Identifikation, Prognose & Kontrolle dynamischer Systeme mit neuronalen Netzen

Hans Georg Zimmermann
Siemens AG, Corporate Technology
Email: Hans_Georg.Zimmermann@siemens.com
Mathematical Neural Networks

**Complex Systems**

- **Calculus**
- **Neural Networks**
  \[ y = W_2 f(W_1 x) \]
- **Linear Algebra**

**Nonlinear Regression**

Based on data identify an input-output relation

\[
y = W_2 f(W_1 x)
\]

\[
E = \sum_{t=1}^{T} (y_t - y_t^d)^2 \rightarrow \min_{W_1, W_2}
\]

Existence Theorem:
(Hornik, Stinchcombe, White 1989)

3-layer neural networks can approximate any continuous function on a compact domain.

Neural networks imply a **Correspondence** of Equations, Architectures, Local Algorithms.
Neural Networks are No Black Boxes

Application: Modeling of a Gas Turbine
- Inputs: 35 sensor measures and control variables of the turbine
- Output: NO\textsubscript{x} emission of the gas turbine

Sensitivity Analysis: Compute the first derivatives along the time series:
\[
\frac{\partial \text{output}}{\partial \text{input}_i} > 0 \quad \frac{\partial \text{output}}{\partial \text{input}_i} < 0
\]

A classification of input-output sensitivities:
- \textit{constant over time} (= linear relationship)
- \textit{monotone} (input can be used in 1dim. control)
- \textit{non-monotone} (only multi-dim control possible)
- \textit{~ zero} (input useless in modeling and control)
The Curse of Dimensionality in Approximation Theory

The curse of dimensionality in Standard Approximation:

\[ f(x) \approx \sum_{j=1}^{m} v_j b_j(x) \quad \text{with} \quad \|v_j\| \approx c^{\dim(x)} \]

This is a linear superposition of basis functions – their number & the number of parameters increase exponentially with \( \dim(x) \).

Neural Networks escape the curse of dimensionality:

\[ f(x) \approx \sum_{j=1}^{m} v_j b(w_j, x) \quad \text{with} \quad \|v_j, w_j\| \approx \Var(f) \]

The independence of the number of parameters from the input dimension is paid with nonlinear optimization.

Support Vector Machines offer an alternative remedy:

\[ f(x) \approx \sum_{j=1}^{m} v_j b(x - x_j) \quad \text{with} \quad \|v_j\| \approx \|\text{data} \{x_j\}\| \]

Here we have a linear superposition of basis functions, which are chosen as part of the data - which can be a drawback.
Finite unfolding in time transforms time into a spatial architecture. We assume that future inputs are constant. The analysis of open systems by RNNs allows a decomposition of its autonomous and external driven subsystems.

Long-term predictability depends on a strong autonomous subsystem.

\[ s_{t+1} = \tanh(As_t + Bu_t) \]

\[ y_t = Cs_t \]

\[ \sum_{t=1}^{T} (y_t - y^d_t)^2 \rightarrow \min_{A,B,C} \]

\[ s_t = f(s_{t-1}, u_{t-1}) \]

\[ y_t = g(s_t) \]
An error correction system considers the forecast error in present time as a reaction on unknown external information.

In order to correct the forecasting this error is used as an additional input, which substitutes the unknown external information.

\[ s_{t+1} = f(s_t, u_t, (y_t - y_t^d)) \]
\[ y_t = g(s_t) \]
\[ \sum_{t=1}^{T} (y_t - y_t^d)^2 \rightarrow \min_{f, g} \]
Combining Variance - Invariance Separation with ECNN

- The **bottleneck autoassociator** solves the variance - invariance decomposition.
- The **Error Correction Neural Network** solves the transformed temporal problem.
- The sub-networks are implicitly coupled by shared weights.
- The model combines the identification of a manifold and a dynamics on this manifold.
Control of Dynamical Systems with Recurrent Neural Networks

Questions to Solve

- System Identification
- State Estimation
- Controller Design
System Identification, State Estimation & Optimal Control with RNNs

\[ s_{\tau+1} = \tanh(A s_\tau + B u_\tau) \]
\[ y_\tau = C s_\tau \]

**Unfold the RNN into the future, given** \(A, B, C\).

**System identification**

\[ \sum_{\tau \leq t} (y_\tau - y_\tau^d)^2 \rightarrow \min_{A, B, C} \]

**Learn a linear feedback controller:**

\[ u_\tau^* = u(s_\tau) = D s_\tau \]

**Normative target**

\[ \sum_{\tau > t} L(y_\tau) \rightarrow \min_D \]
The model is unfolded along history → only 1 training example

… but to understand the dynamics of the observables, we have to reconstruct at least a part of the hidden states of the world. Forecasting is based on observables and hidden states.
The Identification of Dynamical Systems in Closed Form

Embed the original architecture into a larger architecture, which is easier to learn. After the training, the extended architecture has to converge to the original model.

The essential task is NOT to reproduce the past observations, but to identify related hidden variables, which make the dynamics of the observables reasonable.
20 Days Price Forecasting with Causal Neural Networks

**EEX Base Market**
- Price in Euro
- 20 Days Price Forecasting
- 2010-08-09 – 2010-09-03
- 2010-08-23 – 2010-09-17

**LME Copper Market**
- Price in US-Dollar
- 2010-08-09 – 2010-09-03
- 2010-08-23 – 2010-09-17

Causal vs. Actual Prices:
- Actual EEX Base 2012
- Actual LME Copper Spot
Identification of Goal Oriented Dynamical Systems

Causal nets are given by an initial state \( s_0 \) & matrix \( A \)

Retro-causal nets are given by a final state \( s_T \) & matrix \( A' \)

When we are able to identify the goals of the agents we can try to explain the dynamics backward from their goals.

When we have identified the motivation of the suspect, it is easy to reconstruct the sequence of events

\[
s'_t = A' \tanh(s'_{t+1}) \quad s'_T \quad \text{state transition}
\]

\[
y_t = [Id,0]s'_t \quad \text{output equation}
\]

\[
\sum_{t=1}^{T} (y_t - y^d_t)^2 \rightarrow \min_{A',s'_T} \quad \text{identification}
\]
The Identification of Dynamical Systems in Closed Form

Embed the original architecture into a larger architecture, which is easier to learn. After the training, the extended architecture has to converge to the original model.

The essential task is NOT to reproduce the past observations, but to identify related hidden variables, which make the dynamics of the observables reasonable.
Comparison of Causal Forecasts and Retro-Causal Forecasts

Causal Forecasts

Retro-Causal Forecasts

**Insights:** Causal and retro-causal forecasting is complementary. Ex ante there is no chance to know which approach is the appropriate. We should find a way to combine the best of both approaches.
Combining Causal & Retro-Causal Neural Networks (CRCNN)

- We explain observations by a symmetric superposition of causal & retro-causal subnets.
- Both subnets are universal learners, but the more appropriate branch learns faster and reduces the error flow of the opposite branch.
- In non-unique optimization, we have to have attention on the path to the optimum!!!

\[
s_t = A \tanh(s_{t-1}), s_0
\]

\[
s'_t = A' \tanh(s'_{t+1}), s'_T
\]

\[
y_t = [Id,0]s_t + [Id,0]s'_t
\]

\[
\sum_{t=1}^{T} (y_t - y^d_t)^2 \rightarrow \min_{A,s_0,A',s'_T}
\]

causal transition

retro transition

output equation

identification

\[
s'_t \rightarrow A' \tanh \rightarrow Id \rightarrow s'_t
\]

\[
s'_t \rightarrow A' \tanh \rightarrow Id \rightarrow s'_t
\]

\[
s'_t \rightarrow A' \tanh \rightarrow Id \rightarrow s'_t
\]

\[
s'_t \rightarrow A' \tanh \rightarrow Id \rightarrow s'_t
\]

\[
s'_t \rightarrow A' \tanh \rightarrow Id \rightarrow s'_t
\]

bias
Architectural Teacher Forcing for Causal-Retro-Causal Networks

Obviously we do not know the future data for the observables. Thus we can not do a teacher forcing beyond present time.

The ATF has to support the learning of the moving target in a symmetric way:

\[ 0 = \Delta_t + \Delta'_t - y^d_t \quad \Rightarrow \quad \Delta_t(\text{adjust}) = y^d_t - \Delta'_t = \Delta_t - (\Delta_t + \Delta'_t - y^d_t) \]

\[ 0 = \Delta_t + \Delta'_d - y^d_t \quad \Rightarrow \quad \Delta'_t(\text{adjust}) = y^d_t - \Delta_t = \Delta'_t - (\Delta_t + \Delta'_t - y^d_t) \]

At every time step the network has a fix point loop (equality constraint), defining a manifold.
CRCNN incorporating Higher-Order Teacher Forcing

EEX Base Market

- Price in Euro
- 2010-08-09 – 2010-09-03
- 2010-08-23 – 2010-09-17
- 51.25 – 54.75

EEX Base Market

- Price in US-Dollar
- 2010-08-09 – 2010-09-03
- 2010-08-23 – 2010-09-17
- 6600.00 – 8000.00

LME Copper Market

- Price in Euro
- 2010-08-09 – 2010-09-03
- 6950.00 – 7750.00

LME Copper Market

- Price in US-Dollar
- 2010-08-09 – 2010-09-03
- 6800.00 – 7800.00
Approaches to Model Uncertainty in Forecasting

1 Measure uncertainty as volatility (variance) of the target series. The underlying forecast model is a constant. Thus $\sin(\omega t)$ can be highly uncertain!

2 Build a forecast model. The error is interpreted as uncertainty in form of additive noise. The width of the uncertainty channel is constant over time.

3 Describe uncertainty as a diffusion process (random walk). The diffusion channel widens over time, e.g. scaled by the one-step model error.

For large systems 2 & 3 fail: We have to learn to zero error → the uncertainty channel disappears.

4 One large model doesn't allow to analyze forecast uncertainty, but an ensemble forecast shows the characteristics of an uncertainty channel: Given a finite set of data, there exist many perfect models of the past data, showing different future scenarios caused by different estimations of the hidden states.
EEX Electricity Price Heatmap: CRCNNp440f20s300a17
Copper Price Forecasting in Form of Percentile Ranges
Selected References


