A comparative study of Reservoir Computing strategies for monthly time series prediction

F. Wyffels[a,*], B. Schrauwen[a]

[a]*Ghent University, Electronics and Information Systems Department, Sint-Pietersnieuwstraat 41, B-9000 Ghent, Belgium*

Abstract

A good prediction of the future enables companies and governments to plan their investments, production and other needs. The demand for good forecasting techniques motivates many researchers coming from a wide variety of fields to develop methods for time series prediction. Many of these techniques are very complex to apply and demand lots of computational effort to execute. As an answer to this, we propose the use of Reservoir Computing, a recently developed technique for efficient training of recurrent neural networks, for monthly time series prediction. We will explain how Reservoir Computing in its basic form can be applied to time series prediction. Additionally we will extend this approach with different Reservoir Computing strategies such as seasonal adjustment or a Reservoir Computing based voting collective approach. We will investigate the performance of all the proposed strategies and compare its prediction accuracy with the linear forecasting procedure build in the Census Bureau's X-12-ARIMA program and a Nonlinear Autoregressive model using Least-Squares Support Vector Machines.

Keywords: Reservoir Computing, forecasting, monthly time series

1. Introduction

The ability to make a good prediction of the future is advantageous in a broad range of applications. Companies and government organizations use medium- to short-term predictions for planning and operation of their businesses. Because of this great interest in time series prediction, a large number of researchers are working on time series prediction. These researchers come from a wide variety of fields and try to model the underlying process using linear techniques such as Box-Jenkins [1] and exponential smoothing [2] or non-linear techniques such as support vector machines [3] and neural networks [4]. Most of these techniques demand high-level user experience and a lot of computational effort.
If the user has no experience in the field, it is hard to select the right prediction methodology for a certain prediction task due to the number of available techniques and optimizations one can do. Thus there is the necessity to select a prediction methodology which can perform the task. In that spirit researchers started to organize time series prediction competitions which enables the comparison of prediction accuracy and computational efforts on different kinds of time series in the famous Santa Fe competition [5] or in other competitions such as the prediction of chaotic time series in the K.U. Leuven prediction competition [6], daily and monthly financial time series in the NN3 and NN5 competition [1], the CATS benchmark in the IJCNN2004 time series prediction competition [7], industrial time series in the ESTSP2007 competition [8] and time series from various sources in the ESTSP2008 competition [9].
Monthly time series often exhibit strong seasonality caused by factors such as weather, holidays, repeating promotions, as well the behavior of economic agents [10]. Thus, the question of how to cope with seasonality in time series is an important research topic. Most prediction approaches deal with seasonality by decomposition of the time series into a trend-cycle, seasonal and irregular components. These components are predicted independently and afterwards combined to get a good prediction of the original time series [11]. In [11] a comparative study for monthly aggregate retail sales forecasting using both linear and nonlinear techniques was presented. Their results suggested that the overall best method for retail sales forecasting is a neural network model with deseasonalized time series data.
In the domain of time series prediction neural network techniques are increasingly used. Particularly, recurrent neural networks are gaining success because they are ideal for such a temporal task. This kind of networks have internal feedback loops which gives them the ability to model temporal relationship of the time series explicitly within their internal states [12]. But gradient-based training algorithms have been known to suffer from local minima and demand high computational efforts [13]. Recently a novel technique for the efficient training of large recurrent neural networks has been introduced. Instead of training all the weights, the weights are initialized randomly and the desired function is implemented by a full instantaneous linear mapping of the neuron states. For this, standard linear regression methods can be used which eases the training process. When analog neurons are used, the method is referred to as Echo State Networks [14]. When spiking neurons are used, one often speaks of Liquid State Machines [15]. Commonly, they are referred to as Reservoir Computing [16]. Reservoir

*Corresponding author
Email addresses: Francis.wyffels@elis.UGent.be (F. Wyffels), Benjamin.Schrauwen@elis.UGent.be (B. Schrauwen)

[1] http://www.neural-forecasting-competition.com/

Manuscript draft accepted for publication in Neurocomputing: Special Issue ESTSP 2008

Computing has been successfully applied in a wide range of temporal tasks such as robotic localization [17], speech recognition [18, 19] and time series generation [20]. In 2007, Reservoir Computing in combination with seasonal adjustment and a voting collective approach outperformed all other methods in the NN3 competition for monthly financial time series prediction [21]. More recently, an approach combining Reservoir Computing and a decomposition approach based on wavelet decomposition was used for prediction of the time series provided in the ESTSP2008 competition [22] which led to average results in the competition. Additionally, reservoir computing showed outstanding performance in prediction of nonlinear chaotic time series. On a benchmark task of predicting a chaotic time series, accuracy was improved by a factor of 2400 over previous techniques [20].

Based on previous research and results on forecasting using Reservoir Computing, we do a comparative study of several Reservoir Computing strategies for monthly time series prediction. More specifically, we will address following research questions:

- Are Reservoir Computing based techniques suited for monthly time series prediction?
- Can we improve the prediction accuracy by the use of seasonal adjustment, increasing the reservoir size or using voting collectives?
- Are the nonlinear modeling abilities and the dynamical properties of Reservoir Computing beneficial in comparison to other techniques?

The outline of the paper is as follows: first we will describe how Reservoir Computing can be used for forecasting. Next, we review a number of strategies that can be combined with Reservoir Computing in order to improve the prediction accuracy. This is followed by a description of the time series we use for our empirical findings. After that, we present our results and discuss them. Finally our conclusions will be drawn and answers for our major questions will be stated.

2. Reservoir Computing for time series prediction

Reservoir Computing (RC) is a recently developed technique for the efficient training of recurrent neural networks. The technique is based on the use of a large, untrained dynamical system, the reservoir, which can be excited with one or more inputs. The desired output function is usually implemented by a linear memory-less mapping of the full instantaneous state of the dynamical system. Only this linear mapping is learned with commonly used standard linear regression techniques. Because of the nature of the task, generation of future time steps based on a learned history, we will focus on the use of RC for recursive prediction. This means that we will only consider systems which have the delayed output feedback as an input to the reservoir which is illustrated in Figure 1.

Throughout this work the reservoir is implemented by a large

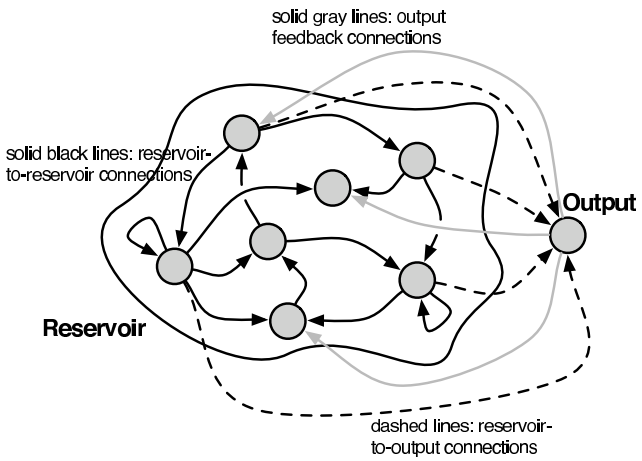

Figure 1: Schematic overview of Reservoir Computing used for recursive prediction. The system has only output feedback as an input. Only connections directed to the output nodes, denoted by dashed lines, are trained. Reservoir-to-reservoir connections and output feedback connections (represented by solid black lines and solid gray lines, respectively), are randomly created and kept fixed during training.

randomly connected recurrent neural network of sigmoid neurons. The neuron states and the output are update by the following equations:

$$\begin{aligned} \mathbf{x}[k+1] &= (1-\lambda)\mathbf{x}[k] \\ &\quad +\lambda \tanh\left(W^{\text{res}}_{\text{res}}\mathbf{x}[k] + W^{\text{res}}_{\text{out}}\mathbf{y}[k] + W^{\text{res}}_{\text{bias}}\right) \\ \widehat{\mathbf{y}}[k+1] &= W^{\text{out}}_{\text{res}}\mathbf{x}[k+1] + W^{\text{out}}_{\text{bias}}, \end{aligned} \tag{1}$$

where $\mathbf{x}[k+1]$ are the neuron states at time $k+1$ depending on the neuron states $\mathbf{x}[k]$ at previous time step k, the teacher forced output $\mathbf{y}[k]$ and a bias. One of the most important parameters, especially for a temporal task such as time series prediction, is the leak-rate λ with which the reservoir's dynamics, and thus the timescale at which the system operates, can be effectively tuned [14]. Before any data processing can begin, one has to first create the system's topology. All connections from the bias and output to the reservoir, denoted with $W^{\text{bias}}_{\text{res}}$ and $W^{\text{out}}_{\text{res}}$ respectively, are drawn from a normal distribution with zero mean and of which the variance is a parameter to tune. For construction of the weight matrix $W^{\text{res}}_{\text{res}}$, which determines connections within the reservoir, weights are drawn from a normal distribution with zero mean and variance 1 and a large part is set to zero according to the connection fraction. The randomly connected matrix $W^{\text{res}}_{\text{res}}$ is usually rescaled such that the largest eigenvalue, spectral radius, is near to 1. This causes the created system to operate at the edge of stability where its processing power is greatest [23]. Note that in the Reservoir Computing setup we use, the effective spectral radius introduced in [24], is also dependent on the bias and the weights from and to the output of which the weights to the output are unknown during the construction of the topology. Thus, the spectral radius as defined in this work loses it significance as a measure for stability and it has been shown previously that you can go even beyond 1 without unstable behavior [25].

Only the connections to the output, denoted by $W_\star^{\text{out}}$, are changed during training in order to learn the desired output function. Because only the output weights are changed, training is extremely fast which can be an additional benefit in comparison with other methods. Additionally, the training of the reservoir system doesn't suffer from local optima like other methods based on neural networks do. When validating or testing the system, the teacher forced output feedback $\mathbf{y}[k]$ in Equation 1 is replaced by the actual output $\widehat{\mathbf{y}}[k]$, which is known as recursive prediction.

We now describe the full process of modeling time series and prediction future data points. We always start with normalizing the given time series. This is done by removing the mean and dividing by the standard deviation of the time series, i.e. makes them Z-scores. This avoids that all neurons get saturated and thus lose processing power. Next, the time series is divided into two parts. The first (largest) part is used for training and optimization of the meta-parameters, the second part (equal in length to the desired prediction horizon) is kept unknown and is only used for testing.
After construction of the reservoir topology, the reservoir system is ready to process the training part of the time series using teacher forcing. The neuron states are updated using Equation 1 and collected. Because we use a dynamical system, it takes some time before the full effects of the teacher forced input is visible in the reservoir states. Therefore, the initial states containing the transient effects are discarded which is known as warmup drop. When all training data is processed the system can be trained by using standard linear regression techniques. A 4-fold cross-validation scheme is used to optimize the meta-parameters, more specific the leak-rate, bias and output feedback scale. Because the reservoir system needs to be initialized, we make sure that there is overlap in de subsets of the training and validation data used for the cross-validation so that almost no data is lost for the warmup drop. The overlapping data points are thus only used for initialization of the reservoir system. During validation, our system is used in recursive prediction mode, generating future data points. The Normalized Mean Squared Error (NMSE) is used as an error metric:

$$\text{NMSE} = \frac{1}{N}\frac{\sum_{k=1}^{N}(\mathbf{y}[k]-\hat{\mathbf{y}}[k])^2}{\sigma_{\mathbf{y}}^2}, \quad (2)$$

with $\mathbf{y}[k]$ the desired predictions and $\hat{\mathbf{y}}[k]$ the outcome of our system for a prediction horizon N.
After optimization of the meta-parameters with 4-fold cross-validation, the system is retrained using the full training and validation part of the time series. Finally, the system is used for recursive prediction of the (unknown) test part of the data. After undoing the normalization, the prediction accuracy is evaluated. For this, based on discussion in [26], we will use multiple error metrics such as the earlier mentioned NMSE and the Symmetric Mean Absolute Percentage Error (SMAPE):

$$\text{SMAPE} = \frac{100}{N}\sum_{k=1}^{N}\frac{|\mathbf{y}[k]-\hat{\mathbf{y}}[k]|}{(\mathbf{y}[k]+\hat{\mathbf{y}}[k])/2}, \quad (3)$$

in which we assume $\mathbf{y}[k]$ and $\hat{\mathbf{y}}[k]$ to be positive.

3. Prediction strategies

Although RC in its standard setup performs well in a broad range of applications, additional techniques have already been introduced to improve its results. In this Section we provide some well known and commonly used strategies in order to improve the performance of RC techniques.

3.1. Influence of reservoir size and regularization

RC techniques are based on the use of a large untrained dynamical system. Previous results showed that increasing reservoir size leads to larger memory capacity [27, 28]. However larger networks, thus more complex models, increase the danger of overfitting leading to poor results on the test set. In [29], ridge regression has been found a good candidate to avoid overfitting when output feedback is necessary. Later, in [30] ridge regression was compared with other regularization techniques such as pruning. But in none of these papers, time series prediction was considered. In this work we extend the previous work by investigating the use of ridge regression for time series prediction more closely.
When using ridge regression instead of standard linear regression an additional term, dependent on the readout weights $W_\star^{\text{out}}$ is added to the cost function [31]:

$$J_{\text{ridge}}(W_\star^{\text{out}}) = \frac{1}{2}\left(CW_\star^{\text{out}}-D\right)^T\left(CW_\star^{\text{out}}-D\right) + \frac{1}{2}\lambda\left\|W_\star^{\text{out}}\right\|_2^2, \quad (4)$$

in which matrix C consists of the concatenation of all inputs to the readout including the collected reservoir states and the bias and matrix D contains the desired outputs.
Ridge regression uses an additional parameter, the regularization parameter λ, which needs to be optimized. When applied well (and the regularization parameter is optimized correctly), ridge regression keeps the output weights small, regularizes the trained trajectory in state-space and gives our system good generalization capabilities. Additionally, it stabilizes the output when using output feedback. In this paper, we will investigate the influence of the reservoir size and regularization on the prediction accuracy.

3.2. Voting collectives

Previous work showed that RC techniques can classify speech data very well if many small reservoirs are combined in a voting collective [32]. By keeping the reservoir size small, the reservoir dynamics change drastically, increasing significantly the variance of the results of the many systems. This makes them suited as a weak learner. Additionally, small reservoirs have intrinsic regularization properties which makes it easier to apply. The idea [32] is that every classifier independently generates a result based on randomly constituted features. The mean of the individual votes is taken, hoping that this averages out the fluctuations that are due to the single classifiers' biases.

In [21] this approach is used for time series prediction. In this paper we will investigate whether this approach is beneficial or not for the time series we consider.

3.3. Seasonal decomposition

One of the current main shortcomings is that RC methods tend to be sensitive to a small temporal range [33]. When data consists of highly different temporal domains, or if interference can occur inside the network, performance can degrade rapidly. The importance of this is also mentioned in [32, 34]. This issue can be partially solved by the use of leaky integrator neurons [14] or band-pass neurons [35, 33]. But, in the domain of forecasting, another technique, known as seasonal decomposition, could be beneficial. Monthly time series exhibit strong seasonal components. How to deal with seasonality is an important research question and its application has shown a large increase in the prediction accuracy when using e.g. neural networks [11]. Most approaches deal with seasonality by decomposition of the time series Y into a trend-cycle C, seasonal S and irregular I components. This can be done by means of the general additive decomposition model:

$$Y = C + S + I. \tag{5}$$

The trend-cycle includes long-term trends and movements including consequential turning points. When the seasonal fluctuations vary proportionally with the level of the time series, which is typical for economical time series, one can better use a multiplicative decomposition model:

$$Y = CSI. \tag{6}$$

This is also the default seasonal adjustment mode for the Census X-12-ARIMA program [36] which we will use for seasonal adjustment. This program gives us the three components of a time series Y. Instead of modeling the original time series Y as described in Section 2, we will model the trend-cycle, seasonal and irregular component individually. After prediction of each component we can try to reconstruct the future of the original time series by applying equation 6. Based on the described issues of RC techniques with respect to multiple timescales, it can be expected that seasonal decomposition will improve the prediction accuracy. This strategy was also used in successful results of the NN3 competition [21].

4. Prediction of monthly time series

In this Section we describe how to apply RC and the three strategies for monthly time series prediction. In this comparative study we will investigate the effectiveness of RC on several monthly time series and the benefits of the different proposed strategies. We will compare the accuracy of the predictions with results from an ARIMA model provided by the Census X-12-ARIMA program. Additionally we compare with a NAR model estimated using Least-Squares Support Vector Machines (LS-SVMs) [37].

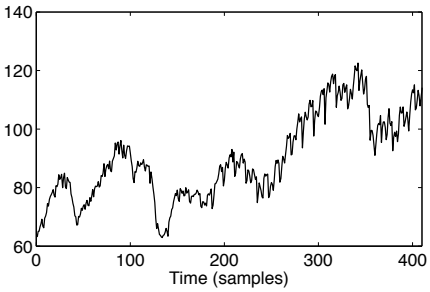

Figure 2: Example of a monthly time series: production of machinery in the USA.

4.1. Methodology

In this study we used five datasets coming from industry. The first time series is the monthly electricity production in Australia starting in September 1959 and ending in August 1995. All four other time series were provided by the US Federal Reserve Board and concern the monthly evolution of plastic and rubber goods production, glass goods production, metal goods production and machinery production. These datasets start in January 1972 and end in December 2007. As an illustration, one of the monthly time series (production of machinery in the USA) is visualized in Figure 2. One can see that the time series include seasonal effects. For all time series the prediction horizon is 24 months. This means that we used the last 24 samples of the provided time series for testing only and they are left out while training (and optimizing) our system.

During the first experiment we investigated the influence of the reservoir size with and without regularization. Therefore we applied Reservoir Computing as described in Section 2 using the Reservoir Computing Toolbox v2.0[2]. The reservoir size varied from 10 to 1000 neurons. Applying RC involves tuning of quite a number of meta-parameters changing the characteristics of the reservoir system. Fortunately not all of them are crucial. The performance of a reservoir system using analog neurons is largely independent of the sparsity of the network [38]. Therefore we will set all connection fractions (fraction of neurons connected to each other, fraction of output connected to the neurons, fraction of neurons that has a bias) to an arbitrary value of 25 %. The spectral radius is set to 1 based on what we previously said in Section 2. We also tried other values including values greater than 1, but this gave no significant improvement. If the spectral radius was chosen too large, prediction accuracy degraded strongly indicating the reservoir was beyond the edge of stability. Other parameters have greater influence on the dynamics of the reservoir system and prediction accuracy. These parameters include the feedback weights, the leak-rate and bias. For each time series we processed with RC, the leak-rate was optimized using 2-level grid-searching in 10 logarithmically spaced values between 0 and 1. This is driven

[2]On http://reslab.elis.ugent.be/rctoolbox, a toolbox consisting of functions that allow the user to easily set up datasets, reservoir topologies, experiment parameters and to easily process results of experiments, can be downloaded.

by the fact that the cut-off frequencies of the low-pass filters you get by using leaky neurons are also logarithmically spaced [35]. The feedback weights are set randomly to w or $-w$, where w is grid-searched in two levels in 10 logarithmically spaced values. Additionally, we tested wether adding bias to the neurons was necessary or not. If bias was added to the neurons, the bias weights are drawn from a normal distribution with zero mean and variance 1. Optimization was done using a 4-fold cross-validation scheme.

When regularization was done, ridge regression was used. This introduces an additional parameter, the regularization parameter, which needs to be optimized. The regularization parameter was line searched in the range $10^{1:-0.2:-8}$ [3] using 4-fold cross-validation.

After the first experiment we considered a reservoir size of 500 neurons using ridge regression as a regularization technique as a basis of comparison for the standard RC approach in the next experiments.

During the second experiment we made use of a RC based voting collective system. We constructed a system consisting of 500 relatively small reservoirs of 75 neurons. Although using relatively small reservoirs, which have intrinsic regularization properties, we applied ridge regression as an additional regularization technique for each trained reservoir system. Each system was trained separately but the bias, leak-rate en feedback weights were optimized globally and chosen equally for each reservoir. For the final result of the voting collective, the outcome (eg. predicted time series) of all the small entities were averaged.

For the third experiment we investigated the use of seasonal decomposition. Therefore each of the provided time series was processed by the Census X-12-ARIMA seasonal adjustment program resulting in a trend-cycle, seasonal and irregular component. For the production of machinery time series (see Figure 2), the three components: trend-cycle, seasonal and irregular components are shown in Figure 3, Figure 4 and Figure 5 respectively. As said before we only used multiplicative seasonal decomposition. Both, RC in its standard form and the voting collective approach were used to forecast each component separately. Afterwards the predictions of all components were recombined resulting in predictions for the original time series. When applying the voting collective approach, the predictions of the components were averaged separately before reconstruction.

Because we work with a non-deterministic technique which involves random generation of the neuron weights, we redo each experiment 50 times which will give us a better idea of the results one can expect.

As a basis for comparison, we constructed both, a linear and nonlinear model for the time series. The linear model was provided by the Census X-12-ARIMA program which uses seasonal decomposition. For the NAR model, we applied LS-SVM on the normalized time series (without using seasonal decomposition) using a RBF kernel. A leave-one-out cross-validation

[3] Which means that the regularization parameter ranged from 10 to 10^{-8}, each time by decreasing the exponent with 0.2.

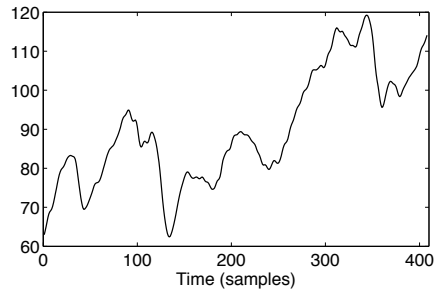

Figure 3: Production of machinery: trend-cycle obtained with the Census X-12-ARIMA program.

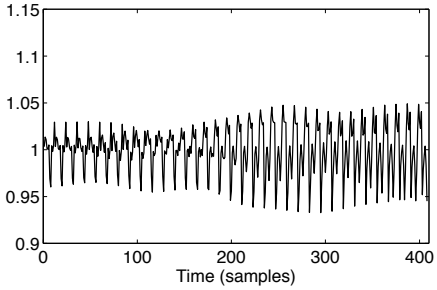

Figure 4: Production of machinery: seasonal component obtained with the Census X-12-ARIMA program.

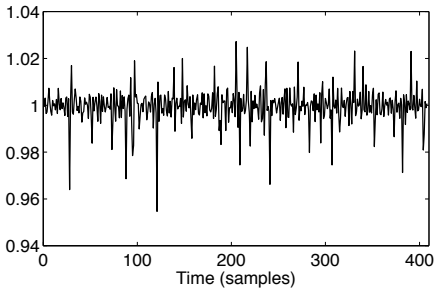

Figure 5: Production of machinery: irregular component obtained with the Census X-12-ARIMA program.

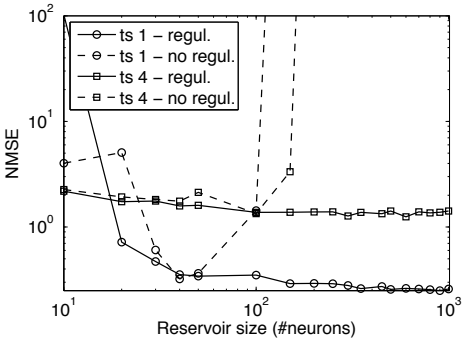

Figure 6: Prediction accuracy for two time series, time series 1: electricity production and time series 4: metal goods production, in function of reservoir size with (solid line) and without (dashed line) using regularization.

scheme with two-level grid-search was used to select the hyper-parameters. The size of the time window was optimized in the range 6 to 84 in steps of 6. This is motivated by the specifications of the data: monthly time series. We did not apply any algorithm to actively select the regressors.

4.2. Results

In Figure 6 the results of our first experiment are summarized. The NMSE on the test set in function of the reservoir size, with and without using regularization, can be seen for two of the five time series. The same behavior could be seen for the other time series but is not shown in order to not overload the figure. We see that without using regularization the prediction accuracy is strongly dependent on the size of the reservoir. For larger reservoirs, the model complexity increases leading to over-fitting the training data and thus poor regularization, while small reservoirs show intrinsic regularization properties. On the other hand, when using ridge regression, prediction accuracy increases asymptotically for increasing reservoir size. Based on these results we decide to apply ridge regression in all further experiments.

In Table 1 we summarized the results for all time series using all the strategies presented in Section 3: a single reservoir setup with 500 neurons, a voting collective setup using 500 reservoirs of 75 neurons, a single reservoir setup with 500 neurons after seasonal decomposition and a voting collective setup using 500 reservoirs of 75 neurons after seasonal decomposition. Additionally, the prediction results of the ARIMA model and the NAR model are given.

If we compare the results of the voting collective setup with these of the single reservoir setup, we can see that they are competitive with each other. However, we can see that the voting collective setup show low variance in prediction accuracy which is beneficial in comparison with the single reservoir setup were results are dependent on the generated reservoir. This comes at a cost: computational efforts are much higher than for the single reservoir approaches. This is mainly due to the cost of optimizing the regularization parameter of each of the 500 reservoirs. To illustrate this, we did time measurements for all the reservoir computing strategies. The computational cost is shown relatively against the standard single reservoir approach in Table 2.

Table 2: Computational efforts

single	single decomp.	voting	voting decomp.
1	3	9.71	29.13

Considering this, one would more likely choose the single reservoir computing approach. However based on the results of the first experiment, in which we investigated the influence of the reservoir size and the regularization, one could reconsider the necessity of using ridge regression since small reservoirs show intrinsic regularization properties. One could decide to optimize the global reservoir size which is less costly in terms of computational demands than optimizing the regularization parameter. This might be an interesting research topic for future work.

From Table 1 we learn that seasonal decomposition improves greatly the prediction accuracy in both, a single and voting collective reservoir setup. This is also illustrated in Figure 7 in which we present the prediction for the metal goods production time series using all strategies presented earlier. We see that the RC strategies without decomposition are not able to capture the seasonality while strategies using decomposition can, not surprisingly, capture the seasonality. Remarkably, the NAR model, which does not use seasonal decomposition, shows seasonal effects in the prediction of the time series. However, the results of the NAR model are comparable with the results from the RC models without decomposition. As we learn from Table 2, seasonal decomposition comes at a cost: the computational efforts increase with factor three since the three components have to be processed separately. However, in our opinion the increase of performance is worth the increase in computational demands.

Of all strategies the voting collective setup after seasonal decomposition shows best overall prediction accuracy. Next in line, the single reservoir setup with regularization shows good performance which proves that decomposition increases greatly prediction accuracy when using RC. If we compare these results with the results from the ARIMA and NAR model we can conclude that RC is an interesting technique to consider when one wants to do monthly time series prediction.

5. Conclusions

It is hard to find the right prediction method for a given application if not a prediction expert. Many time series prediction competitions try to seek an answer to the need for a good general purpose prediction strategy. In this paper we discussed how Reservoir Computing, an efficient training method for recurrent neural networks, can be used for time series prediction, which has previously proven its outstanding performance in prediction of chaotic nonlinear time series. We applied Reservoir Computing for recursive prediction of monthly time series. We investigated the influence of reservoir size and regularization. Additionally, we tried to improve its prediction accuracy using

Table 1: Prediction results

Time series	error metric	single	single decomp.	voting	voting decomp.	NAR (LS-SVM)	X-12-ARIMA
electric. prod.	NMSE (STD)	0.2567 (0.0339)	0.1871 (0.0234)	0.2756 (0.0048)	**0.1687 (0.0012)**	0.5955	0.1703
	SMAPE (STD)	2.1103 (0.1386)	1.8056 (0.1270)	2.1349 (0.0155)	1.7032 (0.0072)	3.3709	**1.5690**
plastics goods	NMSE (STD)	2.6905 (0.4482)	1.2110 (0.1067)	2.7616 (0.0176)	**1.1914 (0.1449)**	1.9937	2.8891
	SMAPE (STD)	2.4002 (0.5580)	**1.5204 (0.0788)**	2.4432 (0.0095)	1.5375 (0.1329)	2.1054	2.6059
glass goods	NMSE (STD)	2.4100 (1.7059)	0.8987 (0.1914)	1.8095 (0.0460)	**0.8378 (0.0038)**	2.2269	0.9161
	SMAPE (STD)	2.2947 (0.8162)	1.3942 (0.1625)	2.0182 (0.0328)	**1.3363 (0.0133)**	2.3349	1.3549
metal goods	NMSE (STD)	1.4162 (0.4248)	**0.6432 (0.1464)**	1.3646 (0.0232)	0.6726 (0.0064)	1.3229	0.6864
	SMAPE (STD)	3.8187 (0.7849)	**2.4923 (0.3570)**	3.7566 (0.0524)	2.5779 (0.0161)	3.927	2.7872
machinery	NMSE (STD)	3.8303 (0.896)	2.1853 (0.4464)	4.2581 (0.0302)	2.1640 (0.0450)	**1.2922**	2.3244
	SMAPE (STD)	4.5261 (0.5651)	3.4308 (0.3822)	4.8412 (0.0192)	3.4485 (0.0352)	**2.4496**	3.4288

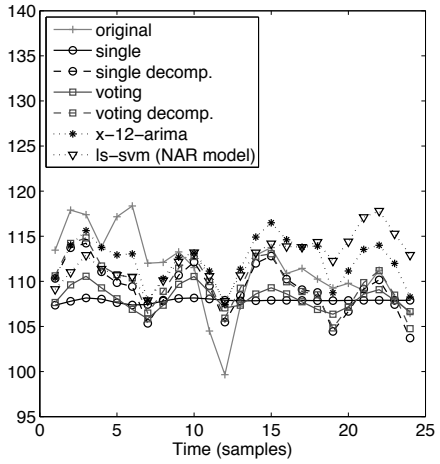

Figure 7: Forecasts of a monthly time series (metal goods production) using six different prediction strategies: single reservoir setup with 500 neurons, single reservoir setup with 500 neurons after seasonal decomposition, voting collective reservoir setup, voting collective reservoir setup after seasonal decomposition, X-12-ARIMA modeling and NAR modeling with LS-SVMs.

different strategies such as seasonal decomposition and a voting collective reservoir setup.

We showed that Reservoir Computing shows competitive results for monthly time series prediction. By increasing the reservoir size, the prediction accuracy of the single reservoir approach can be improved on the condition that regularization is applied correctly. In other work, researchers showed that Reservoir Computing techniques have difficulties to handle multiple timescales in time series. We showed that the problems with multiple timescales can be solved in the field of time series prediction by decomposition of the time series.

The results of the single reservoir setup are dependent on the generated reservoir. This can be seen in the rather large variance in the results. In this work we demonstrate that it is preferable to use a voting collective approach to overcome this problem. This is because having a bad prediction due to a poorly generated reservoir is decreasing significantly due to the averaging out of many votes. This, however comes at the cost of higher computational demands.

Overall we can say that Reservoir Computing strategies can be seen as a viable tool for state-of-the-art monthly time series prediction. It is preferable to use seasonal decomposition to increase prediction accuracy. When high accuracy and reliable results are necessary, a voting collective approach is recommended.

6. Acknowledgements

This research is partially funded by FWO Flanders project G.0088.09 and the Photonics@be Interuniversity Attraction Poles program (IAP 6/10), initiated by the Belgium State, Prime Minister's Services, Science Policy Office.

References

[1] G. E. P. Box, J. Jenkins, Time Series Analysis: Forecasting and Control, San Francisco: Holden-Day, 1976.

[2] P. R. Winters, Forecasting Sales by Exponentially Weighted Moving Averages, Management Science 6 (1960) 324–342.

[3] K. R. Mueller, A. J. Smola, G. Rtsch, B. Scholkopf, J. Kohlmorgen, V. Vapnik, Predicting time series with support vector machines, in: Proceedings of the International Conference on Analog Neural Networks, 1997.

[4] E. Wan, Time Series Prediction by Using a Connectionist Network with Internal Delay Lines, in: Time Series Prediction, Addison-Wesley, 195–217, 1994.

[5] A. S. Weigend, N. A. Gershenfeld (Eds.), Time Series Prediction: Forecasting the Future and Understanding the Past, Addison-Wesley, 1994.

[6] J. McNames, J. A. K. Suykens, J. Vandewalle, Winning entry of the K.U. Leuven time series prediction competition, International Journal of Bifurcation and Chaos 9 (1999) 1485–1500.

[7] A. Lendasse, E. Oja, O. Simula, M. Verleysen, Time Series Prediction Competition: The CATS Benchmark, Neurocomputing 70 (2007) 2325–2329.

[8] A. Lendasse (Ed.), Proceedings of the First European Symposium on Time Series Prediction, 2007.

[9] A. Lendasse (Ed.), European Symposium on Time Series Prediction, 2008.

[10] G. P. Zhang, M. Qi, Neural network forecasting for seasonal and trend time series, European journal of operational research 160 (2005) 501–514.

[11] C. Chu, G. P. Zhang, A comparative study of linear and nonlinear models for aggregate retail sales forecasting, Internation journal of production economics 86 (2003) 217–231.

[12] E. W. Saad, D. V. Prokhorov, D. C. Wunsch II, Comparative Study of Stock Trend Prediction Using Time Delay, Recurrent and Probabilistic Neural Networks, IEEE Transactions on neural networks 9 (6) (1998) 1456–1470.

[13] X. Cai, N. Zhang, G. K. Venayagamoorthy, D. C. Wunsch II, Time series prediction with recurrent neural networks trained by a hybrid PSO-EA algorithm, Neurocomputing 70 (2007) 2342–?2353.

[14] H. Jaeger, The "echo state" approach to analysing and training recurrent neural networks, Tech. Rep. GMD Report 148, German National Research Center for Information Technology, 2001.
[15] W. Maass, T. Natschläger, H. Markram, Real-time Computing without stable states: A New Framework for Neural Computation Based on Perturbations, Neural Computation 14 (11) (2002) 2531–2560.
[16] D. Verstraeten, B. Schrauwen, M. D'Haene, D. Stroobandt, An experimental unification of reservoir computing methods, Neural Networks 20 (2007) 391–403.
[17] E. A. Antonelo, B. Schrauwen, D. Stroobandt, Event detection and localization for small mobile robots using reservoir computing, Neural Networks 21 (2008) 862–871.
[18] D. Verstraeten, B. Schrauwen, D. Stroobandt, J. Van Campenhout, Isolated word recognition with the Liquid State Machine: a case study, Information Processing Letters 95 (6) Bohte, S.M. and Kok J.N.
[19] M. D. Skowronski, J. G. Harris, 2007 Special Issue: Automatic speech recognition using a predictive echo state network classifier, Neural Networks 20 (3) (2007) 414–423.
[20] H. Jaeger, H. Haas, Harnessing nonlinearity: predicting chaotic systems and saving energy in wireless telecommunication, Science 308 (2004) 78–80.
[21] H. Jaeger, Background information: Jacobs University "Smart Systems" seminar wins international financial time series competition, 2007.
[22] F. Wyffels, B. Schrauwen, D. Stroobandt, Using reservoir computing in a decomposition approach for time series prediction, in: A. Lendasse (Ed.), European Symposium on Time Series Prediction, 149–158, 2008.
[23] R. A. Legenstein, W. Maass, Edge of Chaos and Prediction of Computational Performance for Neural Microcircuit Models, Neural Networks (2007) 323–333.
[24] M. Ozturk, D. Xu, J. Principe, Analysis and design of Echo State Networks, Neural Computation 19 (2007) 111–138.
[25] D. Verstraeten, B. Schrauwen, On the Quantification of Dynamics in Reservoir Computing, in: Lecture Notes in Computer Science, 2009.
[26] J. S. Armstrong, Principles of forecasting, Kluwer Academic Publishers, 2001.
[27] H. Jaeger, Short term memory in echo state networks, Tech. Rep. GMD Report 152, German National Research Center for Information Technology, 2002.
[28] O. L. White, D. D. Lee, H. Sompolinsky, Short-Term Memory in Orthogonal Neural Networks, Physical Review Letters 94 (14) (2004) 148102.
[29] F. Wyffels, B. Schrauwen, D. Stroobandt, Stable output feedback in reservoir computing using ridge regression, in: Proceedings of the International Conference on Analog Neural Networks, 2008.
[30] X. Dutoit, B. Schrauwen, J. Van Campenhout, D. Stroobandt, H. Van Brussel, M. Nuttin, Pruning and regularization in reservoir computing, Neurocomputing 72 (2009) 1534–1546.
[31] A. Hoerl, R. Kennard, Ridge regression: Biased estimation for nonorthogonal problems, Technometrics 12 (1970) 55–67.
[32] H. Jaeger, M. Lukosevicius, D. Popovici, Optimization and Applications of Echo State Networks with Leaky Integrator Neurons, Neural Networks 20 (2007) 335–352.
[33] F. Wyffels, B. Schrauwen, D. Verstraeten, D. Stroobandt, Band-pass reservoir computing, in: Z. Hou, N. Zhang (Eds.), Proceedings of the International Joint Conference on Neural Networks, Hong Kong, 3203–3208, 2008.
[34] B. Schrauwen, J. Defour, D. Verstraeten, J. Van Campenhout, The introduction of time-scales in Reservoir Computing, applied to isolated digits recognition, in: Proceedings of the International Conference on Artificial Neural Networks, 2007.
[35] U. Siewert, W. Wustlich, Echo-state Networks with Band-pass Neurons: Towards Generic Time-Scale-Independent Reservoir Structures, Tech. Rep., PLANET Intelligent Systems GmbH, 2007.
[36] D. F. Findley, B. C. Monsell, W. R. Bell, M. C. Otto, B. C. Chen, New capabilities and methods of the X-12-ARIMA seasonal-adjustment program, Journal of Business and Economic Statistics 16 (1996) 127–152.
[37] J. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor, J. Vandewalle, Least Squares Support Vector Machines, World Scientific, Singapore, 2002.
[38] B. Schrauwen, L. Buesing, R. Legenstein, On Computational Power and the Order-Chaos Phase Transition in Reservoir Computing, in: D. Koller, D. Schuurmans, Y. Bengio, L. Bottou (Eds.), Advances in Neural Information Processing Systems, vol. 21, 1425–1432, 2009.