ESANN'2002 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 24-26 April 2002, d-side publi., ISBN 2-930307-02-1, pp. 395-400

Undershooting: Modeling Dynamical Systems by Time Grid Refinements

H. G. Zimmermann,[*] R. Neuneier and R. Grothmann
SIEMENS AG, Corporate Technology, CT IC4, Munich, Germany.

Abstract. Building models of dynamical systems on the basis of observed data, the time grid of the data is typically the same as the time grid of the model. We show that a refinement of the model time grid relative to a wider-meshed time grid of the data provides deeper insights into the dynamics. This ”undershooting” can be derived from the principle of uniform causality. Combining undershooting with recurrent error correction neural networks [3], lead to a *novel* approach which improves the performance of our models by time grid refinements.

1 Introduction

Modeling dynamical systems, the relationship between the model and the data time grid is an important prestructuring element for the modeling. Regarding time discrete systems, the model time normally corresponds to that of the data. Alternatively one may choose a wider-meshed time grid than that of the observed data. As we will show, recurrent neural networks allow to use a finer model time grid than the data grid.

This paper consists of *two* parts: First, we introduce the concept of uniform causality (sec. 2), which allows us to refine the model time grid relative to the data. Undershooting is a neural network based approach of this principle. Second, we combine error correction neural networks [3] and undershooting (sec. 3). As we will show in two empirical studies (forecasting business rentals and foreign exchange rates), the performance of our models can be improved by this novel approach.

2 Principle of Uniform Causality and Undershooting

Uniform Causality provides the basis for the embedding of time discrete systems. Using this principle, we are able to forecast on a finer time grid than that of the data.

First let us assume, that we have to identify an autonomous model for a given time series $s_0, s_1, s_2, \cdots, s_T$. The most obvious approach is to build a discrete time model where the time grid of the model is equal to the grid of the data:

$$s_{n+1} = f(s_n)\,. \tag{1}$$

By iteration of f we obtain the flow F

$$s_n = F(s_0, n) := \underbrace{f \circ \cdots \circ f}_{n}(s_0) \;\text{ for }\; n \in N\,. \tag{2}$$

[*] To whom correspondence should be addressed: Georg.Zimmermann@mchp.siemens.de

Here, we reflect on the consequences of a refinement of the model time grid relative to the data time grid, e. g. modeling annual changes with a 3 months time step. This can be seen as an interpolation scheme between discretely measured values s_n (Fig. 1).
There are lots of possible interpolation schemes. Typically, one uses interpolation techniques, e. g. splines, to find a smooth trajectory along the data points. Since we want to analyze dynamic systems, we introduce the principle of uniform causality (Eq. 3) in order to further constrain the set of possible interpolations (Fig. 1).

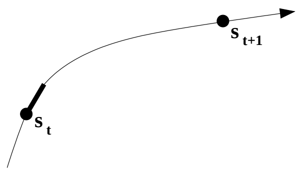

Figure 1. Search for a continuous embedding of a discretely measured dynamic system.

For each $s \in \mathbb{R}^{\mathrm{m}}$ and $t, t_1, t_2 \in \mathbb{R}_+$, the *principle of uniform causality* is given by

$$\textit{embedding:} \quad s_t = f^t(s), \qquad \textit{additivity:} \quad f^{t_1+t_2}(s) = f^{t_2}\left(f^{t_1}(s)\right). \tag{3}$$

The embedding (Eq. 3) can be satisfied by any continuous interpolation. It is formulated as a continuous iteration [2]. The additivity (Eq. 3) is a description of *causality*. Intending to follow a dynamics over $t_1 + t_2$, we can start at s and track the system to $s_{t_1} = f^{t_1}(s)$. Then, we begin at s_{t_1} for the rest of the trajectory $s_{t_1+t_2} = f^{t_2}(s_{t_1})$.

For the case of $t \in \mathbb{N}$, uniform causality (Eq. 3) is self-evident: the embedding describes the data and the additivity is a simple iteration of functions. The necessity of $t \in \mathbb{R}_+$ is a strong additional constraint. Let us assume, that we choose f^t as

$$s_t = f^t(s), t \in [0, \epsilon]. \tag{4}$$

No matter how small ϵ is, due to both the initialization of the first part of the trajectory and the additivity of Eq. 3, there is only one way to follow the path in a causal way.

Modeling a dynamical system by an ordinary differential equation $\frac{ds}{dt} = f(s)$, the principle of uniform causality is always guaranteed:

$$\begin{array}{lrcl} \textit{embedding:} & s_t & = & \int_0^t f(s_\tau)d\tau \\ \textit{additivity:} & s_0 + \int_0^{t_1+t_2} \cdots & = & \left(s_0 + \int_0^{t_1} \cdots\right) + \int_{t_1}^{t_1+t_2} \cdots \end{array} \tag{5}$$

The embedding is given by the integral of the differential equation and the causal additivity is the additivity of the integral. For details of the continuous iteration or continuous embedding of a given time discrete dynamic system see e. g. [2].

We are interested in finding a dynamic law of the discrete dynamics and thus, look for a uniform causal model fitting the data. In a standard approach, we would identify $f(s)$ e. g. by using training data such that

$$s_{n+1} = f(s_n), n \in \mathbb{N}. \tag{6}$$

Thus, we can compute the discrete flow $(s_0, s_1, \ldots)$ by iterations of f

$$s_n = F(s_0, n) = f^n(s_0). \tag{7}$$

Note, that the standard approach of Eq. 6 only uses a natural numbers as iteration indices. To introduce rational iterations, we need the following property of f:

$$f^{tn}(s) = \left(f^t\right)^n(s)\,, \quad n \in \mathbb{N} \quad \mathrm{t} \in \mathbb{R}_+\,, \tag{8}$$

which is a direct consequence of iterating the additivity condition (Eq. 3).

Let us assume, that we are able to identify a function $f^{\frac{1}{q}}(s)$ satisfying

$$s_{n+1} = \underbrace{f^{\frac{1}{q}} \circ \cdots \circ f^{\frac{1}{q}}}_{q}(s_n)\,. \tag{9}$$

This is an iterated system. The parameters $f^{\frac{1}{q}}$ can be estimated by e. g. using error backpropagation. We call $f^{\frac{1}{q}}$ a q-root of f. By $f^{\frac{1}{q}}$ we identify the trajectory for every rational number to a basis of q. The embedding and additivity of Eq. 3 are fulfilled, since operations with rational exponents can be reduced to natural exponents (Eq. 8):

$$\begin{aligned} &\textit{embedding} & s_{\frac{p}{q}} &= f^{\frac{p}{q}}(s_0) = \left(f^{\frac{1}{q}}\right)^p(s_0), \\ &\textit{additivity} \quad f^{\frac{p_1}{q}+\frac{p_2}{q}}(s_0) & &= \left(f^{\frac{1}{q}}\right)^{p_1+p_2}(s_0) = f^{\frac{p_1}{q}}\left(f^{\frac{p_2}{q}}(s_0)\right) \end{aligned} \tag{10}$$

By increasing q our approach would lead to more and more uniform causal solutions, but the system identification (Eq. 9) becomes also more difficult [3].

Undershooting is the refinement of the model time grid by recurrent neural networks, which is directly related to the considerations on uniform causality. Specifying an undershooting network (Fig. 2), let us assume, that we want to forecast a price shift $\ln(p_{t+1}/p_t)$. If we introduce 3 intermediate time steps ($q = 4$) in the description of the dynamics, we obtain the network of Fig. 2, left. This network cannot be trained, because the intermediate targets are not available. We solve this problem by exploiting the identity of $\ln(p_{t+1}/p_t) = \sum_{k=0}^{3} \ln(p_{t+(k+1)/4}/p_{t+k/4})$. This leads to the network of Fig. 2, right, which directly reflects our considerations on uniform causality. Applying shared weights A, C, we introduce an important regularization property

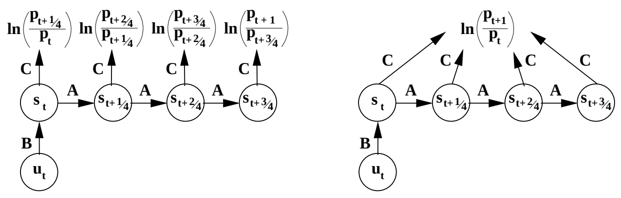

Figure 2. Undershooting neural networks using unfolding in time and shared weights. For simplicity, the initializing input of both recurrent networks in Fig. 2 is only given by u_t using matrix B. The output of the networks is computed by matrix C. The recursive structure of the dynamics is coded in matrix A.

for the recurrent networks of Fig. 2: We assume the same underlying dynamics for all sub-intervals. In addition, multiple gradients for each shared weight are generated. This enables us to estimate our models on the basis of small data sets [3]. The vanishing of gradients is a well-know problem in long chains of hidden layers. Here, we propose to use vario-eta learning, which incorporates a rescaling of the gradients [3].

Empirical Study: Next, we apply undershooting to forecast the annual development of business rentals in Munich. Besides the rental prices, the data set consists of

ESANN'2002 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 24-26 April 2002, d-side publi., ISBN 2-930307-02-1, pp. 395-400

several economic indicators u_t (e. g. German business cycle, inflation, stock and bond market indices). We work with annual data from 1982 to 1996. Each raw input is preprocessed by calculating its scaled momentum (relative change) [3].

We chose a 3 months model time grid to describe the annual rental dynamics. The resulting neural network corresponds to Fig. 2, right. As a benchmark, we refer to a comparable neural network without undershooting. Due to shared weights, both networks have the same number of parameters [1, 3]. The recurrent neural networks are trained until convergence by backprogation through time and *vario-eta* learning using a batch-size of $k = 1$ and a small learning rate of $\eta = 0.005$ [1, 3].

In Fig. 3 we compare the output of the models to the actual rental price shifts.

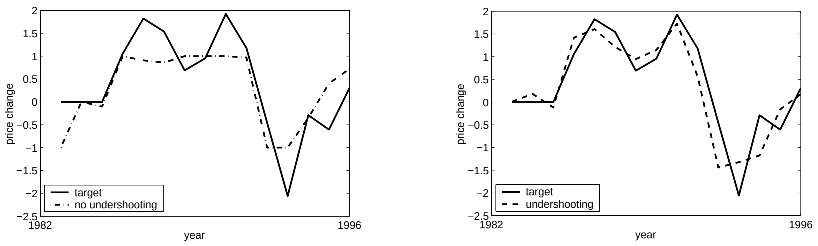

Figure 3. Forecasting annual changes of business rents: A recurrent neural network without undershooting, left, and an approach with undershooting (similar to Fig. 2), right.

The undershooting neural network (Fig. 3, right) is able to fit the rental dynamics more accurately. Due to the refinement of the time grid, the forecast error of this network is much lower than the one of the network without undershooting (Fig. 3, left).

3 Error Correction Networks & Undershooting

We apply error correction neural networks (ECNN) [3] to forecast the British Pound (BPD) / US-Dollar (USD) FX-rate. The ECNN is appropriate for the modeling of dynamical systems in the presence of external shocks or noise. The regularization of the ECNN by undershooting enables us to improve the model performance.

Error Correction Neural Networks (ECNN): Most dynamical systems are partly autonomous and partly externally driven [3]. For discrete time grids, such a dynamics can be modeled by a state transition $s_{t+1} = f(s_t, u_t, y_t - y_t^d)$ and an output equation $y_t = g(s_t)$. The state transition s_{t+1} is a function of the previous state s_t, external influences u_t and a comparison between the model output y_t and the observation y_t^d. If the model error $y_t - y_t^d$ is zero, we have a perfect description of the dynamics. However, due unknown external influences u_t or noise, our knowledge about the dynamics is often limited. We quantify the resulting misspecification of the model by the model error $y_t - y_t^d$ at time t. Hence, the model error $y_t - y_t^d$ serves as an indicator of short-term effects or external shocks [3].

Using weight matrices A, B, C, D of appropriate dimensions corresponding to s_τ, u_τ and $z_\tau = y_\tau - y_\tau^d$, a neural network model of the open system can be written as

$$\begin{aligned} s_{t+1} &= \tanh(As_t + Bu_t + D\tanh(Cs_t - y_t^d)) \\ y_t &= Cs_t \end{aligned} \tag{11}$$

$$\frac{1}{T}\sum_{t=1}^{T}(y_t - y_t^d)^2 \to \min_{A,B,C,D} \tag{12}$$

In Eq. 11, the output y_t is computed by Cs_t and compared to the observation y_t^d. Different dimensions in s_τ and z_τ are adjusted by D. The system identification of Eq. 12 is a parameter optimization task of the weight matrices A, B, C, D [3]. We solve Eq. 12 by finite unfolding in time using shared weights [1]. Fig. 4 depicts the resulting neural network. The ECNN (Fig. 4) is understood best by analyzing the

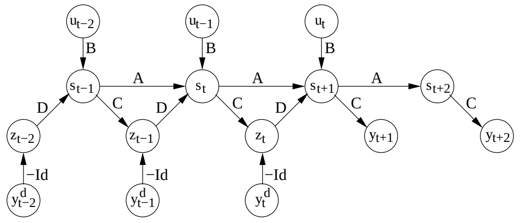

Figure 4. ECNN using unfolding in time and overshooting. Note, that $-Id$ is a fixed negative identity matrix. $z_{t-\tau}$ are output clusters with target values of zero in order to optimize the error correction part.

dependencies of s_{t+1} and s_t, u_t as well as $z_t = Cs_t - y_t^d$. At future time $t+\tau$, there is no compensation of the internal expectations $y_{t+\tau}$ and thus the system offers forecasts $y_{t+\tau} = Cs_{t+\tau}$. An ECNN forecast is based on a modeling of the autonomous part of the dynamics (coded in A), external influences (coded in B) and the error correction mechanism which is also acting as an external input (coded in C and D) [3].

The autonomous part of the ECNN is extended by *overshooting* [3], i. e. we iterate matrices A and C in future direction. Overshooting regularizes the learning and thus may improve the model performance. We supply the additional output clusters $y_{t+\tau}$ with target values. Because of shared weights, no additional parameters are used [3].

ECNN and Undershooting: Next, we combine the ECNN with undershooting:

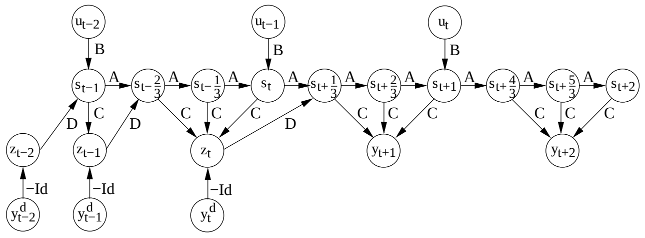

Figure 5. Combining Error Correction Neural Networks and Undershooting.

As depicted in Fig. 5, undershooting is integrated into the ECNN by a redesign of the autonomous part. Assuming a uniform time structure, the underlying system dynamics of the ECNN is divided into autonomous sub-dynamics. The output of the network is computed by gathering the information of the intermediate states.

Empirical Study: We construct *two* ECNNs on the basis of specific economic data (e. g. international stock, bond and commodity indices) to forecast the quarterly and semi-annual development of the BPD / USD FX-rate. Working with monthly data, the in-sample period covers the time from Feb. 1991 to Dec. 1997, while the out-of-sample period ranges from Jan. 1998 to Apr. 2001. The preprocessing of the data is done by calculating the scaled momentum of each input [3].

The first ECNN is similar to Fig. 4, while the other ECNN corresponds to Fig. 5. Here, we model the monthly development of the FX-rate by *two* intermediate time steps. Except the undershooting, both ECNNs are equal. The FX-rate shifts are predicted by overshooting branches of the ECNNs, which provide a sequence of forecasts $t+1, \ldots, t+6$. The ECNNs are trained until convergence by *vario-eta* learning [3].

On the basis of profit & loss curves, the performance of both ECNNs is evaluated by a comparison with a naive strategy, which assumes market trends (see Fig. 6).

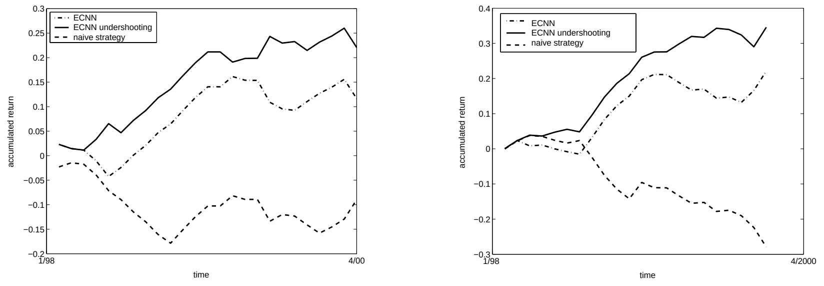

Figure 6. Out-of-sample comparison of profit & loss curves trading the BPD / USD FX-rate. Profit & loss curves resulting form (*i.*) quarterly forecasts, left, and (*ii.*) semi-annual forecasts, right.

Concerning both forecast horizons, the ECNN combined with undershooting is not only able to outperform the benchmark but is also superior to the standard ECNN. This indicates, that undershooting provides the ECNN with additional prior knowledge about the dynamics, which enables us to improve the model performance. Note, that the standard ECNN is also superior to the naive strategy.

4 Conclusion

We introduced an important prestructuring element for the modeling of dynamical systems. Undershooting allows to model the relationship between the model and the data time grid, such that it is possible, to use a finer model time grid than that of the observed data. Interestingly, also a statistical improvement is provided: The network output is computed as an average which may reduce the noise.

Combining undershooting and ECNN, we are not only able to handle noise, missing external influences and external shocks but also gain a deeper understanding of the dynamics. Future work will consider several extensions of the ECNN architecture, since we believe, that ECNN is a promising framework for financial forecasting.

References

[1] Haykin S.: *Neural Networks. A Comprehensive Foundation.* Macmillan College Publishing, New York, 1994. Second edition, 1998.

[2] Kuczma M., Choczewski B., Ger R.: *Iterative Functional Equations*, Cambridge University Press, 1990.

[3] Zimmermann H. G., Neuneier R. and Grothmann R.: *Modeling of Dynamical Systems by Error Correction Neural Networks*, in: Modeling and Forecasting Financial Data. Eds. Soofi, A. and Cao, L., Kluwer, March 2002.