperature inside the riser, V_r is the riser volume, ρ_l is the density of the liquid phase, g is the standard gravity constant, A is the cross section area of the riser, and θ is the the mean inclination of the riser.

3. RESERVOIR COMPUTING

3.1 ESN model

An ESN is composed of a discrete hyperbolic-tangent RNN, the reservoir, and of a linear readout output layer which maps the reservoir states to the actual output. Let n_i, n_r and n_o represent the number of input, reservoir and output units, respectively, $\mathbf{u}[n]$ the n_i-dimensional external input, $\mathbf{x}[n]$ the n_r-dimensional reservoir activation state, $\mathbf{y}[n]$ the n_o-dimensional output vector, at discrete time n. Then the discrete time dynamics of the ESN is given by the state update equation

$$\begin{aligned}\mathbf{x}[n+1] =& (1-\alpha)\mathbf{x}[n] + \alpha f(\mathbf{W}_\mathrm{r}^\mathrm{r}\mathbf{x}[n] + \mathbf{W}_\mathrm{i}^\mathrm{r}\mathbf{u}[n] + \\ & \mathbf{W}_\mathrm{o}^\mathrm{r}\mathbf{y}[n] + \mathbf{W}_\mathrm{b}^\mathrm{r}),\end{aligned} \tag{5}$$

and by the output computed as:

$$\mathbf{y}[n+1] = g\left(\mathbf{W}_\mathrm{r}^\mathrm{o}\mathbf{x}[n+1] + \mathbf{W}_\mathrm{i}^\mathrm{o}\mathbf{u}[n] + \mathbf{W}_\mathrm{o}^\mathrm{o}\mathbf{y}[n] + \mathbf{W}_\mathrm{b}^\mathrm{o}\right) \tag{6}$$

$$= g\left(\mathbf{W}^\mathrm{out}\left(\mathbf{x}[n+1], \mathbf{u}[n], \mathbf{y}[n], 1\right)\right) \tag{7}$$

$$= g\left(\mathbf{W}^\mathrm{out}\mathbf{z}[n+1]\right), \tag{8}$$

where: α is the leak rate (Jaeger et al., 2007; Schrauwen et al., 2007); $f(\cdot) = \tanh(\cdot)$ is the hyperbolic tangent activation function, commonly used for ESNs; g is a post-processing activation function (in this paper, g is the identity function); $\mathbf{W}^\mathrm{out}$ is the column-wise concatenation of $\mathbf{W}_\mathrm{r}^\mathrm{o}$, $\mathbf{W}_\mathrm{i}^\mathrm{o}$, $\mathbf{W}_\mathrm{o}^\mathrm{o}$ and $\mathbf{W}_\mathrm{b}^\mathrm{o}$; and $\mathbf{z}[n+1] = (\mathbf{x}[n+1], \mathbf{u}[n], \mathbf{y}[n], 1)$ is the extended reservoir state, i.e., the concatenation of the state, the previous input and output vectors and a bias term, respectively.

The matrices $\mathbf{W}_\mathrm{from}^\mathrm{to}$ represent the connection weights between the nodes of the complete network, where r, i, o, b denotes *reservoir*, *input*, *output*, and *bias*, respectively. All weight matrices representing the connections to the reservoir, denoted as $\mathbf{W}_\cdot^\mathrm{r}$, are initialized randomly (represented by solid arrows in Figure 1), whereas all connections to the output layer, denoted as $\mathbf{W}_\cdot^\mathrm{o}$, are trained (represented by dashed arrows in Figure 1).

Although output feedback given by the projection $\mathbf{W}_\mathrm{o}^\mathrm{r}\mathbf{y}[n]$ is not always used for ESNs, the learning task in this paper requires this feedback connection for generating fixed-point and oscillatory dynamics. We disregard the connections $\mathbf{W}_\mathrm{b}^\mathrm{r}$ and $\mathbf{W}_\mathrm{o}^\mathrm{o}$.

Next, the procedures for reservoir creation and dynamics tuning are presented. The non-trainable connection matrices $\mathbf{W}_\mathrm{r}^\mathrm{r}, \mathbf{W}_\mathrm{i}^\mathrm{r}, \mathbf{W}_\mathrm{o}^\mathrm{r}, \mathbf{W}_\mathrm{b}^\mathrm{r}$ are usually generated from a random distribution, such as a Gaussian distribution $N(0,1)$ or a uniform discrete set $\{-1, 0, 1\}$. During this initialization, two parameters are used:

- the connection fraction $c_\mathrm{from}^\mathrm{to}$ corresponds to the percentage of nonzero weights in the respective connection matrix $\mathbf{W}_\mathrm{from}^\mathrm{to}$.
- $\upsilon_\mathrm{from}^\mathrm{to}$ corresponds to the scaling of the respective connection matrix $\mathbf{W}_\mathrm{from}^\mathrm{to}$.

While the connectivity between units in $\mathbf{W}_\mathrm{i}^\mathrm{r}$ and $\mathbf{W}_\mathrm{r}^\mathrm{r}$ is not that important (Schrauwen et al., 2009) for analog networks, the **scaling** of these matrices have a great influence on the reservoir dynamics (Verstraeten et al., 2007) and must be tuned for achieving optimal performance. Nevertheless, sparse connectivity is usually set for these matrices, which, depending on the implementation, can save memory space.

The reservoir connection matrix $\mathbf{W}_\mathrm{r}^\mathrm{r}$ is initialized to values drawn from a Normal distribution. The dynamic regime of the reservoir is set by rescaling the weights $\mathbf{W}_\mathrm{r}^\mathrm{r}$ such that the resulting system is stable and exhibits rich dynamics. As the ESN is usually nonlinear, this can be achieved by studying a linearized version of the ESN around the equilibrium point (Kuznetsov, 1998). Under this assumption, a necessary condition to guarantee the **Echo State Property** (ESP) (Jaeger, 2001) for ESNs, i.e., a reservoir with fading memory [1], is to rescale $\mathbf{W}_\mathrm{r}^\mathrm{r}$ such that the maximal singular value of $\mathbf{W}_\mathrm{r}^\mathrm{r}$ is smaller than unity.

However, a better alternative is to rescale $\mathbf{W}_\mathrm{r}^\mathrm{r}$ such that its spectral radius $\rho(\mathbf{W}_\mathrm{r}^\mathrm{r}) < 1$ (Jaeger, 2001), in order to get richer dynamics. Although it does not guarantee the ESP, in practice it has been empirically observed that this criterium works well and often produces analog sigmoid ESNs with ESP for any input. For most applications, the best performance is attained with a reservoir that

[1] The Echo State Property states conditions for the ESN principle to work. It can be understood as having a reservoir with fading memory which asymptotically washes out any information from initial conditions.

operates at the **edge of stability**, e.g., $\rho(\mathbf{W}_\mathrm{r}^\mathrm{r}) = 0.99$. Furthermore, spectral radius closer to unity as well as larger input scaling makes the reservoir more non-linear, which has a deterioration impact on the memory capacity as side-effect (Verstraeten et al., 2010).

The scaling of these non-trainable weights is a parameter which should be chosen according to the task at hand empirically, analyzing the behavior of the reservoir state over time, or by grid searching over parameter ranges.

Most temporal learning tasks require that the timescale present in the reservoir match the timescales present in the input signal as well as in the task space. This matching can be done by the use of a leak rate ($\alpha \in (0, 1]$) and/or by resampling the input signal. For instance, low leak rates yield reservoirs with more memory which can *hold* the previous stimuli for longer time spans.

When more complex learning tasks are required, which need unbounded-time memory and oscillatory dynamics, then feedback connections (from the output layer to the reservoir layer) are essential. The presence of feedback connections allow the reservoir to enter in a free run mode after training: the predicted output at timestep n will be used as input to the reservoir at the next timestep. This allows for the formation of attractors and oscillatory dynamics. During the training stage, instead, teacher-forcing is used: the desired output (target) from the training samples is fed back to the reservoir. Furthermore, stabilization of the system with output feedback is a concern to be handled. That can be achieved by state noise injection (Jaeger, 2002) or regularizing the readout output (wyffels et al., 2008).

3.2 Readout Output Training

The training of RC networks takes place specifically for the readout output layer from Fig. 1, that is, we need to find $\mathbf{W}^\mathrm{out}$ in (6). For that, the reservoir is driven by an input sequence $\mathbf{u}(1), \ldots, \mathbf{u}(n_s)$ which yields a sequence of extended reservoir states $\mathbf{z}(1), \ldots, \mathbf{z}(n_s)$ using (5).

The desired target outputs $\hat{\mathbf{y}}[n]$ are collected row-wise into a matrix $\hat{\mathbf{Y}}$. The generated extended states are collected row-wise into a matrix $\mathbf{X}$ of size $n_s \times (n_r + n_i + n_o + 1)$ if output feedback is present. The training of the output layer is done by using the **Ridge Regression** method (Bishop, 2006), also called *Regularized Linear Least Squares* or *Tikhonov regularization* (Tychonoff and Arsenin, 1977):

$$\tilde{\mathbf{W}}^\mathrm{out} = (\mathbf{X}^\top \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^\top \hat{\mathbf{Y}} \tag{9}$$

where $\tilde{\mathbf{W}}^\mathrm{out}$ is the column-wise concatenation of $\mathbf{W}_\mathrm{r}^\mathrm{o}$, $\mathbf{W}_\mathrm{i}^\mathrm{o}$, $\mathbf{W}_\mathrm{o}^\mathrm{o}$ (without the bias term $\mathbf{W}_\mathrm{b}^\mathrm{o}$); n_s denotes the total number of training samples and the initial state is $\mathbf{x}(0) = \mathbf{0}$.

It is important to note that there is an initial transient during the generation of reservoir states $\mathbf{x}[n]$ using (5) due to the fading memory of the reservoir, which may be undesired for the readout training. So, the usual procedure to deal with this is to disregard the first n_{wd} samples in a process called **warm-up drop** so that only the samples $\mathbf{z}[n], n = n_{wd}, n_{wd} + 1, ..., n_s$ are collected into the matrix $\mathbf{X}$.

The learning of the RC network is a fast process without local minima. Once trained, the resulting RC-based system can be used for real-time operation on moderate hardware since the computations are very fast (only matrix multiplications of small matrices).

Error measure For regression tasks, the Normalized Root Mean Square Error (NRMSE) is used as a performance measure and is defined as:

$$\mathrm{NRMSE} = \sqrt{\frac{\langle (\hat{y}[n] - y[n])^2 \rangle}{\sigma^2_{\hat{y}[n]}}}, \tag{10}$$

where the numerator is the mean squared error of the output $y[n]$ and the denominator is the variance of desired output $\hat{y}[n]$.

3.3 System identification

The challenge of identifying the vertical riser model from Di Meglio et al. (2009) is that it presents two distinct regions: one stable, and another unstable area characterized by oscillations (see Fig. 2). The behavior of the target signal (the bottom hole pressure) is qualitatively distinct in these two regions.

It has been shown that RC networks can model self-generating attractor patterns such as the digit 8 in Cartesian coordinates (wyffels et al., 2008); and central pattern generators with modulable amplitude, and shift (Wyffels and Schrauwen, 2009; Li J, 2011). The feedback connections for this type of task are mandatory, as it requires a long-term (non-fading) memory sufficient to sustain either an oscillation or a constant value. In Fig. 2, the relationship between the changes in input (top plot) and the dynamical outcome in the output (middle plot) can be modeled directly by training a single RC network, as it will be shown in the following section. Furthermore, the simultaneous learning of oscillatory and stationary signals with a single RC network is first reported here, as far as the authors know.

4. EXPERIMENTAL RESULTS

The dataset used to train the RC network was generated in Matlab by simulating the ordinary differential equations (ODEs) of the vertical riser model described by Di Meglio et al. (2009) (see Section 2). The dataset consists of a desired single input - single output relationship $(u[n], \hat{y}[n])$, where $u[n]$ is the *production choke opening* (actuator), while $\hat{y}[n]$ corresponds to the *bottom hole pressure* variable. The input $u[n]$ can take values in $(0, 1]$ and $y[n]$ from $[3 * 10^6, 17 * 10^6]$ Pa approximately. We generated $n = 24,000$ seconds (about six and a half hours) of simulation using the ODE equations to collect the pairs $(u[n], \hat{y}[n])$ using a randomly created, squared-shaped, input signal $u[n]$.

For parameter selection, we used grid search with a 9-fold random cross-validation over the following set of parameters: leak rate α, input scaling υ_i^r, spectral radius $\rho(\mathbf{W}_\mathrm{r}^\mathrm{r})$ and the regularization parameter λ. Other parameters are configured arbitrarily, such as the reservoir which has 400 neurons. It is known that as the reservoir gets bigger in number of neuronal units, and if accompanied by a properly regularized training procedure to avoid overfitting, its performance gets better since its memory capacity and processing power also increase. We found that a 400 neuron reservoir was enough to achieve good results, but the task could be achieved with smaller reservoirs. The remaining parameters are set according to Section 3.1, that is, all weight matrices connecting to the reservoir ($\mathbf{W}^{\mathrm{r}}_{\mathrm{i}}$ and $\mathbf{W}^{\mathrm{r}}_{\mathrm{o}}$) are randomly generated from a uniform distribution $[-1, 1]$ (which means a connection fraction of 1) and scaled according to the values given by the input scaling υ^r_i and output scaling υ^r_o (in our case, $\upsilon^r_i = \upsilon^r_o$). This means that the magnitude of the influence of the input *production choke opening* on the reservoir is the same compared to the magnitude of the influence of the output *bottom hole pressure* (note that both signals are normalized). $\mathbf{W}^{\mathrm{r}}_{\mathrm{b}}$ is set to zero since the experiments have shown that this extra bias non-linearity did not help to improve performance. Training the network (computing $\tilde{\mathbf{W}}^{\mathrm{out}}$) is done applying equation (9). A test set of $2,400$ seconds (or 40 minutes) was used to evaluate the trained network. The experiments were implemented in Python using the Oger toolbox (Verstraeten et al., 2012).

The optimal parameter configuration given by the aforementioned procedure for the results shown in the next figures are as follows: $\alpha = 0.1$, $\upsilon^r_i = 0.35$, $\rho(\mathbf{W}^{\mathrm{r}}_{\mathrm{r}}) = 1$ and $\lambda = 10^{-2.5}$. Figure 2 shows the estimations of the trained RC network using a training dataset consisting of $20,000$ samples (one per second), and a test dataset composed of $4,000$ samples (or 66.6 minutes). The first plot shows the input signal, i.e., the production choke opening used to test the identified trained system. The target and actual network outputs for the bottom hole pressure are shown in the next plot in dashed and solid black lines, respectively. The red vertical line defines the timestep at which the reservoir starts running in free-run mode: using its own output predictions $\mathbf{y}[n]$ as feedback signals. Previous to that, the target signal $\hat{\mathbf{y}}[n]$ from the samples is teacher-forced in order to set the internal reservoir state to an appropriate state (i.e., $\hat{\mathbf{y}}[n]$ is used in (5) in place of $\mathbf{y}[n]$). It can be seen that after the red vertical line, the network can adequately model the behavior of the identified system: it was able to model fixed point and oscillatory regions with only one network, two distinct behaviors whose simultaneous modeling with a single neural network is not a trivial task.

From these two plots, one can also note that these two behaviors or, also, the different operating points which $y[n]$ can achieve depending on the input signal $u[n]$ fed to the network are, actually, learned through shifting the operating point of the reservoir with the input signal $u[n]$. This can be seen in the third plot of Fig. 2, which shows the first three principal components from applying Principal Component Analysis (PCA) on the reservoir states. As $u[n]$ changes value, the operating point of the reservoir is taken, for instance, from a fixed point region to an oscillatory region between minutes 10 and 15. The distinction between these dynamic regions are learned during the training phase. Apart from the role of $u[n]$ in the reservoir, $y[n]$, by being fed back to the reservoir, functions as reinforcing memory for maintaining either the fixed point or the oscillatory behavior. Both are very important for the final result.

The stability of the generated $y[n]$ output signal is essential for the identification task and can be achieved by using noise injection during training (Jaeger, 2002) or finding the optimal regularization parameter λ in ridge regression (wyffels et al., 2008). To test the hypothesis of stability, two experiments were devised using the test data: the first experiment consisted of adding a single large and increasing perturbation during 6 seconds, whereas the second was done by adding Gaussian noise to $y[n]$ at each timestep. The results in Fig. 3 show in the first plot the stability of the generated output in response to two large perturbations during minutes 35 and 48. The perturbations take place during a fixed point and oscillatory behavior, and are handled effectively by the RC network which is able to bring back the output to the desired value or behavior. In particular, during the oscillation, the perturbation affects the reservoir states (Fig. 3(c)) such that the magnitude of the oscillation is increased and not removed until the next change in the input signal $u[n]$. Further experiments should also include the improvement of this issue.

Fig. 3(b) shows the reservoir stability robustness to random Gaussian noise on the output $y[n]$, considering a standard deviation of 10^{-2} for the normalized output signal in $[0, 1]$. Other magnitude values of Gaussian noise were tested and the results summarized in Fig. 3(d). The performance deteriorates only from $\sigma_{noise} = 10^{-1}$ on.

5. CONCLUSION

In this paper, it was shown that RC networks are very well suited for performing system identification of vertical riser models having multiple different dynamic behaviors. Depending on the interval in which the production choke opening input variable was situated, either a stationary signal or oscillatory behavior followed (in the bottom hole pressure output variable), corresponding to stable and unstable regions for the vertical riser model, respectively. The trained RC network is able to reproduce the dynamics of the model, and were shown to be robust to perturbations. Furthermore, the procedure described in this paper to identify the system with the RC network is not limited to the chosen riser model, but is almost directly applicable to other dynamic models including gas-lift oil wells and other riser models.

It is important to note that it is the shifting of the operating point of the reservoir resulting from the changes in the input signal that allow the organization of subspaces in the high-dimensional reservoir state space. These subspaces formed in the training process are able to constrain the dynamic behavior of the reservoir to the ones chosen by the input-output relationship present in the training samples. In this work, these nonlinear behaviors could be classified as stationary or oscillatory signals, which are simultaneously learned by a single RC network.

Further research includes the use of the identified system for speeding up simulations in optimization and model predictive control tasks as well as the additional investigation of the capabilities of RC for modeling more complex dynamic models based on samples from the OLGA simulator and from a real vertical riser. The results from the RC viewpoint could be transfered to the modeling of robotic

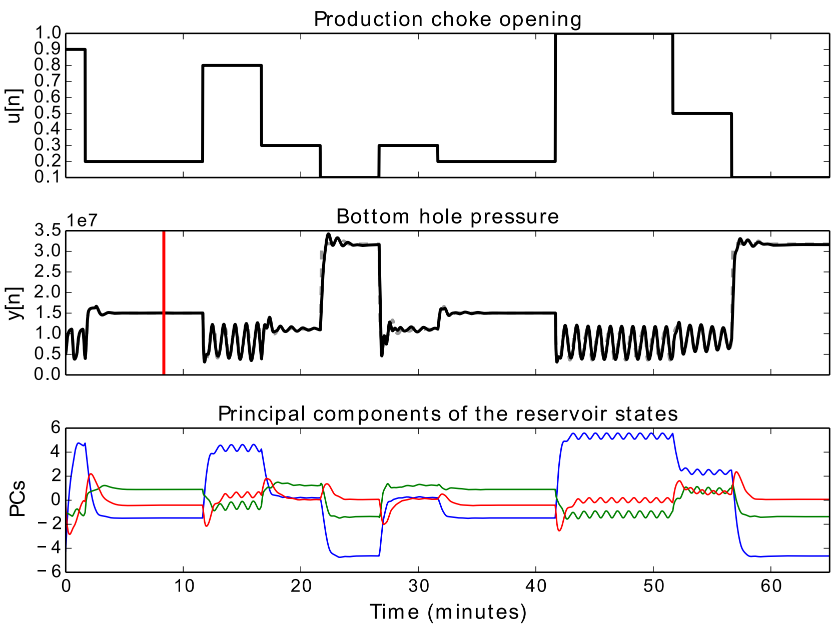
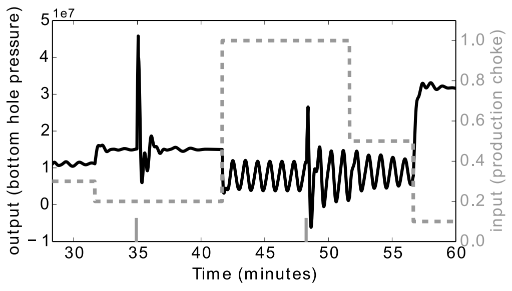
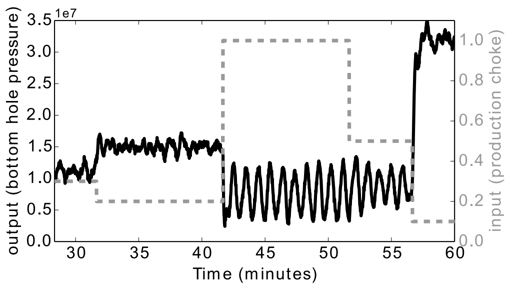


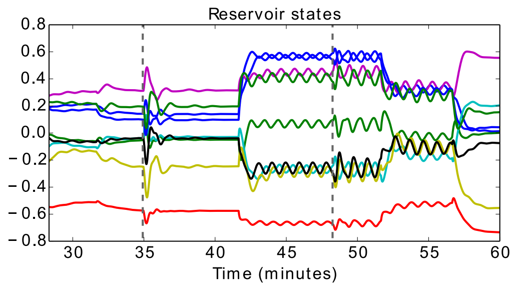
Fig. 2. Estimation of bottom hole pressure with trained RC network. The first plot shows the test input fed to the RC network (the production choke opening), whereas the second plot shows the target and predicted output (the bottom hole pressure) as dashed grey and solid black lines, respectively. The red vertical line marks the time at which the reservoir runs in free-run mode, feeding back its output prediction. The bottom plot shows the three principal components of the reservoir states over time, resulting from applying the PCA algorithm.



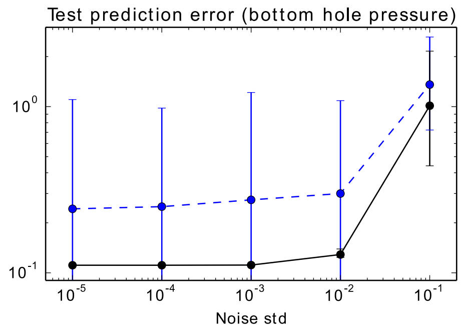
(a) two perturbations



(b) $\sigma_{noise} = 10^{-2}$



(c) reservoir states during perturbation



(d) Effect of noise on performance

Fig. 3. Noise robustness results during testing (output prediction). (a) 2 large perturbations applied during 6 seconds to $y[n]$, at minutes 35 and 48 (see indication by grey ticks), are overcome by the trained network. The dashed grey line represents the corresponding input signal $u[n]$. (b) Random noise is applied to $y[n]$ at each timestep, sampled from a Gaussian distribution with zero mean and standard deviation σ_{noise}. (c) The corresponding reservoir states for the same perturbations in (a) whose application moments are marked with dashed vertical lines. The first two top plots show that the trained system is very robust to noise. (d) shows the prediction error over different levels of noise (σ_{noise}). The solid curve corresponds to results for the optimal reservoir from Fig. 2 and the dashed curve considers different randomly generated reservoirs.

behaviors, where the change to a stationary signal would mean that the dynamical system commands the robot to alter from a moving behavior to stay stably still, for instance.

ACKNOWLEDGEMENTS

The authors gratefully acknowledge the contribution of Marco Aurélio S. Aguiar related to the understanding of the Di Meglio et al. (2009) vertical riser model.

REFERENCES

Antonelo, E.A. and Schrauwen, B. (2014). On learning navigation behaviors for small mobile robots with reservoir computing architectures. *IEEE Transactions on Neural Networks and Learning Systems*, PP(99). doi: 10.1109/TNNLS.2014.2323247.

Antonelo, E.A. and Schrauwen, B. (2012). Learning slow features with reservoir computing for biologically-inspired robot localization. *Neural Networks*, 25(1), 178–190.

Billings, S.A. (2013). *Nonlinear system identification: NARMAX methods in the time, frequency, and spatio-temporal domains*. Wiley.

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

Cessac, B. (2010). A view of neural networks as dynamical systems. *International Journal of Bifurcation and Chaos*, 20(6), 1585–1629.

Di Meglio, F., Kaasa, G.O., and Petit, N. (2009). A first principle model for multiphase slugging flow in vertical risers. In *Proceedings of the 48th IEEE Conference on Decision and Control*, 8244–8251. doi: 10.1109/CDC.2009.5400680.

Di Meglio, F., Kaasa, G., Petit, N., and Alstad, V. (2010). Model-based control of slugging flow: an experimental case study. In *American Control Conference (ACC), 2010*, 2995–3002. IEEE.

Fiers, M., Van Vaerenbergh, T., wyffels, F., Verstraeten, D., Schrauwen, B., Dambre, J., and Bienstman, P. (2014). Nanophotonic reservoir computing with photonic crystal cavities to generate periodic patterns. *IEEE Transactions on Neural Networks and Learning Systems*, 25(2), 344–355.

Jaeger, H., Lukosevicius, M., and Popovici, D. (2007). Optimization and applications of echo state networks with leaky integrator neurons. *Neural Networks*, 20(3), 335–352.

Jaeger, H. (2001). The "echo state" approach to analysing and training recurrent neural networks. Technical Report GMD Report 148, German National Research Center for Information Technology.

Jaeger, H. (2002). Short term memory in echo state networks. Technical Report GMD Report 152, German National Research Center for Information Technology.

Jaeger, H. (2014). Controlling recurrent neural networks by conceptors. *CoRR*, abs/1403.3369.

Jaeger, H. and Haas, H. (2004). Harnessing nonlinearity: predicting chaotic systems and saving energy in wireless telecommunication. *Science*, 304(5667), 78–80.

Kuznetsov, Y.A. (1998). *Elements of Applied Bifurcation Theory*. Springer.

Li J, J.H. (2011). Minimal energy control of an esn pattern generator. Technical report, Jacobs University Bremen, School of Engineering and Science.

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.

Nelles, O. (2001). *Nonlinear system identification: from classical approaches to neural networks and fuzzy models*. Springer.

Petit, N. (2011). Control problems for one-dimensional fluids and reactive fluids with moving interfaces. In *Advances in the theory of control, signals and systems with physical modeling*, 323–337. Springer.

Rugh, W.J. (1981). *The Volterra and Wiener Theories of Nonlinear Systems*. The Johns Hopkins University Press.

Schrauwen, B., Busing, L., and Legenstein, R. (2009). On Computational Power and the Order-Chaos Phase Transition in Reservoir Computing. In *Advances in Neural Information Processing Systems 21 (NIPS 2008)*, 1425–1432. Curran Associates, Inc.

Schrauwen, B., Defour, J., Verstraeten, D., and Van Campenhout, J. (2007). The introduction of time-scales in reservoir computing, applied to isolated digits recognition. In *Proceedings of the 17th International Conference on Artificial Neural Networks (ICANN 2007)*, volume 4668 of *LNCS*, 471–479. Springer Berlin Heidelberg.

Triefenbach, F., Jalalvand, A., Demuynck, K., and Martens, J.P. (2013). Acoustic modeling with hierarchical reservoirs. *IEEE Trans. Audio, Speech, and Language Processing*, 21(11), 2439–2450. doi: 10.1109/TASL.2013.2280209.

Tychonoff, A. and Arsenin, V.Y. (1977). *Solutions of Ill-Posed Problems*. Washington: Winston & Sons.

Verstraeten, D., Dambre, J., Dutoit, X., and Schrauwen, B. (2010). Memory versus non-linearity in reservoirs. In *Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN)*, 1–8. Barcelona, Spain.

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

Verstraeten, D., Schrauwen, B., Dieleman, S., Brakel, P., Buteneers, P., and Pecevski, D. (2012). Oger: modular learning architectures for large-scale sequential processing. *Journal of Machine Learning Research*, 13, 2995–2998.

Waegeman, T., Hermans, M., and Schrauwen, B. (2013). MACOP modular architecture with control primitives. *Frontiers in Computational Neuroscience*, 7(99), 1–13.

Werbos, P.J. (1990). Backpropagation through time: what it does and how to do it. *Proc. IEEE*, 78(10), 1550–1560.

Wyffels, F. and Schrauwen, B. (2009). Design of a central pattern generator using reservoir computing for learning human motion. In *Proceedings of the ECSIS Symposium on Advanced Technologies for Enhanced Quality of Life*, 118–122. doi:10.1109/AT-EQUAL.2009.32.

wyffels, F., Schrauwen, B., and Stroobandt, D. (2008). Stable output feedback in reservoir computing using ridge regression. In *International Conference on Artificial Neural Networks*.