Univariate Time Series Forecasting using Echo State Networks: An Empirical Application

26th International Conference on Computational Statistics

Alexander Häußer

Justus-Liebig-University Giessen
Faculty 02 - Economics and Business Studies
Chair of Statistics and Econometrics

August 28, 2024

alexander.haeusser@wirtschaft.uni-giessen.de

Outline

Outline

Introduction

Research objectives

- Develop an algorithm for fast and fully automatic time series modeling and forecasting using Echo State Networks (ESN)
- Benchmark approach against state-of-the-art forecasting methods
- Functions and methods available in the R package `echos`

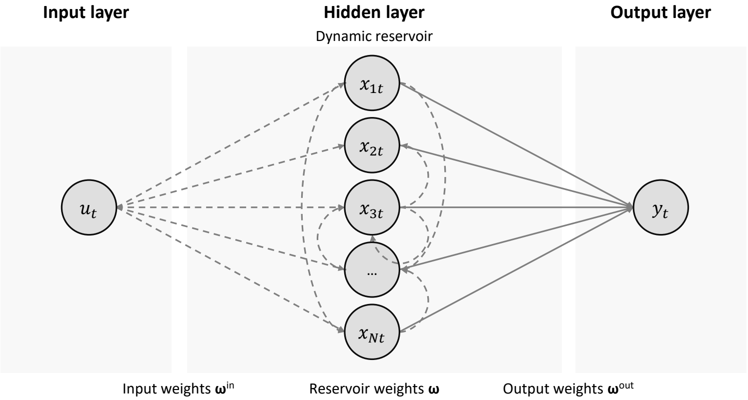
Experimental setup

- Data from the M4 Forecasting Competition
  - 2,400 monthly and 1,200 quarterly time series
  - Forecast horizon $h = 18$ for monthly and $h = 8$ for quarterly data
- Evaluate forecast accuracy (MASE, sMAPE) and measure computational run-time
- Compare against simple methods, statistical models and neural networks

Outline

Echo State Network - Architecture

- Input $u_t = y_{t-1}$ is non-linearly expanded into a high-dimensional feature space (i.e., internal states)
- Output y_t is combined via trainable weights from these internal states
- Dashed lines indicate fixed weights (ω^{in}, ω) and solid lines indicate trainable weights ω^{out}



Echo State Network - Basic model[1]

1. Data pre-processing

- Identify non-stationarity via KPSS test
- If required, calculate (first) difference to remove non-stationarity
- Scale (stationary) time series to interval $[-0.5, 0.5]$

2. Reservoir generation

- Calculate the internal states according to

$$\mathbf{x}_t = \tanh\left(\boldsymbol{\omega}^{in} u_t + \boldsymbol{\omega}\mathbf{x}_{t-1}\right)$$

- First autoregressive lag as input, i.e., $u_t = y_{t-1}$
- Input and reservoir weight matrices $\boldsymbol{\omega}^{in}$ and $\boldsymbol{\omega}$
- Collect internal states in design matrix $\mathbf{X} = [\mathbf{1}, x_{1t}, x_{2t}, ..., x_{Nt}]$

3. Model estimation and selection

- Linear model

$$\mathbf{y} = \mathbf{X}\boldsymbol{\omega}^{out} + \epsilon$$

- Estimate coefficients via ridge regression

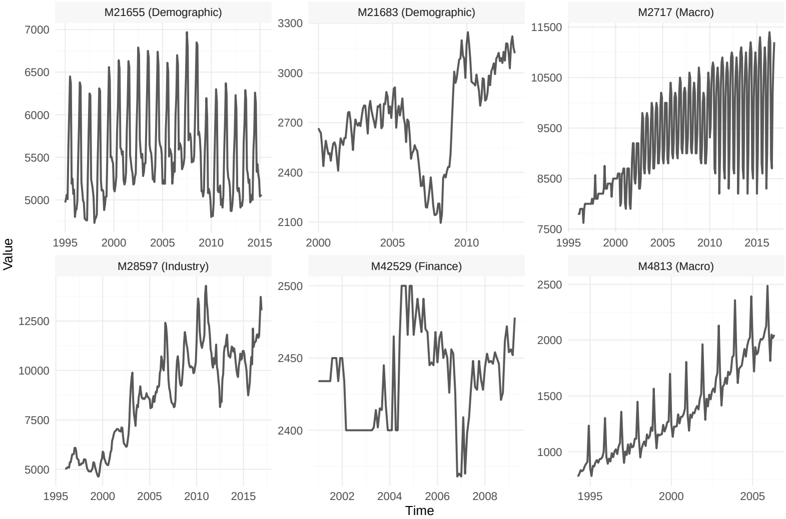
$$\hat{\boldsymbol{\omega}}^{out} = (\mathbf{X}^\top\mathbf{X}+\mathbf{R}_\lambda)^{-1}\mathbf{X}^\top\mathbf{y}$$

- Regularization parameter λ is determined via random search by minimizing the BIC

[1] More details in the appendix

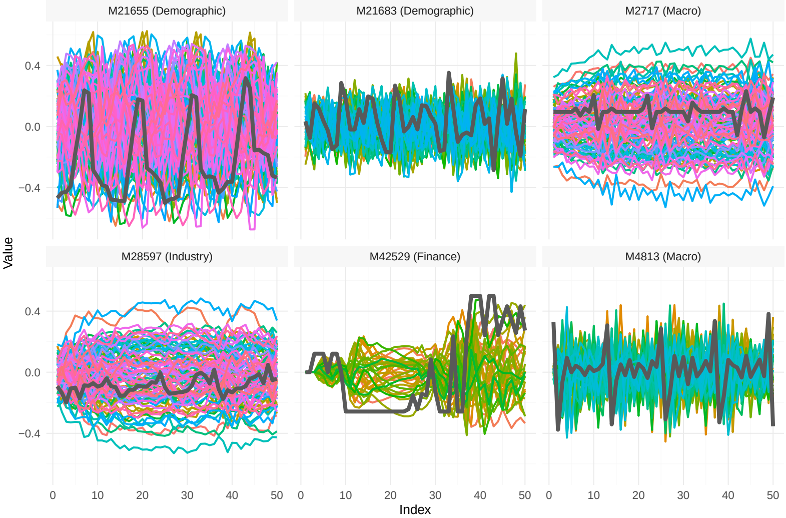
Data from the M4 Forecasting Competition

- Original M4 dataset consists of 100,000 time series
- Different frequencies and applications fields
- Randomly selected 2,400 monthly and 1,200 quarterly series
- Diverse dataset with different characteristics (trend, season, non-stationarity, etc.)



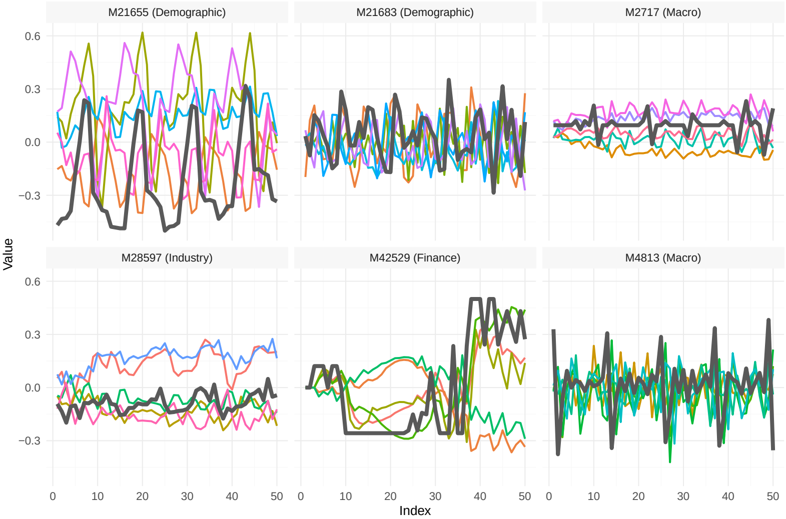
Reservoir generation - Feature engineering using the echo state approach

- Internal states (colored lines) and the pre-processed output variable (black line)
- Non-linear dimensionality expansion as a feature engineering technique for time series
- Problems
  - Multicollinearity
  - Overfitting
  - etc.



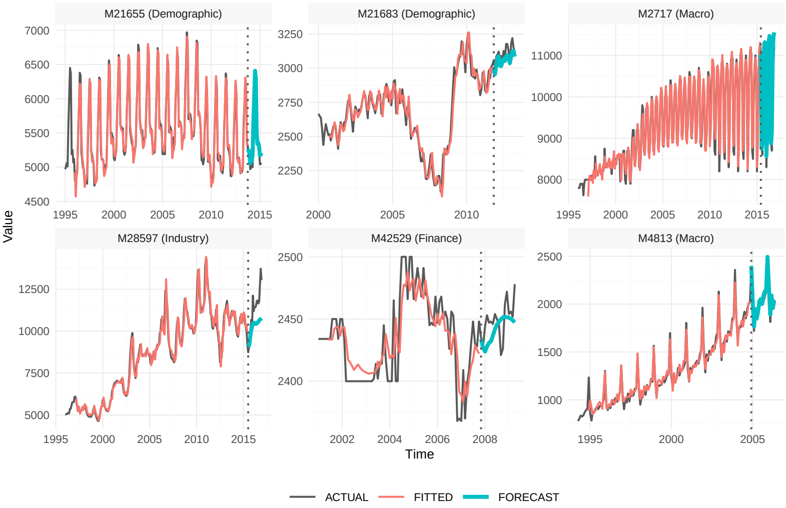
Model selection and estimation - Random search and ridge regression

- Top 5 internal states (colored lines) and the pre-processed output variable (black line)
- Selection based on the size of the absolute value of the coefficients (high correlation $\rightarrow$ predictive power)
- Lead-lag relationship captures auto-correlation, seasonality, etc.



Actual values, fitted values and forecasts from trained ESN

- Recursive forecasting due to *autoregressive nature*
- Actual values (black), fitted values (red) and out-of-sample forecasts (green)
- Vertical dotted line: split into training and testing (holdout)
- ESN model produces reasonable forecasts



Outline

Monthly dataset - Forecast accuracy and run-time

Key takeaways

- ESN achieved very good results in terms of forecast accuracy and run-time
- Differences between the top methods are marginal
- Statistical methods outperform neural networks

Model	MASE		sMAPE [%]		Run Time [sec]	
	Mean	Median	Mean	Median	Mean	Total
ESN	**0.885**	0.717	17.877	12.296	0.201	483.406
TBATS	0.887	**0.711**	17.075	12.246	0.650	1561.136
ARIMA	0.891	0.729	17.963	12.415	0.256	613.874
ETS	0.902	0.722	17.763	**12.182**	0.167	401.617
THETA	0.902	0.716	**16.814**	12.196	0.024	58.558
ELM	0.930	0.739	18.430	13.081	23.995	57588.779
MLP	1.026	0.839	21.398	14.637	2.209	5302.472
NNETAR	1.042	0.830	19.663	14.374	39.318	94363.406
DRIFT	1.077	0.806	20.013	13.750	0.025	60.752
NAIVE	1.097	0.847	19.573	14.419	0.030	72.308
PROPHET	1.143	0.890	23.901	15.859	0.783	1879.991
SNAIVE	1.165	0.948	20.489	15.770	0.025	60.429
TSLM	1.538	1.156	31.740	20.203	0.030	71.990
MEDIAN	2.841	1.788	37.020	31.912	**0.024**	**56.925**
MEAN	2.849	1.958	38.178	33.766	0.026	61.436

Quarterly dataset - Forecast accuracy and run-time

Key takeaways

- ESN achieved good results in terms of forecast accuracy and run-time
- Statistical methods outperform neural networks

Model	MASE		sMAPE [%]		Run Time [sec]	
	Mean	Median	Mean	Median	Mean	Total
ETS	**1.082**	**0.839**	10.264	5.398	0.062	74.100
TBATS	1.104	0.843	**9.975**	5.579	0.336	403.254
ELM	1.114	0.880	10.771	**5.352**	1.382	1658.841
ESN	1.114	0.908	10.672	5.540	0.052	61.836
ARIMA	1.119	0.876	10.410	5.703	0.106	127.593
THETA	1.150	0.908	10.339	5.859	0.026	31.040
DRIFT	1.155	0.888	10.915	5.524	0.025	30.375
MLP	1.187	0.917	11.673	5.833	0.602	722.079
NAIVE	1.329	1.070	11.358	6.813	0.035	41.978
PROPHET	1.435	1.089	14.316	7.070	1.653	1983.211
NNETAR	1.444	1.126	12.793	7.358	25.078	30093.072
SNAIVE	1.513	1.282	12.753	8.003	0.025	29.628
TSLM	1.879	1.478	16.222	9.759	0.028	33.849
MEAN	4.241	3.567	29.691	24.863	**0.024**	**28.402**
MEDIAN	4.361	3.510	31.259	24.729	0.024	29.179

Outline

Summary and further research

Summary

- Proposed model achieved high accuracy and can outperform or compete against state-of-the-art forecasting methods
- Empirical results demonstrate the universal learning capabilities and the potential of ESNs
  - Data-driven instead of model-driven forecasts
  - Being more generic and making less assumptions

Further research

- Probabilistic forecasting, i.e., enhance point forecasts with forecast distributions
- Multivariate forecasting and exogenous inputs
- Deep learning framework, i.e., reservoir-to-reservoir
- Reservoir generation as feature engineering technique

Outline

References I

[1] James Bergstra and Yoshua Bengio. "Random Search for Hyper-Parameter Optimization". In: *Journal of Machine Learning Research* 13.10 (2012), pp. 281–305. URL: http://jmlr.org/papers/v13/bergstra12a.html.

[2] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The elements of statistical learning: data mining, inference and prediction*. 2nd ed. New York: Springer, 2009. URL: http://www-stat.stanford.edu/~tibs/ElemStatLearn/.

[3] Rob J. Hyndman and George Athanasopoulos. *Forecasting: principles and practice*. 3rd edition. OTexts: Melbourne, Australia, 2021. URL: OTexts.com/fpp3.

[4] Alexander Häußer. *echos: Echo State Networks for Time Series Modeling and Forecasting*. R package version 0.1.0. 2024. URL: https://github.com/ahaeusser/echos.

[5] Herbert Jaeger. "The "echo state" approach to analysing and training recurrent neural networks-with an erratum note". In: *Bonn, Germany: German National Research Center for Information Technology GMD Technical Report* 148.34 (2001), p. 13.

References II

[6] Herbert Jaeger. "Tutorial on training recurrent neural networks, covering BPPT, RTRL, EKF and the" echo state network" approach". In: (2002).

[7] Mantas Lukoševičius. "A Practical Guide to Applying Echo State Networks". In: (Jan. 2012). DOI: 10.1007/978-3-642-35289-8_36.

[8] Mantas Lukoševičius and Herbert Jaeger. "Reservoir computing approaches to recurrent neural network training". In: *Computer Science Review* 3.3 (2009), pp. 127–149.

[9] Spyros Makridakis, Evangelos Spiliotis, and Vassilios Assimakopoulos. "The M4 Competition: 100,000 time series and 61 forecasting methods". In: *International Journal of Forecasting* 36.1 (2020). M4 Competition, pp. 54–74. ISSN: 0169-2070. DOI: https://doi.org/10.1016/j.ijforecast.2019.04.014. URL: https://www.sciencedirect.com/science/article/pii/S0169207019301128.

Outline

Echo State Network - Settings and hyperparameters (1/2)

Reservoir generation

- Input weight matrix $\boldsymbol{\omega}^{in} \in \mathbb{R}^{N\times 1}$ and the reservoir weight matrix $\boldsymbol{\omega} \in \mathbb{R}^{N\times N}$ are generated randomly
- Dynamic rules for determining the number of internal states, initial drop-out, etc.

Setting	**Value/Formula**
Input ($\mathbf{u}$)	$u_t = y_{t-1}$
Output ($\mathbf{y}$)	y_t
Activation function	$tanh(.)$
Number of internal states	$N = min(\lfloor 0.4T \rfloor, 100)$
Initial drop-out	$\delta = \lfloor 0.05T \rfloor$
Input weight matrix ($\boldsymbol{\omega}^{in}$)	
Dimension	$N \times 1$
Random uniform	$[-0.5, 0.5]$
Density	100%
Reservoir weight matrix ($\boldsymbol{\omega}$)	
Dimension	$N \times N$
Random uniform	$[-0.5, 0.5]$
Density	50%
Spectral radius	$\rho = 1.0$

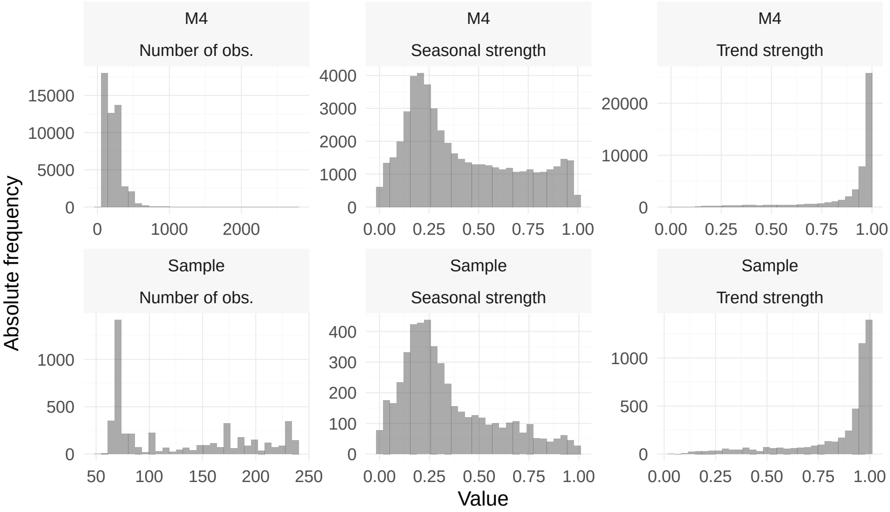
Echo State Network - Settings and hyperparameters (2/2)

Model estimation and selection

- Ridge Regression reduces overfitting and eliminates difficulties with ill-posed optimization problems
- Regularization parameter λ optimized via random search
- Effective degrees of freedom to determine model complexity in BIC

Setting	Value/Formula
Model type	$\mathbf{y} = \mathbf{X}\omega^{out} + \epsilon$
Output weight matrix	$\hat{\omega}^{out} = (\mathbf{X}^\top \mathbf{X} + \mathbf{R}_\lambda)^{-1}\mathbf{X}^\top \mathbf{y}$
Optimization algorithm	Random search
Search space (λ)	
Number of random values	$K = 2N$
Interval of random uniform	$[10^{-4}, 2]$
Information criterion	$BIC_\lambda = -2L + \ln(T) df_\lambda$
Effective degrees of freedom	$df_\lambda = tr[\mathbf{X}(\mathbf{X}^\top \mathbf{X} + \mathbf{R}_\lambda)^{-1}\mathbf{X}^\top]$

Monthly dataset - Number of obs. and strength of trend and seasonality



Quarterly dataset - Number of obs. and strength of trend and seasonality