Echo State Networks for Data-driven Downhole Pressure Estimation in Gas-lift Oil Wells

Eric A. Antonelo[a], Eduardo Camponogara[a], Bjarne Foss[b]

[a]*Department of Automation and Systems, UFSC, Brazil*
[b]*Department of Engineering Cybernetics, NTNU, Norway*

Abstract

Process measurements are of vital importance for monitoring and control of industrial plants. When we consider offshore oil production platforms, wells that require gas-lift technology to yield oil production from low pressure oil reservoirs can become unstable under some conditions. This undesirable phenomenon is usually called *slugging flow*, and can be identified by an oscillatory behavior of the downhole pressure measurement. Given the importance of this measurement and the unreliability of the related sensor, this work aims at designing data-driven soft-sensors for downhole pressure estimation in two contexts: one for speeding up first-principle model simulation of a vertical riser model; and another for estimating the downhole pressure using real-world data from an oil well from Petrobras based only on topside platform measurements. Both tasks are tackled by employing Echo State Networks (ESN) as an efficient technique for training Recurrent Neural Networks. We show that a single ESN is capable of robustly modeling both the slugging flow behavior and a steady state based only on a square wave input signal representing the production choke opening in the vertical riser. Besides, we compare the performance of a standard network to the performance of a multiple timescale hierarchical architecture in the second task and show that the latter architecture performs better in modeling both big transients and small oscillations.

Keywords: echo state network, gas-lift oil wells, vertical riser, reservoir computing, soft sensor, system identification, data-driven

1. Introduction

In order to achieve enhanced oil production in offshore oil wells, the well's downhole conditions must be monitored. For that, permanent downhole gauge sensors (PDG) are usually installed at the bottom of the well providing measurements for temperature and pressure. The downhole pressure measured is one of the most important variables for monitoring, optimization and control of oil well production, being essential in assessing the dynamics of the oil well.

One important phenomenon observed in pipelines and oil wells corresponds to high oscillatory flow or *slugging* flow. This is usually the case for gas-lift oil wells, which employ the gas-lift technique in order to extract the oil from deepwater or low pressure wells. The artificially injected gas diminishes the density of the well fluid, which, in turn, makes possible its extraction with the created difference in pressure. Stabilization techniques which tackle these oscillatory behaviors in multiphase flows are necessary and have been designed by experts in academia and industry [25, 19, 29, 32, 14, 12]. These methods are usually based on the stabilization of the downhole pressure through choke actuators on the gas-lift flow rate and the well production. Unfortunately, PDG sensors, as they are installed in hazardous environments, have a prohibitive cost for maintenance or replacement [13], and also their premature failure is not uncommmon. Additionally, perturbations and noise can affect the PDG sensor measurements, making it an unreliable information source.

Given the importance of measuring or estimating the downhole pressure and the unreliability of the PDG sensor which measures this pressure, there have been several works which seek to create models that can estimate the downhole pressure based on other topside measurements. These predictive models, usually called soft-sensors, are important for quality control and production safety and have been extensively developed in the past decades [34]. Some of them use knowledge of the oil well physics [1] to design a nonlinear observer for the states of the multiphase flow in order to estimate the downhole pressure, while others are based on black-box system identification approaches [33, 31]. While the first approach can take advantage of the a priori knowledge for a refined analysis and more advanced control schemes [12], the latter approach is quicker, does not require extensive modeling, being well suited to identify unknown models. The current work follows the latter approach and assumes hardly any a priori model knowledge.

Much of the literature in system identification relies on the use of NARMAX models [8] or feedforward artificial neural networks (ANNs) with tapped delayed lines at the input layer [33] to account for dynamic behaviors or temporal processing. Although it is possible to introduce dynamics into the model using a time-window of previous inputs, a more interesting general way is to use Recurrent Neural Networks (RNNs) as universal approximators for dynamical systems [17]. However, training RNNs is not straightforward since gradient descent on the cost function, implemented as the backpropagation-though-time (BPTT) technique [41], has drawbacks which include a slow training process, no global convergence guarantee, possibility of bifurcations and problem of the vanishing gradient [20]. It also requires substantial expert practice to do it correctly.

RNNs can provide a type of state-dependent computation much like cortical functioning in the brain [10], where the trajectory of a high-dimensional dynamical system reflects both the current input as well as previously received input stimuli. Reservoir Computing (RC) [38] is a term recently coined to designate this paradigm of computation based on transients of a fixed dynamical system (such as an RNN). Most common RC models are the Echo State Networks (ESNs) [23] when analog neurons are used and Liquid State Machines (LSMs) [28] when spiking neurons are considered as dynamical reservoirs. In RC, the network (see Fig. 1) should be composed of two main parts, a recurrent high-dimensional pool of neurons, with randomly generated and fixed synaptic weights, called reservoir [1], and a linear adaptive readout output layer which projects 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 (unlike in BPTT). The reservoir can be viewed as a dynamic nonlinear kernel, projecting the input to a high-dimensional dynamic space, in which linear regression or classification can 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 [5], periodic signal generation with nanophotonic reservoir computing [15], hierarchical control of robotic arms [40], speech recognition

[1] The term *reservoir* is used to designate the randomly generated RNN in RC throughout this paper, and is not related to reservoirs in oil and gas industry unless explicitly stated.

[35], etc.

This work builds on previous results [3, 2] to elaborate on an unified RC-based approach for estimation of the downhole pressure in oil wells. Two main cases are addressed in this paper: system identification of a simulated vertical riser model and the design of a soft-sensor for gas-lift oil wells using real-world data. Both tasks are solved by using the same efficient black-box RC-based architecture [23] for modeling the particular input-output mappings of the target dynamical nonlinear system (i.e., the oil well). The motivation to use RC for building soft-sensors of downhole pressure is four-fold:

1. it can be applied to problems when the model is unknown (most real-world processes can not be completely modeled or a considerable modeling effort is needed) when compared to an observer design approach;
2. RC allows the addition of new output units at the output layer (using the same reservoir) which can be trained separately, without corrupting previously trained units, being useful if additional output estimation units are required with time;
3. inverse models can easily be built so that even a subset of the input measurements can be predicted in case some sensors become faulty, ultimately improving overall performance (as shown in [6] for a robotic task);
4. lastly, RC provides a quick and efficient training for RNNs when compared to methods based on gradient descent.

The first task in this work is motivated by the fact that the simulation of nonlinear process models in optimization tasks usually requires a significant computational effort, especially when the model is composed of many inter-related higher-order components. Thus, the replacement of the relatively computationally expensive simulation by a trained RC network yields a significant economy on execution time (as it will be shown in Section 3). The second task tackles modeling dynamical nonlinear relationships from real-world oil well data obtained from Petrobras in order to design a soft-sensor which estimates the downhole pressure (given by the PDG sensor under ordinary situations) based on measurements from the seabed production platform.

This paper is organized as follows. Section 2 presents the RC model used in the rest of this work: the Echo State Network. The following section (Sec. 3) introduces the problem and relevance of vertical riser modeling, the experimental setup and corresponding results. Section 4 tackles the second

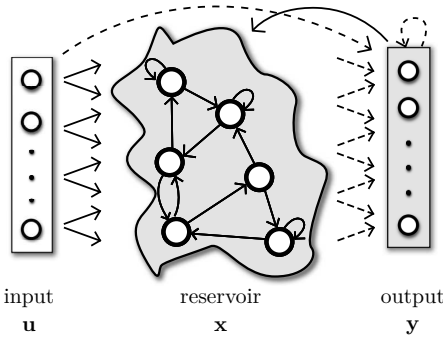


Fig. 1: Reservoir Computing (RC) network. The reservoir is a non-linear dynamical system usually composed of recurrent sigmoid units. Solid lines represent fixed, randomly generated connections, while dashed lines represent trainable or adaptive weights.

task of designing a soft-sensor for estimating the downhole pressure in a gas-lift oil well. Conclusions and future work are drawn in Section 5.

2. Reservoir Computing

2.1. 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

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

and by the output computed as:

$$\mathbf{y}[n+1] = g\left(\mathbf{W}^{\mathrm{o}}_{\mathrm{r}}\mathbf{x}[n+1] + \mathbf{W}^{\mathrm{o}}_{\mathrm{i}}\mathbf{u}[n] + \mathbf{W}^{\mathrm{o}}_{\mathrm{o}}\mathbf{y}[n] + \mathbf{W}^{\mathrm{o}}_{\mathrm{b}}\right) \tag{2}$$

$$= g\left(\mathbf{W}^{\mathrm{out}}\left(\mathbf{x}[n+1], \mathbf{u}[n], \mathbf{y}[n], 1\right)\right) \tag{3}$$

$$= g\left(\mathbf{W}^{\mathrm{out}}\mathbf{z}[n+1]\right), \tag{4}$$

where: α is the leak rate [24, 30]; $f(\cdot) = \tanh(\cdot)$ 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); $\mathbf{W}^{\text{out}}$ is the column-wise concatenation of $\mathbf{W}^{\text{o}}_{\text{r}}$, $\mathbf{W}^{\text{o}}_{\text{i}}$, $\mathbf{W}^{\text{o}}_{\text{o}}$ and $\mathbf{W}^{\text{o}}_{\text{b}}$; 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 $\mathbf{W}^{\text{to}}_{\text{from}}$ 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 $\mathbf{W}^{\text{r}}$, are initialized randomly (represented by solid arrows in Fig. 1), whereas all connections to the output layer, denoted as $\mathbf{W}^{\text{o}}$, are trained (represented by dashed arrows in Fig. 1). We disregard the connections $\mathbf{W}^{\text{r}}_{\text{b}}$ and $\mathbf{W}^{\text{o}}_{\text{o}}$. The non-trainable connection matrices $\mathbf{W}^{\text{r}}_{\text{r}}$, $\mathbf{W}^{\text{r}}_{\text{i}}$ are usually generated from a Gaussian distribution $N(0, 1)$ or a uniform discrete set $\{-1, 0, 1\}$. During this random initialization, the matrix $\mathbf{W}^{\text{r}}_{\text{i}}$ is multiplied by the parameter called input scaling $\upsilon^{\text{r}}_{\text{i}}$ (or $\upsilon^{\text{r}}_{\text{o}}$ for $\mathbf{W}^{\text{r}}_{\text{o}}$).

The weights from the reservoir connection matrix $\mathbf{W}^{\text{r}}_{\text{r}}$ are obtained randomly through a Normal distribution ($N(0, 1)$) and then rescaled such that the resulting system is stable but still exhibits rich dynamics. A general rule to create good reservoirs is to set the reservoir weights such that the reservoir has the *Echo State Property* (ESP) [21], i.e., a reservoir with fading memory. A common method used in literature is to rescale $\mathbf{W}^{\text{r}}_{\text{r}}$ such that its spectral radius $\rho(\mathbf{W}^{\text{r}}_{\text{r}}) < 1$ [21]. 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. It is important to note that 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 [37]. This *scaling* of matrices is important because it influences greatly the reservoir dynamics [38] and, in this way, must be chosen according to the task at hand empirically, analyzing the behavior of the reservoir states over time, or by grid searching.

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 (as the task in Section 3), then feedback connections from the output layer to the reservoir layer ($\mathbf{W}^{\text{r}}_{\text{o}}\mathbf{y}[n]$) are essential. The presence of feedback connections allows 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. During the training stage, instead, teacher-forcing is used: the target output 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 [21] or regularizing the readout output [43].

2.2. Training

Training the RC network means finding $\mathbf{W}^{\text{out}}$ in (2), that is, the weights for readout output layer from Fig. 1. That is usually done by linear regression methods on the reservoir states generated by simulating (1) with a given input signal $\mathbf{u}[n]$. In this work, we use Ridge Regression [9]. See Appendix A for more details on the training process.

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).

3. Vertical riser modeling

3.1. Introduction

In this first part, RC is employed for identifying a model [11] which displays the complex phenomena involved in multiphase flow dynamics observed in vertical risers. A scheme of the riser can be seen in Fig. 2. 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 elongated bubbles that is formed below the bottom of the riser, as a consequence of an obstruction to the gas flow. The pressure in the bubbles 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. A detailed description of the model is given in Appendix B.

The challenge of identifying the vertical riser model from [11] with a single network is that it presents two distinct regions: one stable, and another area characterized by oscillations (see Fig. 3(a)). The behavior of the target signal (the bottom hole pressure) is qualitatively distinct in these two regions depending on the value of the actuator input (the production choke opening).

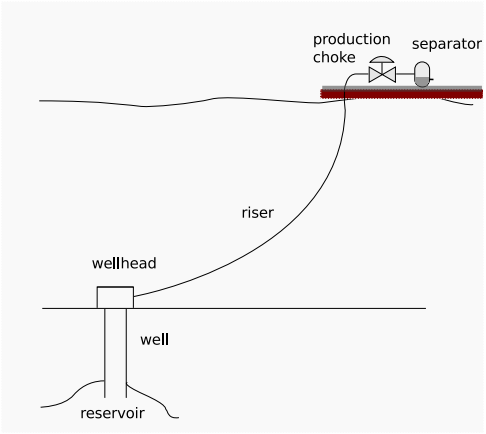


Fig. 2: Riser setup.

It has been shown that RC networks can model self-generating attractor patterns such as the digit 8 in Cartesian coordinates [43]; and central pattern generators with modulable amplitude, and shift [42, 27]. 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. As far as the authors know, the simultaneous learning of oscillatory and stationary signals with a single RC network is first reported here [2].

3.2. Experimental setup

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 [11] (see Appendix B). 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 v_i^r, spectral radius $\rho(\mathbf{W}_\mathrm{r}^\mathrm{r})$ and the regularization parameter λ. Other parameters

[2] This paper builds upon a previous conference publication [3].

are configured arbitrarily, such as the reservoir which has 400 neurons. It is known that as the reservoir increases in its 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 [3]. The remaining parameters are set according to Section 2.1, that is, all weight matrices connected to the reservoir ($\mathbf{W}_{\mathrm{i}}^{\mathrm{r}}$ and $\mathbf{W}_{\mathrm{o}}^{\mathrm{r}}$) 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_i^r and output scaling υ_o^r (in our case, $\upsilon_i^r = \upsilon_o^r$). 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). $\mathbf{W}_{\mathrm{b}}^{\mathrm{r}}$ 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 (A.1). 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 [39].

The optimal parameter configuration given by the aforementioned procedure for the results shown in the next figures are as follows: $\alpha = 0.1$, $\upsilon_i^r = 0.35$, $\rho(\mathbf{W}_{\mathrm{r}}^{\mathrm{r}}) = 1$ and $\lambda = 10^{-2.5} = 0.0032$.

3.3. Results

Fig. 3(a) 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 black and blue 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

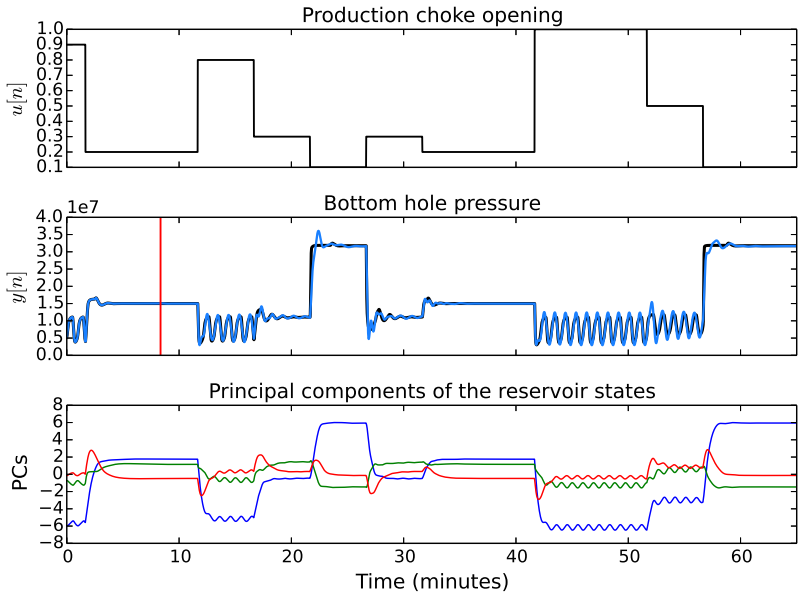
[3] One can use a 30-unit reservoir for this task, but the probability of randomly obtaining a rich dynamical reservoir (and consequently a good performance) is lower.

used in (1) 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 both fixed point and oscillatory regions using only a single network.

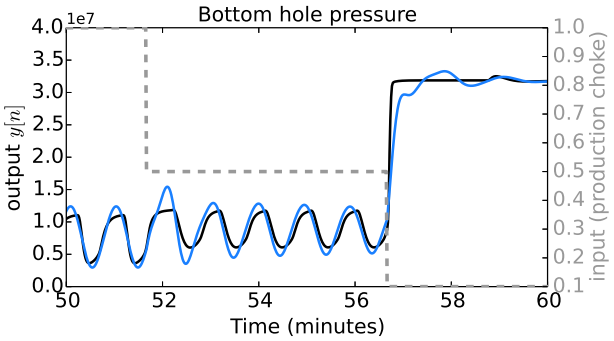
From these two plots, one can also note that these two behaviors or, also, the different operating points which $y[n]$ can achieve depending on the input signal $u[n]$ fed to the network are, actually, learned through shifting the operating point of the reservoir with the input signal $u[n]$. This can be seen in the third plot of Fig. 3(a), which shows the first three principal components from applying Principal Component Analysis (PCA) on the reservoir states. As the value $u[n]$ changes, 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 is 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. Fig. 3(b) shows the prediction of the bottom hole pressure zoomed in over an interval of 10 minutes.

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 [21] or finding the optimal regularization parameter λ in ridge regression [43]. 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. 4(a) show 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. 4(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. 4(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. 4(d). The performance deteriorates only from $\sigma_{noise} = 10^{-1}$ on.



(a)



(b)

Fig. 3: Estimation of bottom hole pressure with trained RC network. (a) 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 black and blue 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. (b) Closer look at the predicted downhole pressure for 10 minutes (the dashed grey line corresponds to the choke input).

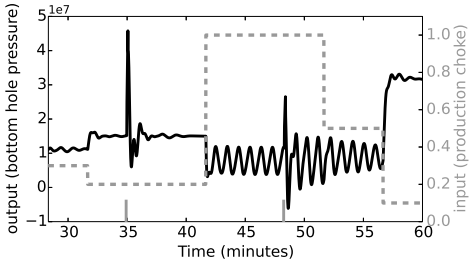
As a comparison in terms of simulation time, we observed that running a 200 unit (400 unit) trained RC network (in Python) is circa 18 times (12 times) faster than running the ODE solver (*ode23tb* function in Matlab) for the same number of time steps and using the same computer.

4. Soft-sensor for downhole pressure estimation

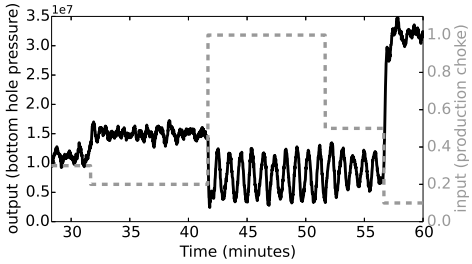
4.1. Introduction

In the previous section, the RC network was trained with simulation data in order to estimate the bottom hole pressure only from the production choke opening as input signal. To be able to sustain either a constant value or oscillations at the output layer, output feedback connections to the reservoir layer were essential. Now, in this section, we deal with noisy and messy data originating from a real-world oil well from Petrobras to build RC-based soft-sensors [16]. The schematics of the oil well can be seen in Fig. 6. The task here is to build a soft-sensor which can infer the downhole pressure $y(t)$ based on a set of input sensor measurements $\mathbf{u}(t)$ coming only from the more easily accessible platform location. Although feedback connections were not required, the task here is considerably complex since the underlying process generate very nonlinear behaviors which change over time probably due to well and oil reservoir changing conditions. Additionally, these signal behaviors present multiple timescales. Because of this, we propose a hierarchical deep architecture with 3 hidden layers called H-RC (Fig. 4.1). The first layer (*Res.1*) has multiple decoupled small reservoirs, each one having a different leak rate. This layer yields a state space sensitive to signals working at different timescales. A similar approach was shown in [7], where a single reservoir with multiple leak rates for individual neurons yielded better performance in robot localization tasks than using only one leak rate. The second layer (PCA) learns the principal components of the previous *Res.1* layer by finding a linear projection from a high-dimensional reservoir space into a low dimension orthogonal space. This is done by Principal Component Analysis [26], which computes the eigenvectors of the covariance matrix with largest eigenvalues. The third hidden layer (*Res.2*) is composed of reservoir units, representing a nonlinear and temporal expansion on the previous PCA layer. The final output layer (which estimates the downhole pressure $y(t)$) receives signals from *Res.2* layer and optionally from *Res.1* layer.

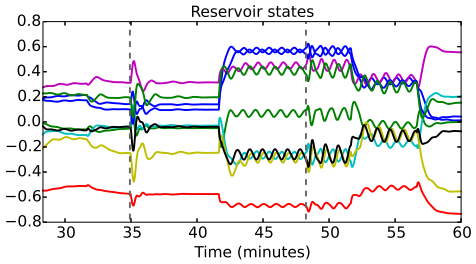
Among several existing hierarchical RC approaches in the literature, we can cite, for instance: [40] for robotic arm control using a hierarchical ar-



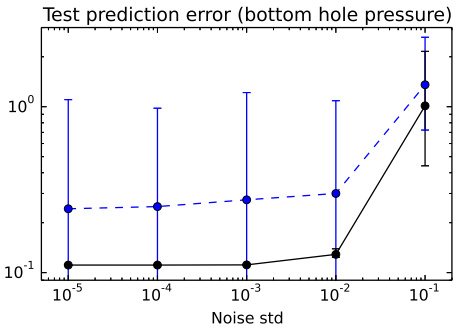
(a) two perturbations



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



(c) reservoir states during perturbation



(d) Effect of noise on performance

Fig. 4: 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. 3(a) and the dashed curve considers different randomly generated reservoirs.

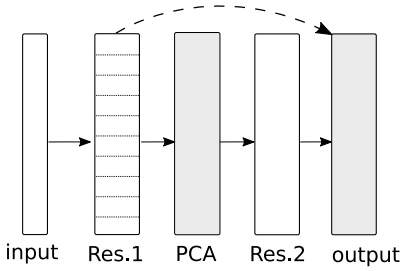


Fig. 5: Proposed H-RC architecture for soft-sensor design. The *Res.1* reservoir layer is composed of multiple decoupled reservoirs with different leak rates. Shaded layers are trained sequentially: the first one is trained by PCA and the second one by ridge regression. The *Res.2* layer expands non-linearly on the previous PCA layer. See text for a detailed description.

chitecture which combines motor primitives to achieve control of complex movements; [22] for simultaneous signal de-noising and online classification using a 3-layer recurrent architecture. [4] for self-organized (unsupervised) learning of robot localization from low-dimensional noisy infra-red sensory data. Each of them use different unsupervised and/or supervised learning techniques and are well suited to different classes of applications.

As it will be verified below, the H-RC helps to model big transients as well as small signal oscillations simultaneously when compared to a plain RC network. Besides, the PCA layer has a role of improvement on the generalization performance of the model.

4.2. Experimental setup

The input of the RC-based soft-sensor consists of 10 inputs normalized to the interval $[0, 1]$, corresponding to the 8 platform variables from Table 1 plus the openings of the gas-lift choke and production choke (unless otherwise stated). The target output variable $y(t)$ corresponds to the PDG pressure sensor. Although there are 5 months of available data (with a sampling frequency of 1 sample per minute), in this section we focus on the two most interesting months: August/2010 and December/2011. Previously, an RC model using all 5 months data [2] has been built, but it does not take into account the slugging flow phenomenon which happens in a smaller timescale. Furthermore, some unexpected transients were also not investigated more closely. The following sections aim to close this gap.

The results shown below use two types of feature selection: domain knowledge to combine input variables; or backwards variable removal (also called

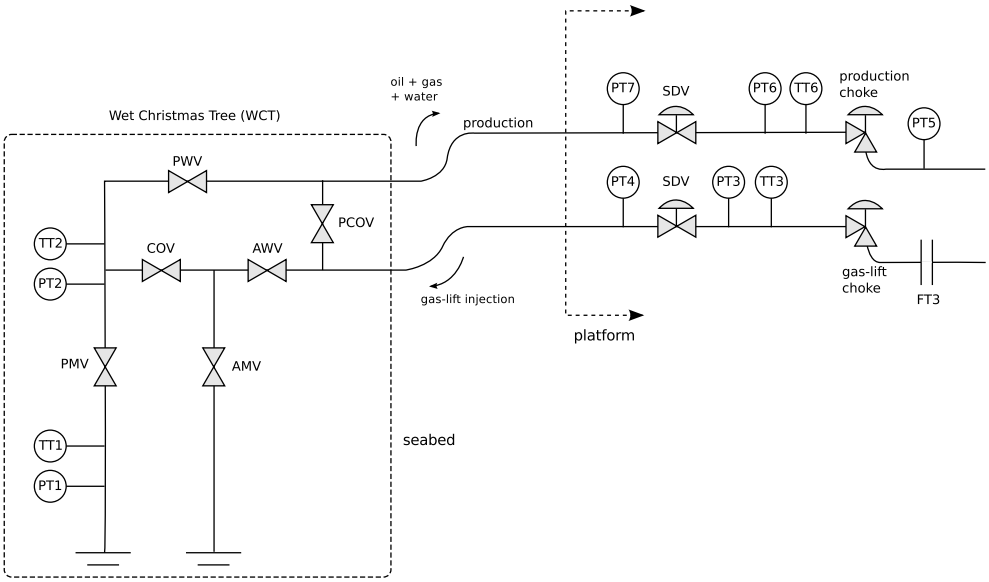


Fig. 6: Oil well scheme of a real-world well from Petrobras showing the location of sensors and chokes. FT3 is a flow rate sensor; PT# and TT# are pressure and temperature sensors. SDV stands for ShutDown Valve.

backward elimination) to find the minimal set of variables which leads to the best generalization performance. The first method substitutes PT6 and PT7 by their average $(PT6+PT7)/2$ as well as PT3 and PT4 by $(PT3+PT4)/2$, justified by the fact that the SDV valve between them is fully open, making both variables conveying the same information. Another preprocessing corresponds to substituting PT5 by the pressure drop $(PT5-PT6)$ across the production choke, since the downhole pressure is more sensitive to this pressure drop than a single pressure measurement. Thus, after this domain-based preprocessing, we get 8 input variables in total from the initially 10 available variables. On the other hand, backward elimination starts with all 10 input variables for evaluating the RC architecture, and gradually removes the variable which results in the least generalization error. From this, we can find a minimum set of variables which better models the signal during a particular period.

For learning slugging flow oscillations in August 2010 (first task), the H-RC network is configured as such: *Res.1* with 10 reservoirs of 50 units each; 3 units in PCA layer; 100 neurons in *Res.2* layer; output layer connected to PCA and *Res.1* layers. For learning both oscillations and bigger transients

simultaneously with data from December 2011 (second task), the configuration is as follows: *Res.1* with 10 reservoirs of 50 units each; 10 units in PCA layer; two pools of 100 neurons in *Res.2* layer; output layer connected only to PCA layer. The leak rate for the ten reservoirs in *Res.1* layer is as follows: $\alpha = (0.1\ 0.2\ 0.3\ 0.4\ 0.5\ 0.6\ 0.7\ 0.8\ 0.9\ 1)^T$ (i.e., each element defines the leak rate of a 50-units reservoir). We have arbitrarily set $v_{\mathrm{i}}^{\mathrm{r}} = 0.06$ and $\rho(\mathbf{W}_{\mathrm{r}}^{\mathrm{r}}) = 0.99$ for both *Res.1* and *Res.2* layers. For the second task, *Res.2* layer has one pool of units with $v_{\mathrm{i}}^{\mathrm{r}} = 0.06$ and $\alpha = 1$, and another pool with $v_{\mathrm{i}}^{\mathrm{r}} = 0.1$ and $\alpha = 0.1$.

The experiments are done using the H-RC network as well as a plain RC network with one hidden layer of 500 neurons for comparison. The latter is called 1-RC architecture, and its parameters are set with the same values as for the H-RC, except for the leak rate $\alpha = 0.5$ for all neurons, and unless otherwise stated. The setting of parameters is not very critical, and was chosen to give best generalization performance. Note that a decrease in the input scaling counteracts an increase in the spectral radius to avoid a possible loss of memory capacity and of performance [37].

4.3. Experimental results

4.3.1. Slugging flow

In this section, the results on modeling the slugging flow phenomenon for data from August/2010 are presented. The training/test datasets are created including primarily the intervals with oscillations while disregarding other irrelevant behaviors, totalling 21,600 samples (which corresponds

Table 1: Process variables

Tag	Process variable	Location	Variables Set
PT1	Downhole pressure	Seabed	Output
TT1	Downhole temperature	Seabed	—
PT2	WCT pressure	Seabed	—
TT2	WCT temperature	Seabed	—
PT3	Pressure before SDV	Platform	Input
TT3	Temperature before SDV	Platform	Input
FT3	Instantaneous gas-lift flow rate	Platform	Input
PT4	Pressure after SDV	Platform	Input
PT5	Pressure after production choke	Platform	Input
PT6	Pressure before production choke	Platform	Input
TT6	Temperature before production choke	Platform	Input
PT7	Pressure before SDV	Platform	Input

to 15 days of measurements), of which 75% are used for training. The selection of the training samples is done as in [2], interleaving training and test intervals randomly. Besides, the regularization parameter is $\lambda = 0.0001$. Backward elimination is also employed to select a subset β of the variables corresponding to $\{PT7, TT6, PT5, G.C, TT3, P.C, PT3\}$ as input to the H-RC architecture. (see Fig. 7).

Table 2 shows the NRMSE and RMSE averaged over 30 runs for different experiments, where each run considers randomly generated reservoir weights. The train and test error rates are shown in the first two rows for the 1-RC architecture and in the last two rows for the H-RC network. The asterisks in 1-RC* and H-RC* mean that the subset β are used as input variables, whereas their absence means the input variables were domain-based preprocessed (see previous section). The minimum test errors for 1-RC and H-RC are in marked in bold. In both cases, H-RC shows a better generalization performance than the 1-RC network. We also note that the backward variable elimination helped only slightly to improve the test error.

The predicted bottom hole pressure using both 1-RC and H-RC and the subset β as input variables is shown in Fig. 8 for two particular oscillatory periods. The overall behavior is captured by both networks, but a closer inspection reveals that the H-RC architecture approximates better the downhole pressure and also shows a more stable signal on average, i.e., less sensitive to noise. For instance, just before hour 56 in time axis (marked by a dashed rectangle as well as arrows), the blue line for 1-RC varies quickly showing some undesired sensibility whereas the blue line for H-RC has a smooth curve. Another example of slight quick variation in the predicted signal can be viewed between hours 52 and 54 for the 1-RC network.

Table 2: Error rates - August 2010

	1-RC	1-RC*	H-RC	H-RC*
NRMSE				
Train	0.0881	0.0851	0.0764	0.0773
Test	0.168	**0.164**	0.155	**0.150**
RMSE				
Train	0.00170	0.00164	0.00148	0.00149
Test	0.00355	**0.00338**	0.00327	**0.00308**

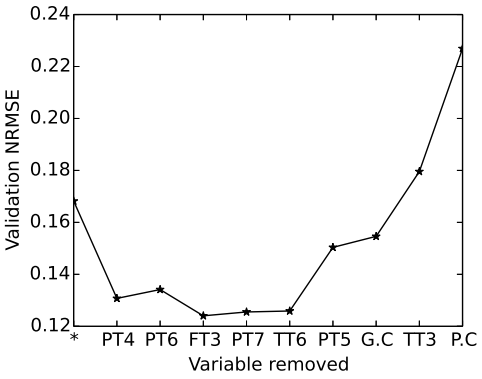


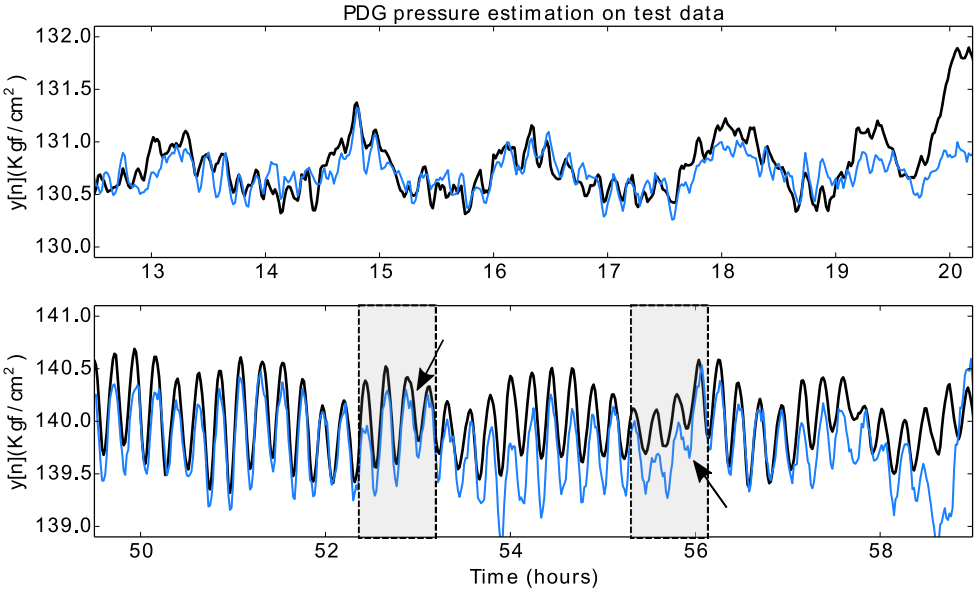
Fig. 7: Backwards Variable Removal for August 2010 and H-RC architecture. The set of variables with minimum error is: $\{PT7, TT6, PT5, G.C, TT3, P.C, PT3\}$.

4.3.2. Transients

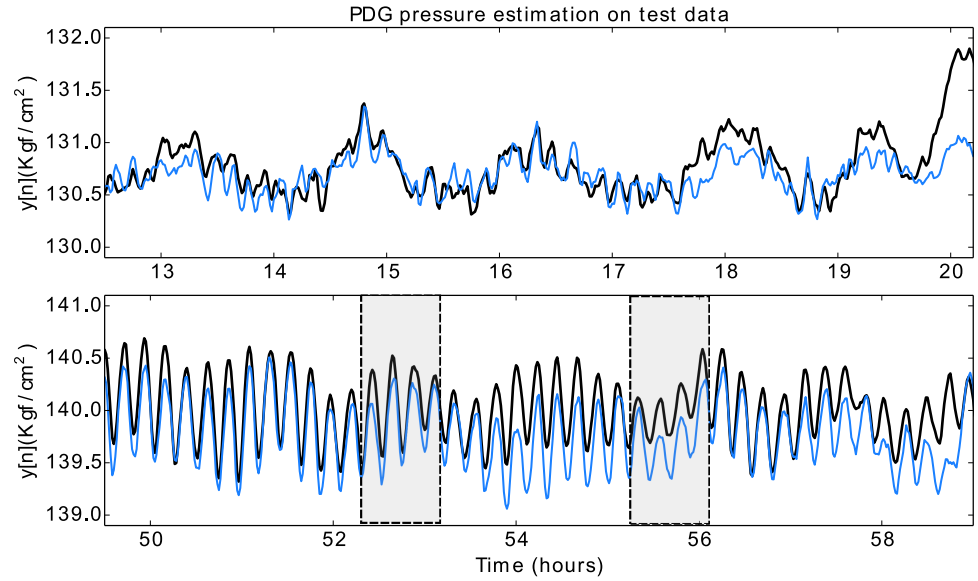
Uncommon transients in the downhole pressure can happen for instance when some control variables such as the production choke or the gas-lift choke are altered. Closing these chokes results in undesirable behavior for the downhole pressure. One example is given in Fig. 10, where the unusual transient in the downhole pressure is concomitant to the closing of the G.C (gas-lift choke). These behaviors are difficult to model because they do not occur frequently, which provides few training data for building RC estimation models.

In this section, results consider the whole month in December 2011, totaling 43,200 samples, where the first 70% of the samples were used for training and the rest for test. The parameter setting was done as described in Section 4.2, except for the 1-RC network, where the regularization parameter λ had to be set higher ($\lambda = 0.05$), while the input scaling is set to $\upsilon_{\mathrm{i}}^{\mathrm{r}} = 0.2$ and spectral radius is set to $\rho(\mathbf{W}_{\mathrm{r}}^{\mathrm{r}}) = 0.5$ as in [2].

We can check the effect of the regularization parameter λ in the test error rate by inspecting Fig. 9(a). Considering (randomly chosen) fixed weight matrices in the reservoir layers, λ can be adjusted so that generalization of the model is achieved. We can note that, for this particular initialization of the reservoir weights, the 1-RC network needs to be much more regularized than H-RC, indicating that the H-RC may have an inherent regularization due to the PCA layer in the hierarchical multiple timescale structure. We can also verify that the best 1-RC architecture for reservoirs with number

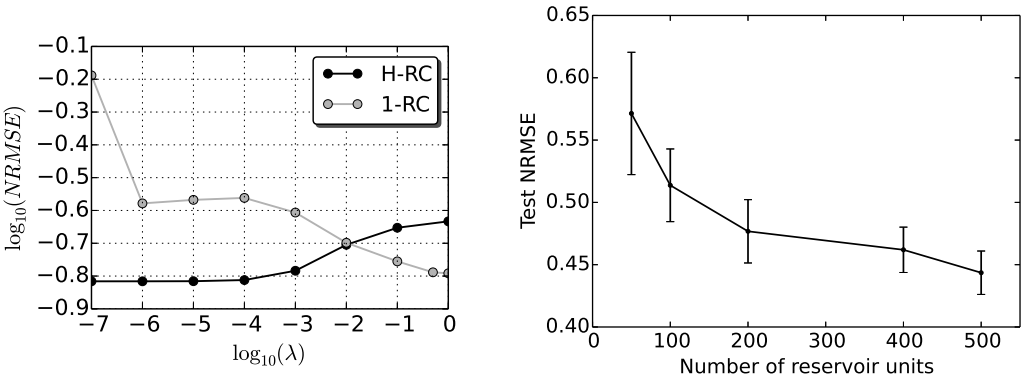


(a) 1-RC



(b) H-RC

Fig. 8: Results for learning the slugging flow phenomenon in August 2010 with the 1-RC architecture (a) and the H-RC architecture (b). Black and blue lines correspond to the target and predicted downhole pressure. See text for more details.



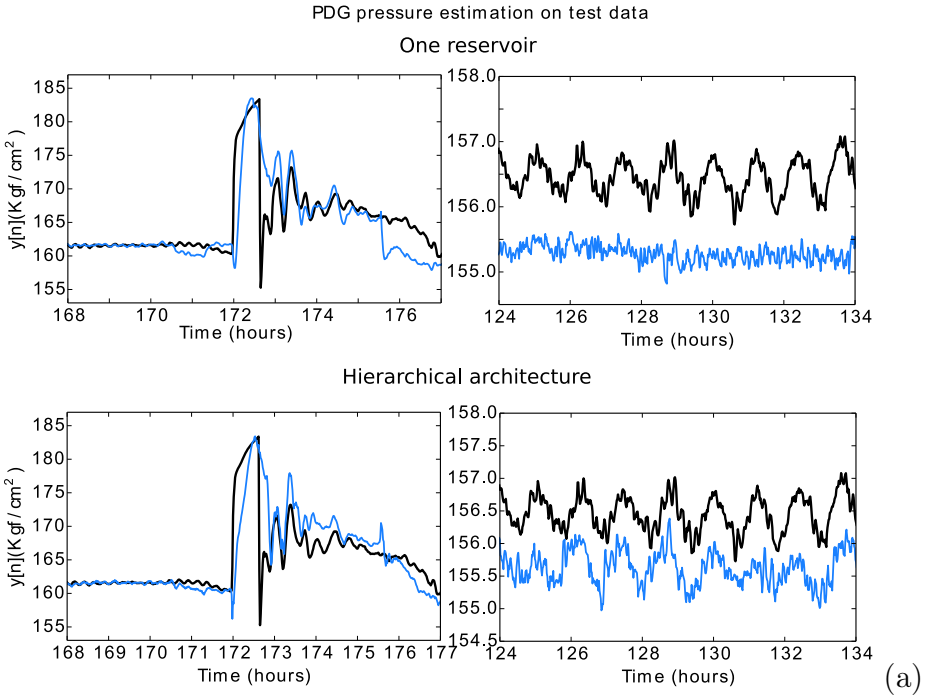
(a) (b)

Fig. 9: (a) Effect of the regularization parameter λ on the test error for H-RC and 1-RC networks for data from December 2011. (b) Test NRMSE by number of reservoir units in 1-RC architecture. Error bars are set by the standard deviation for 10 randomly generated reservoirs.

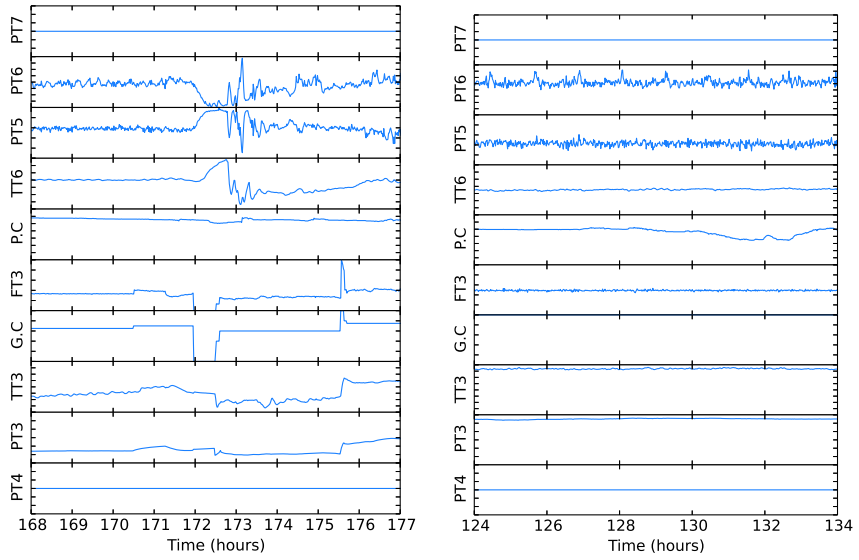
of units in the interval $[50, 500]$ is the 500 unit reservoir (Fig. 9(b)). This is expected as the training method used, Ridge Regression, regularizes the model.

Note that, although a low error rate is achieved for 1-RC when $\lambda = 0.5$, this does not mean that the network is always modeling the signal behavior correctly. In Fig. 10, we can see the target downhole pressure and the predicted output for 1-RC and H-RC networks during two different moments. The left plots show a big transient when compared to the smaller oscillations in the right plots. Although 1-RC can do well in the left plot, the same network does badly for the second oscillatory period. On the other hand, H-RC handles both the big transient as well as the second signal behavior, probably due to its multiple timescale processing in *Res.1* and *Res.2* layers.

In Fig. 11 (a), we can see the error rate according to the number of units in the PCA layer in H-RC. Each point in a line uses the same randomly generated weights (for *Res.1* and *Res.2*) so that different lines correspond to differently configured reservoirs. Additionally, each point in a given line represents the test error after sequentially training the PCA and the output layer. We can note that depending on which (random) weights were chosen for the reservoir(s), the minimum test error varies according to the number of PCA units in the H-RC architecture. This implies that the number of



(a)



(b)

Fig. 10: Results for learning a particular transient in December 2011 with 1-RC and H-RC architectures. (a) Black and blue lines correspond to the target and predicted downhole pressure. The left plots show a transient while the right plots show an oscillatory period (b) Input variables for the corresponding intervals.

PCA units is a parameter which may be aiding regularization of the model. Indeed, the number of principal components used through PCA has effects similar but not equal to ridge regression (for a detailed discussion, see [18]).

Fig. 11 (b) shows the error rate based on the number of PCA units from a different perspective. Now, the H-RC network (solid line) is run 10 times for each PCA unit configuration, where each run considers a different randomly generated reservoir for *Res.2* layer (*Res.1* layer is kept fixed). The plotted average value of the NRMSE shows a different result: on average, and given a particular fixed setting of the *Res.1* weights, networks with 7 and 8 PCA units produce lower error rates. Note that this solid line is not a general behavior for the PCA layer, as seen in Fig. 11 (a). For comparison, the removal of the *Res.2* layer from H-RC yields the results given by the grey line. As *Res.1* has fixed weights for each number of PCA units, the results are deterministic (no error bars). These 2 curves show clearly that *Res.2* layer is important for the improved performance. The dashed horizontal line represents the error rate for the *Res.1* layer (with the same weights) connected directly to the output layer, with PCA and *Res.2* layers removed.

5. Conclusion

This paper has presented a data-driven RC-based approach for learning dynamical nonlinear behaviors present in processes in the oil production system considering both simulation and real-world data. While in the first part, a single network was able to reproduce the dynamics of the model based only an unidimensional input (the oil production choke) very satisfactorily, and was shown to be robust to perturbations, in the second part, a hierarchical multiple timescale architecture (H-RC) has been proposed to deal with more complex nonlinear phenomena present in the real-world oil well data. We can conclude that the H-RC network was better suited to learn big unusual transients and small oscillations simultaneously, having a better test performance and stability of prediction, when compared to a plain RC network. The current data-driven approach supposes that no a priori knowledge is available. Thus, it is an interesting candidate for learning models from real-world noisy datasets and eventually speeding up simulation of nonlinear plants.

With respect to the first part of this paper (Section 3), further research includes additional investigation of the capabilities of RC for modeling more complex dynamical models, for instance, using the OLGA simulator or samples from a real vertical riser. Future work in the context of soft-sensors

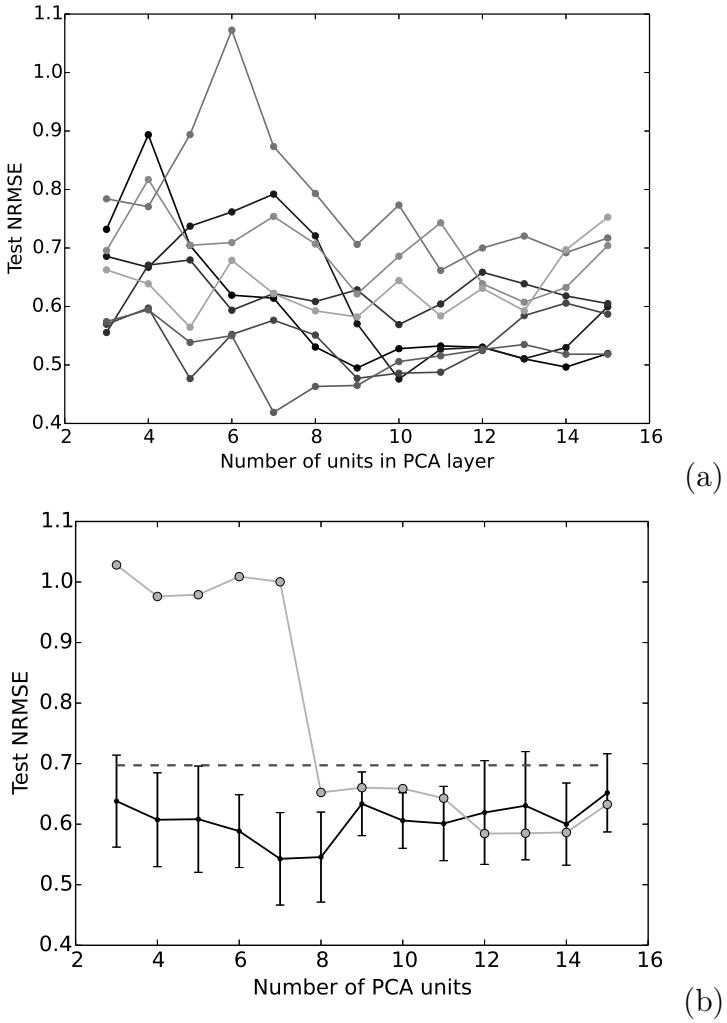


Fig. 11: Analisys of PCA layer of H-RC network for data from December 2011. (a) Test NRMSE for different number of units in PCA layer. For each line, a different randomly generated reservoir was used. (b) The solid line shows the mean test error when considering different number of PCA units and randomly generated reservoirs (standard deviation given by error bars). The grey line considers the H-RC network without the last reservoir layer (*Res.2*). The horizontal line gives the error for the Res.1 connected directly to the output layer (without PCA and Res.2 layers). See text for more details.

should tackle the online learning of the readout layer with methods such as Recursive Least Squares (RLS). This is useful for adapting the model in real time. Preliminary work has shown that it is very difficult to learn a good model through RLS and suggests that regularized online learning approaches are necessary to achieve better generalization. Another research direction is to create inverse models which take the (predicted) downhole pressure as input and predict other sensor variables. Given the existence of noisy or faulty sensors, this inverse model can then be used to improve the overall (downhole pressure) prediction.

Acknowledgements

The authors thank CNPq for the supporting fellowship, Petrobras for providing the real-world data from an oil well, and Agustinho Plucenio for the Matlab code for simulation of the vertical riser dynamics.

Appendix A. ESN training

Appendix A.1. Readout Training

Training the RC network means finding $\mathbf{W}^{\text{out}}$ in (2), that is, the weights for readout output layer from Fig. 1. 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 (1) (the initial state is $\mathbf{x}(0) = \mathbf{0}$). 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 $n_s \times (n_r + n_i + n_o + 1)$ using (1).

Then, the training of the output layer is done by using the **Ridge Regression** method [9], also called *Regularized Linear Least Squares* or *Tikhonov regularization* [36]:

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

where $\tilde{\mathbf{W}}^{\text{out}}$ is the column-wise concatenation of $\mathbf{W}_{\text{r}}^{\text{o}}$, and the optional matrices $\mathbf{W}_{\text{i}}^{\text{o}}$, $\mathbf{W}_{\text{o}}^{\text{o}}$ and n_s denotes the total number of training samples.

In the generation of $\mathbf{X}$, a process called **warm-up drop** is used to disregard a possible undesired initial transient in the reservoir starting at $\mathbf{x}(0) = \mathbf{0}$. This is achieved by dropping the first n_{wd} samples 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).

Appendix A.2. Error measures

For regression tasks, the Root Mean Square Error (RMSE) and Normalized Root Mean Square Error (NRMSE) are used as performance measures and are defined as:

$$\mathrm{RMSE} = \sqrt{\langle (\hat{y}[n] - y[n])^2 \rangle}, \tag{A.2}$$

$$\mathrm{NRMSE} = \frac{\mathrm{RMSE}}{\sigma_{\hat{y}[n]}}, \tag{A.3}$$

where the $\langle\rangle$ denotes temporal averaging, and $\sigma_{\hat{y}[n]}$ is the standard deviation of desired output $\hat{y}[n]$.

Appendix B. Vertical riser model

The model in [11] 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 bubbles ($m_{g,eb}$). These state variables are related by the following mass balance equations:

$$\dot{m}_{g,eb}(t) = (1-\epsilon) w_{g,in}(t) - w_g(t) \tag{B.1a}$$

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

$$\dot{m}_{l,r}(t) = w_{l,in}(t) - w_{l,out}(t) \tag{B.1c}$$

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 bubbles 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 bubbles.

A virtual valve is introduced at the bottom point of the riser to represent the obstruction to gas flow: the pressure in the gas bubbles rises when this valve is closed; the valve opens when the pressure in the gas bubbles exceeds the pressure at bottom of the riser, allowing the gas in the bubbles 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 bubbles 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 changing the opening $u \in [0, 1]$ of this choke. The model in [11] 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:

$$w_{l,out} \approx C_c \max(p_{r,top} - p_s, 0)u \tag{B.2a}$$

$$w_{g,out} \approx \frac{m_{g,r}}{m_{l,r}} w_{l,out} \tag{B.2b}$$

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:

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

where p_{eb} is the pressure of the gas in the elongated bubbles, $p_{r,bh}$ is the pressure downstream the virtual choke (in the bottom hole), 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:

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

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

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

where V_{eb} is the volume of the elongated bubbles 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 mean inclination of the riser.

References

[1] Aamo, O., Eikrem, G., Siahaan, H., and Foss, B. (2005). Observer design for multiphase flow in vertical pipes with gas-lift—theory and experiments. *Journal of Process Control*, 15(3):247 – 257. Available from: `http://www.sciencedirect.com/science/article/pii/S0959152404000848`.

[2] Antonelo, E. A. and Camponogara, E. (2015). An echo state network-based soft sensor of downhole pressure for a gas-lift oil well. In *16th International Conference on Engineering Applications of Neural Networks*.

[3] Antonelo, E. A., Camponogara, E., and Plucenio, A. (2015). System identification of a vertical riser model with echo state networks. In *2nd IFAC Workshop on Automatic Control in Offshore Oil and Gas Production*.

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

[5] Antonelo, E. A. and Schrauwen, B. (2015). On learning navigation behaviors for small mobile robots with reservoir computing architectures. *IEEE Transactions on Neural Networks and Learning Systems*, 26(4):763–780.

[6] Antonelo, E. A., Schrauwen, B., and Campenhout, J. V. (2007). Generative modeling of autonomous robots and their environments using reservoir computing. *Neural Processing Letters*, 26(3):233–249.

[7] Antonelo, E. A., Schrauwen, B., and Stroobandt, D. (2008). Event detection and localization for small mobile robots using reservoir computing. *Neural Networks*, 21(6):862–871.

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

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

[10] Buonomano, D. and Maass, W. (2009). State-dependent computations: Spatiotemporal processing in cortical networks. *Nature Reviews Neuroscience*, 10(2):113–125.

[11] 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*, pages 8244–8251.

[12] Di Meglio, F., Kaasa, G. O., Petit, N., and Alstad, V. (2012). Model-based control of slugging: advances and challenges. In *IFAC Workshop on Automatic Control in Offshore Oil and Gas Production 2012*.

[13] Eck, J. et al. (1999). Downhole monitoring: the story so far. *Oilfield Review*, 11(3):18–29.

[14] Eikrem, G. O., Aamo, O. M., Foss, B. A., et al. (2008). On instability in gas lift wells and schemes for stabilization by automatic control. *SPE Production & Operations*, 23(02):268–279.

[15] 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. Available from: `http://dx.doi.org/10.1109/TNNLS.2013.2274670`.

[16] Fortuna, L., Graziani, S., Rizzo, A., and Xibilia, M. G. (2007). *Soft sensors for monitoring and control of industrial processes*. Springer Science & Business Media.

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

[18] Hastie, T., Tibshirani, R., Friedman, J., and Franklin, J. (2005). *The elements of statistical learning: data mining, inference and prediction*, volume 27. Springer.

[19] Havre, K., Stornes, K. O., and Stray, H. (2000). Taming slug flow in pipelines. *ABB review*, 4:55–63.

[20] Hochreiter, S. (1998). The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 6(02):107–116.

[21] Jaeger, H. (2002). Short term memory in echo state networks. Technical Report GMD Report 152, German National Research Center for Information Technology. Available from: `http://minds.jacobs-university.de/sites/default/files/uploads/papers/STMEchoStatesTechRep.pdf`.

[22] Jaeger, H. (2014). Controlling recurrent neural networks by conceptors. Technical Report 31, Jacobs University. Available from: `http://arxiv.org/abs/1403.3369`.

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

[24] Jaeger, H., Lukosevicius, M., and Popovici, D. (2007). Optimization and applications of echo state networks with leaky integrator neurons. *Neur. Netw.*, 20(3):335–352.

[25] Jansen, F., Shoham, O., and Taitel, Y. (1996). The elimination of severe slugging—experiments and modeling. *International journal of multiphase flow*, 22(6):1055–1072.

[26] Jolliffe, I. (2002). *Principal component analysis*. Wiley Online Library.

[27] Li, J. and Jaeger, H. (2011). Minimal energy control of an ESN pattern generator. Technical Report 26, Jacobs University Bremen, School of Engineering and Science.

[28] 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.

[29] Sarica, C., Tengesdal, J. Ø., et al. (2000). A new technique to eliminate severe slugging in pipeline/riser systems. In *SPE annual technical conference and exhibition*. Society of Petroleum Engineers.

[30] 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 *Proc. of the 17th ICANN*, volume 4668 of *LNCS*, pages 471–479. Springer.

[31] Shang, C., Yang, F., Huang, D., and Lyu, W. (2014). Data-driven soft sensor development based on deep learning technique. *Journal of Process Control*, 24(3):223–233.

[32] Skofteland, G. and Godhavn, J. (2003). Suppression of slugs in multiphase flow lines by active use of topside choke-field experience and experimental results. In *Proceedings of multiphase*, volume 3.

[33] Teixeira, B. O., Castro, W. S., Teixeira, A. F., and Aguirre, L. A. (2014). Data-driven soft sensor of downhole pressure for a gas-lift oil well. *Control Eng. Practice*, 22:34–43.

[34] Tham, M. T., Montague, G. A., Morris, A. J., and Lant, P. A. (1991). Soft-sensors for process estimation and inferential control. *Journal of Process Control*, 1(1):3–14.

[35] 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.

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

[37] Verstraeten, D., Dambre, J., Dutoit, X., and Schrauwen, B. (2010). Memory versus non-linearity in reservoirs. In *Proc. of the IEEE IJCNN*, pages 1–8.

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

[39] 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.

[40] Waegeman, T., Hermans, M., and Schrauwen, B. (2013). MACOP modular architecture with control primitives. *Frontiers in Computational Neuroscience*, 7(99):1–13. Available from: `http://dx.doi.org/10.3389/fncom.2013.00099`.

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

[42] 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*, pages 118–122.

[43] wyffels, F., Schrauwen, B., and Stroobandt, D. (2008). Stable output feedback in reservoir computing using ridge regression. In *International Conference on Artificial Neural Networks*, pages 808–817. Springer.