Preprocessing Seasonal Time Series for Improving Neural Network Predictions

Francesco Virili\(^1\) and Bernd Freisleben\(^2\)

\(^1\)Department of Business Information Systems, University of Siegen, Hölderlinstr. 3, D–57068 Siegen, Germany, E-Mail: francesco.virili@acm.org

\(^2\)Department of Electrical Engineering & Computer Science, University of Siegen, Hölderlinstr. 3, D–57068 Siegen, Germany, E-Mail: freisleb@informatik.uni-siegen.de

Abstract

In many applications of neural networks to economic time series prediction tasks, the statistical properties of the time series are not taken into account. In this paper, it will be shown that the seasonality of time series will have a major impact on the prediction performance, and appropriate data preprocessing techniques must be applied to take care of its effects. We illustrate the influence of several preprocessing methods on the quality of forecasting the mortgage market loans purchased in the Netherlands. The results demonstrate that using an appropriate preprocessing technique based on the properties of the time series can have a valuable effect on the neural network outcome.

1 Introduction

The benefits - and the drawbacks - of using neural networks for economic forecasting have already been discussed in a variety of application areas, see e.g. [3, 19, 21, 22]. Typically, a supervised learning algorithm in a multi-layer feedforward network is used to learn the mapping between the input and output data in order to discover the implicit rules governing the movement of particular factors and predict their continuation in the future.

The inputs to the neural networks are either the raw time series values, or some type of normalization is applied to transform the inputs into the domain of the (sigmoid) output units. In contrast to conventional forecasting approaches, there is rarely an effort to explicitly investigate the statistical properties of the time series, such as nonstationarity and seasonality, before the training phase is started. However, the presence of nonstationarity or seasonality may have a large impact on the prediction quality, and consequently should be addressed by applying suitable preprocessing techniques.

In this paper, we investigate the effects of seasonality on neural network predictions. Appropriate data preprocessing methods will be discussed, and experimental results for the time series of the Dutch mortgage loan market (see Fig. 1) will be presented. The results produced by a standard backpropagation network and a linear autoregressive model indicate that applying the right preprocessing technique significantly improves the performance of the forecasts.

The paper is organized as follows. In Section 2, the problem of seasonality in time series is discussed, and methods to deal with seasonality are explained. Section 3 presents the properties of the empirical investigation conducted as part of this paper. In Section 4, the results of the experiments are given. Section 5 concludes the paper and outlines areas for future research.

2 Seasonality

The existence of seasonal patterns is very frequent in economic time series, and consequently, there is a growing interest in a number of issues related to seasonality, such as modelling changing seasonal patterns, testing for seasonal unit roots, and analyzing the effects of seasonal adjustment methods [7, 9].

Systematic intra-year seasonal fluctuations in economic time series can have different origins ([9], p. 1): some can be (at least partially) deterministic, such as cyclical climatic effects, but some may be mainly caused by the behaviour of economic agents, on the basis of a multiplicity of possibly unobserved factors, and may therefore not be constant and even be stochastic. We may distinguish at least three approaches in modelling seasonal time series: (a) models based on seasonal dummies (which imply a deterministic seasonal pattern)[10], (b) the Box-Jenkins seasonal ARIMA models [4] (when the underlying process is seasonal and stochastic), or (c) the more complex models for changing seasonality, such as the
PAR (Periodic Autoregressive) model [9]. Instead of explicitly modelling seasonality, an alternative approach is trying to remove it via a seasonal adjustment procedure, such as the widely used Census X-11 program (see [9], section 2.2 and references therein).

Neural networks have been shown to be universal function approximators [16], and in principle they are able to fit seasonal patterns of any kind. For example, they have already been used by Franses and Draisma [11] to detect changing seasonality using the output of the hidden layer units in a multi-layer network. Nevertheless, in applications of artificial neural networks for univariate time series forecasting, the careful identification of a seasonal pattern and the use of an appropriate preprocessing technique to remove it could be useful to better understand the underlying process, possibly leading to improvements of the predictions. On the other hand, using an inappropriate preprocessing stage in the presence of more complex seasonal patterns could result in severe distortions of the original time series, as shown for linear models applied to a large class of widely used US quarterly macroeconomic series in [13]. Thus, if seasonality is correctly specified and there is an appropriate choice of preprocessing, neural network predictions are likely to improve.

A useful technique to detect seasonal patterns in time series is spectral analysis; for a sound and very readable introduction on the topic, see [5], chapters 6 and 7; see also [15], chapter 2. With spectral analysis we point the attention to the (eventually) cyclical components of a time series, in order to discover whether they exist, and what is their frequency, relative importance and contribution in explaining the variance of the time series. As demonstrated by the work of Wiener and others [15], any discrete stationary process measured at unit intervals may be regarded, loosely speaking, as a linear combination of orthogonal sinusoidal terms. We could be interested in investigating the time series in this form, that is in the frequency domain, to discover whether its variance is determined by all the infinitely possible components in the same proportion, (one example being a pure white noise series) or whether there exists some component which has a higher relative importance. For this purpose, we compute the autocovariance function of \( X_t \), that is \( \gamma(k) = E\{ [X_t - E(X_t)] [X_{t+k} - E(X_{t+k})] \} \), where \( E \) is the expectation operator. The autocovariance function gives a measure of the mutual dependency of \( X_t \) and \( X_{t+k} \); \( \gamma(k) = 0 \) if they are correlated. In a covariance stationary time series, \( \gamma(k) \) will not depend on time. Notice that \( \gamma(0) = \text{Var}(X) \).

It is possible to show that for any stationary stochastic process with autocovariance function \( \gamma(k) \), there exists a monotonically increasing function \( F(\omega) \), called the power spectral distribution function such that \( \gamma(k) = \int_0^\infty \cos(\omega k) dF(\omega) \), where \( \omega \)
Figure 2: The periodogram of the target time series $\ln(MV)$ (bold line), compared with that of the sinusoidal time series described in the text, rescaled by a factor of 1/10: the correspondence of the peaks suggests the presence of periodic components in MV at the same frequencies as the artificial time series: 1/12, 2/12, 3/12, 4/12 and 5/12.

denotes a particular frequency in radians per unit time, within the range $(0, \pi)$. For $k = 0$, since $\cos(0) = 1$, it follows that $\gamma(0) = \int_0^\pi dF(\omega) = F(\pi)$ so that $F(\pi) = Var(X)$.

The fact that $F(\omega)$ at the upper extreme of the frequency range is equal to the total variance of the time series suggests a direct physical interpretation: $F(\omega)$ measures the total contribution to variance of cycles with frequencies in the range $(0, \omega)$. Its derivative, $f(\omega) = \frac{dF(\omega)}{d\omega}$, is the so called power spectral density function, or spectrum; it represents the contribution to the variance of components with frequencies in the range $(\omega, \omega + d\omega)$.

The power spectrum of a seasonal time series typically has very evident peaks at the fundamental seasonal frequency and sometimes at the harmonics (see [15], pg. 67). Fig. 2 shows the power spectrum (or periodogram) of the MV (Mortgage Volume) series preprocessed with a natural logarithm transformation (bold line), compared with the periodogram of an artificial time series generated by adding 5 sines with the following periods: 12, 12/2, 12/3, 12/4 and 12/5 (thin line). In the two power spectra, the peaks at frequencies of 1/12, 2/12, 3/12, 4/12 and 5/12 of the MV series have a correspondence in the sinusoidal time series. Although their magnitude is different (in Fig. 2 the periodogram of the sinusoidal time series is rescaled by a factor of 1/10), this suggests a fundamental frequency of 1/12 (annually) and possibly harmonics at 2, 3, 4, 5 oscillations per year. The low frequency peak in the MV spectrum suggests that the series could be nearly nonstationary, as confirmed by the visual inspection, and that a seasonal unit root could be present [17].

The choice of an appropriate preprocessing technique may be based on performing the following actions: a) correcting the increasing variance of the time series and the skewness of its distribution; b) removing the seasonal pattern; c) processing the low frequency component as a possible indicator of a degree of nonstationarity. Consequently, we may use the natural logarithm transformation to correct the distribution, then calculate the first differences to correct nonstationarity and finally we could correct the seasonal pattern using a moving average of the last 12 monthly observations. It is also possible to apply the yearly seasonal differences $y_t = x_t - x_{t-12}$, instead of the first differences plus the moving average. The two methods are substantially equivalent, since the last 12 months moving average of the first differenced series $(x_{t-11} - x_{t-12} + x_{t-10} - x_{t-11} + x_{t-1} - x_{t-2} + x_t - x_{t-1}) \frac{1}{12}$ is equal to $(x_t - x_{t-12}) \frac{1}{12}$. Therefore, we simply use the seasonal differences, knowing that they can also be interpreted as a moving average of the differenced series. A visual analysis of the periodogram of the seasonal differenced
series shows the elimination of the seasonal pattern, without noticeable distortions.

The final series can be viewed alternatively as the result of applying a smoothing filter to the series of the first differences or as the series of the seasonal differences \((x_i - x_{i-12})\). In both cases we have an implicit choice for a stochastic seasonality pattern in which the original series was nonstationary and seasonal and the preprocessed series is stationary. Sometimes, as suggested e.g. by [10], we should take into consideration the alternative of a deterministic seasonal pattern, which could be better approximated using seasonal dummies. In that case using seasonal differences instead of the more appropriate raw values + seasonal dummies could result in a biased model, due to the over differenced series. Some preliminary analysis in this direction did not suggest this conclusion in our case. However, further research may be devoted to testing this hypothesis.

3 Empirical Investigation

In order to test the effects of the preprocessing methods on the prediction performance, a simple neural network architecture was considered. The same set of training parameters was used in all experiments. The only parameter differing was the length of the training phase, since it is based on the test set error that can vary with the preprocessed data. The neural network model used in the experiments was a standard single hidden layer network, with full interconnections between the layers, built using the SPSS Neural Connection module [20]. We used a topology with 6 inputs, 4 hidden units, and 1 output unit. The inputs were normalized to the interval \([-1,1]\). The activation functions of the hidden units were sigmoids. The learning algorithm used was conjugate gradient, and the weights were initialized randomly within the range \([-0.1,0.1]\), with a uniform distribution. Four restarts were performed with random rearrangements of weights every 70 epochs, to avoid getting stuck in local minima.

For regularization based on early stopping, the data set was split to leave aside the last 10 patterns (the period 95.01–95.10) for out-of-sample predictions. The first 81 of the remaining patterns (period 86.07–93.03) were assigned to the training set, and the last 21 (the period 93.04–94.12) to the test set. In each experiment, the training was stopped when the error of the test set reached the minimum. With neural network models we should care about several sources of variability in the model, such as the random initialization of the initial weights, the setting of the training parameters, the splitting of the data set in training, test and validation set, and so on; see [18], who call it 'model uncertainty' and propose a bootstrap experiment to measure its relevance, concluding that in that case the effects of the data splitting are much more relevant than the effects of the initial setting of the training parameters. In our case we focused on the effects of the preprocessing, keeping fixed all the other factors; for this reason we did not attempt, with our experiment, to select the 'best' neural network. The (very important) model selection issue is currently subject of further research.

As a measure of the overall fitting goodness we used the coefficient of determination \(R^2\)

\[
R^2 = \frac{\frac{1}{n} \sum_{i=1}^{n} (P(i) - \bar{A})^2}{\left(\frac{1}{n} \sum_{i=1}^{n} (P(i) - \bar{A})^2 + \frac{1}{n} \sum_{i=1}^{n} (P(i) - A(i))^2\right)}
\]

where \(A(i)\) are the actual values of month \(i\), \(P(i)\) the corresponding predicted values and \(\bar{A}\) is the average of actual values over the whole data set of \(n\) patterns. To evaluate the prediction quality, the squared error of the predictor divided by the squared error of predicting no change was used. This ratio – we call it RWI (Random Walk Index) – was used for the financial data set in the Santa Fe Competition [22]:

\[
RWI = \frac{\sum_{i=1}^{n} (A(i) - P(i))^2}{\sum_{i=1}^{n} (A(i) - A(i-1))^2}
\]

The RWI is useful to identify how much the predictions are outperforming the random walk model, in which the best prediction for \(x_{t+1}\) is just \(x_t\): a prediction is better than a random decision if the RWI index is below 1, because the prediction error (numerator) is lower than the error of predicting no change (denominator). To add some more information and to compensate the weakness of its null hypothesis in the presence of autocorrelations, we also built for comparison a linear autoregressive model AR(6) of the same order as the neural network estimator, according to the suggestions given in [22], p. 40. To evaluate the out of sample performance, we used the RMS (Root Mean Squared) error, normalized in order to take account of the different magnitude of the time series:

\[
RMS = \sqrt{\frac{\frac{1}{n} \sum_{i=1}^{n} (A(i) - P(i))^2}{\left(\frac{1}{n} \sum_{i=1}^{n} (A(i) - \bar{A})^2\right)}}
\]

where \(m\) is the number of predicted patterns (i.e. in our case for out of sample predictions \(m=10\)) and \(n\) is the total number of patterns.

1 In previous experiments [12] with the same target time series, but with a multivariate version of the mortgage demand model, using cross validation we found a reasonably limited variability given to model uncertainty. We decided not to perform cross validation in the present experiment to shorten the computation time.
<table>
<thead>
<tr>
<th>NH model</th>
<th>( R^2 )</th>
<th>( R^2 \text{ postproc.} )</th>
<th>RWI</th>
<th>RWI \text{ postproc.}</th>
<th>RMS</th>
<th>RMS \text{ postproc.}</th>
</tr>
</thead>
<tbody>
<tr>
<td>raw values</td>
<td>0.708</td>
<td>0.708</td>
<td>0.716</td>
<td>0.716</td>
<td>1.123</td>
<td>1.123</td>
</tr>
<tr>
<td>( \text{ln} )</td>
<td>0.754</td>
<td>0.689</td>
<td>0.363</td>
<td>0.559</td>
<td>0.833</td>
<td>0.941</td>
</tr>
<tr>
<td>( \text{ln+first differences} )</td>
<td>0.818</td>
<td>0.553</td>
<td>0.064</td>
<td>2.042</td>
<td>0.651</td>
<td>1.113</td>
</tr>
<tr>
<td>( \text{ln+seasonal differences} )</td>
<td>0.484</td>
<td>0.829</td>
<td>0.966</td>
<td>0.354</td>
<td>0.887</td>
<td>0.605</td>
</tr>
<tr>
<td>AR(6) model</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>raw values</td>
<td>0.599</td>
<td>0.599</td>
<td>1.648</td>
<td>1.648</td>
<td>1.389</td>
<td>1.389</td>
</tr>
<tr>
<td>( \text{ln} )</td>
<td>0.177</td>
<td>0.197</td>
<td>1.411</td>
<td>1.879</td>
<td>1.399</td>
<td>1.479</td>
</tr>
<tr>
<td>( \text{ln+first differences} )</td>
<td>0.474</td>
<td>0.324</td>
<td>0.196</td>
<td>6.296</td>
<td>0.682</td>
<td>2.317</td>
</tr>
<tr>
<td>( \text{ln+seasonal differences} )</td>
<td>0.519</td>
<td>0.810</td>
<td>0.892</td>
<td>0.304</td>
<td>0.784</td>
<td>0.475</td>
</tr>
</tbody>
</table>

Table 1: Results of the neural networks (NN) and the AR(6) models; the \( R^2 \) and the RWI indexes are calculated on the whole data set (in sample + out of sample); the Root Mean Squared (RMS) error is normalized and calculated only within the out of sample predictions.

4 Results

We built a neural network model and an AR(6) model for each of the following configurations: raw values, natural logarithms, ln+first differences, ln+seasonal differences. The models’ outcomes of single-step predictions were checked both on the transformed values and on the postprocessed data (i.e. converted back to the original form). For each of the 14 resulting sets, we show the \( R^2 \) as a measure of the overall fitting and the RWI as an index of the general predictive performance (both of them referred to the whole data set, in sample + out of sample); we also give a measure of the generalization capability, showing for each model the out of sample RMS error. The results are shown in Table 1. Clearly, if the preprocessing methods are useful, we are expecting an improvement, for both the linear and the neural network models, in the overall fitting and in the generalization ability; this should finally lead to an improvement of the forecasting performance.

As expected, the application of the logarithms and the first differences generally produces a better fitting on the preprocessed set. The NN model shows a much better out of sample normalized RMS (0.65 vs. 1.12) and \( R^2 \) (0.82 vs. 0.71); the AR(6) model shows an improved out of sample normalized RMS (0.68 vs. 1.39), even in the presence of a lower \( R^2 \) (0.47 vs. 0.6). Unfortunately, the results get worse with postprocessed data, due to the amplification effect of the ln transformation on errors which are converted from additive to multiplicative: we could also incur a bias similar to that shown for Gaussian time series in [14] and [15], pg. 311: if \( x \) is a Gaussian time series and we have a model for \( y = \log(x) \), the optimal forecast of \( x \), say \( x_{t+1}^* \), is not equal to the ‘naive’ forecast obtained by taking the exponentiation of the optimal forecast for \( y_{t+1} \), \( \exp(y_{t+1}^*) \), but \( x_{t+1}^* = E_t[\exp(y_{t+1}^*)] = \exp(y_{t+1}^* + \sigma^2/2) \), where \( E_t \) is the expectation operator and \( \sigma^2 \) is the variance of the white noise errors in the model for \( y \).

At a first view, the results on the data set preprocessed with ln+seasonal differences are quite bad. The RWI indexes are close to 1 (0.97 for the neural network, 0.89 for the AR model) and confirm that the results are just slightly better than chance. After postprocessing the results are totally different, and there is a significant improvement in the prediction performance. The AR(6) model shows the lowest out of sample normalized RMS, (0.48 vs. 1.39 with raw values), and the NN model also shows its best performance (0.61 vs. 1.12 with raw values), which confirms that the reconstruction of the seasonal pattern is the determining factor for the predictions. Furthermore, the highest \( R^2 \) (0.83) is obtained by the NN on the postprocessed in seasonal differences experiment. Figure 3 shows the predictions of the raw values model (top) and of the postprocessed in seasonal difference model (bottom).

5 Conclusions

In this paper, we investigated the effects of seasonality on the quality of neural predictions, using the time series of the mortgage market loans purchased in the Netherlands. The selected preprocessing technique, based on seasonal differences of the logarithmic series, was so effective that the original predictive performances of both a neural network estimator and of an autoregressive linear model were greatly improved.

There are several areas for further research. Obviously, a much wider application of the same methodology to different seasonal time series, both real and simulated, is needed. With respect to the choice of preprocessing, the information obtained
Figure 3: Results of the neural network applied to the raw values (top) and after preprocessing with ln seasonal differences (bottom)

with spectral analysis might be integrated with other sources, particularly with a seasonal unit root test like HEGY [17]. The choice of the seasonal differences as opposed to seasonal dummies implies an assumption of stochastic versus deterministic seasonality; the effects of this choice where investigated by Franses with linear models of monthly time series [10], indicating that often the quite common assumption of stochastic seasonality is not justified and can lead to incorrect predictions, and we could expect to obtain similar results with neural network models. The final aim is to develop a general standard and affordable procedure for the choice of preprocessing, including the choice of an appropriate transformation, such as taking logs [2, 6]. Important issues to be faced along this way are concerning the role
and relevance of preprocessing for modelling nonlinear nonstationary processes.

References


