Forecasting for Data Scientists

Theory Session 2 – Global Models and Machine Learning for Forecasting

Christoph Bergmeir

October, 2025

Universidad de Granada, Spain
Monash University, Melbourne, Australia
https://www.cbergmeir.com

Global Models

Global models

Name "global models" was introduced by Januschowski et al. (2020) . It is arguably not a good name but it's the name we got for now.

- Traditionally, *one* time series is seen as a dataset
- One model is built per time series
- low sampling frequencies and non-stationarities like structural breaks make usually that we don't have enough data to fit complex (ML) models
- M3, M4 datasets are put together under this paradigm; very different series from different frequencies, different domains, etc.

Global models (cont'd)

Paradigm shift:

- a *set* of time series is a dataset (e.g., a set of series from retail, smart meters, etc.)
- build a model across the series
- names: global modelling, cross-learning, multi-task learning, pooled regression

–> Now, enough data, due to more series.

–> ML methods are competitive now

Global models (cont'd)

- Local model: typically fitting a model with few (<10) parameters to a single series
- if you have 10k series and fit 5 parameters, you end up with 50k parameters

–> fit a global model with 5k parameters instead

–> Overall complexity of set of local models grows when dataset grows; complexity of global model stays the same

Global models (cont'd)

- Global models can afford to be more complex
- Complexity can be added as:
  - longer memory (longer input windows, more lags)
  - non-linear/non-parametric models (NNs, GBT, ...)
  - data partitioning

Global models are not multivariate models

- Global models learn across series but predict every series in isolation (input is a particular series at a time, but parameters are shared across series)
- they can work on datasets where series have different lengths and/or are not aligned, like the M3, M4 datasets
- they do not take into account interactions between series
- the concepts are orthogonal: methods can be local/univariate, global/univariate, local/multivariate, global/multivariate

Global models are not multivariate models (cont'd)

History of global models

- Dating back to the early 2000s and earlier (Duncan et al., 2001)
- Pooled regression a standard statistics technique, see, e.g.: Gelman et al. (2007)
- Pooled regression for forecasting: Trapero et al. (2015)
- 2016: Smyl and Kuber (2016), CIF competition: Štěpnička and Burda (2016)
- 2017: DeepAR (Salinas et al., 2019b) and other works from Amazon, e.g., Wen et al. (2017); our work in Bandara et al. (2017)
- after 2018: ... many more works

Kaggle competitions

- A good overview give Bojer and Meldgaard (2020)
- The following are relevant forecasting competitions held on Kaggle:
  - Walmart Store Sales Forecasting (2014)
  - Walmart Sales in Stormy Weather (2015)
  - Rossmann Store Sales (2015)
  - Wikipedia Web Traffic Forecasting (2017)
  - Corporación Favorita Grocery Sales Forecasting (2018)
  - Recruit Restaurant Visitor Forecasting (2018)
- All competitions were won by global models, the latter four by either GBT or NN models or ensembles of those.

M5 competition

- held in 2020 on Kaggle
- dominated by LightGBM: Makridakis et al. (2020), DeepAR and NBEATS also successful
- A team of my students won a Kaggle Gold, 17th in the competition out of >5k participants.
- They used an ensemble of LightGBM and pooled (linear) regression.

Global models (cont'd)

- Global models have shown success to a surprising / unreasonable degree
- Idea at the beginning was that the series have to be in some way "related/similar" so that we can learn something useful across them
- "Related" in terms of similarity of their DGP (not mere correlations)

Global models (cont'd)

Montero-Manso and Hyndman (2020):

- Global model can produce the same forecasts as local models, without any assumptions about similarity
- Instead of fitting one complex pattern across series, a global model even works well to fit many simple patterns that are different in the series

$\Rightarrow$ The series don't have to be "related"

$\Rightarrow$ Series are related through the evaluation regime: We evaluate as an average error over all of them.

Global models (cont'd)

Hewamalage et al. (2021a):

- Simulation study around when Global Models actually work well
- Simulated simple and complex DGPs, long and short series, homogeneous and heterogeneous datasets
- Local linear models work well if patterns are simple and enough data per series
- RNN, LGBM globally trained is competitive both if there are simple but quite different patterns, as well as if there are complex non-linear patterns
- Global models improve the more series are available, local models don't

Traditional Machine Learning Methods for Forecasting

Machine Learning methods for forecasting

- global modelling across series
- can apply them to a single time series, if long enough
- longer series due to finer granularities (secondly, minutely, half-hourly series available over years)
- additional metadata

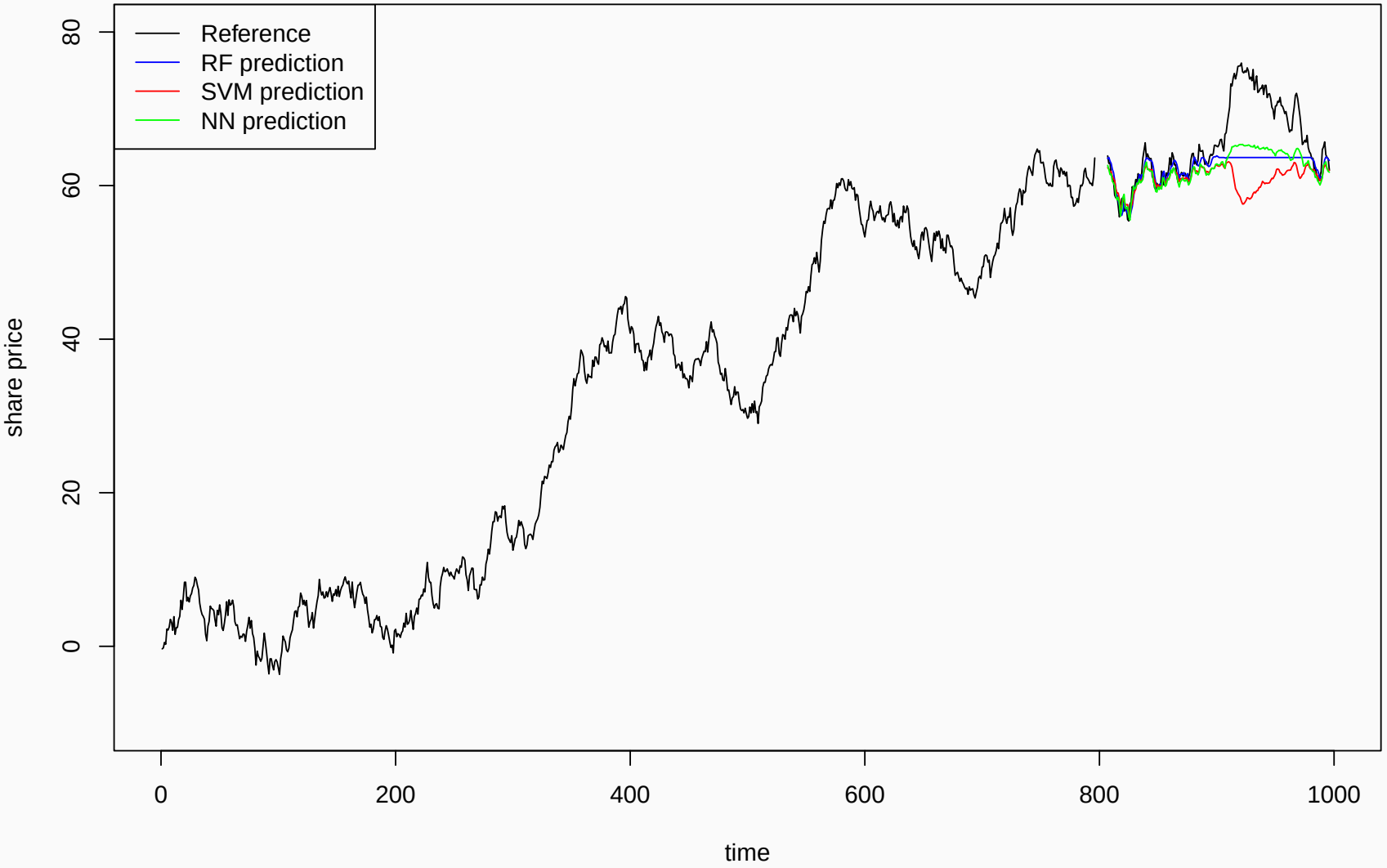
Non-linear autoregression

- basic setup is a (non-linear, non-parametric) autoregressive model
- then, you can use your favourite ML method out of the box
- we've seen that this can approximate an ARMA model, if we choose the input window larger than the ARMA model has it

Problem: Non-stationarity

- Data Distribution changes over time
- Many (most?) real-world problems have a time component and changing distributions
- Think about detecting cars on the street with a dataset from the 1970s
- In time series it is more explicit though and has more impact

Problem: Non-stationarity (2)



Problem: Non-stationarity (3)

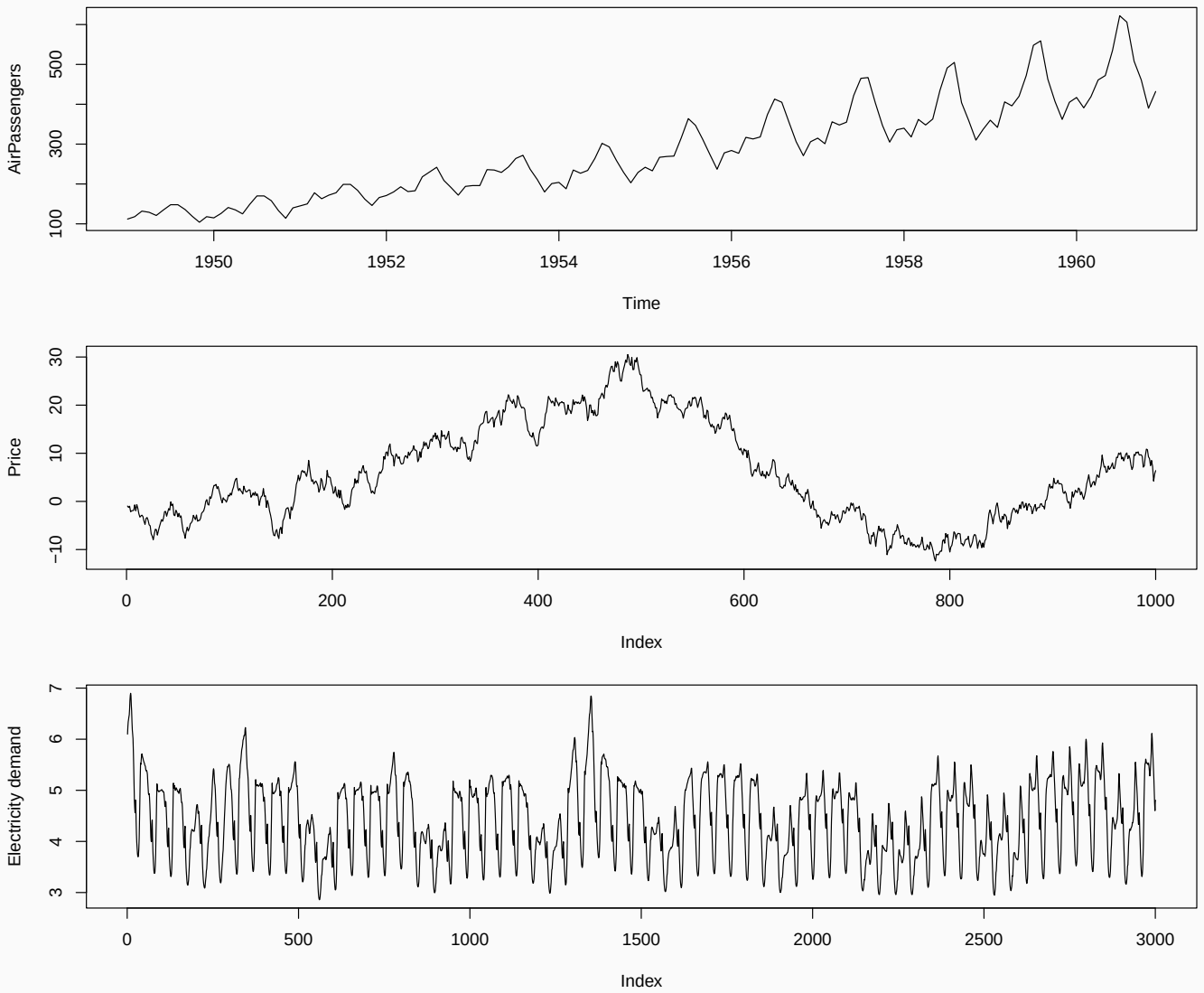
> *Dividing the world into linear and non-linear is like dividing the world into bananas and non-bananas.*

Just as there are many different forms of non-linearity, there are also many different forms of non-stationarity

Typical non-stationarities in time series:

- Change in mean: seasonality, trend
- Change in variance: heteroskedasticity
- Random walks (stochastic trends)
- Structural breaks

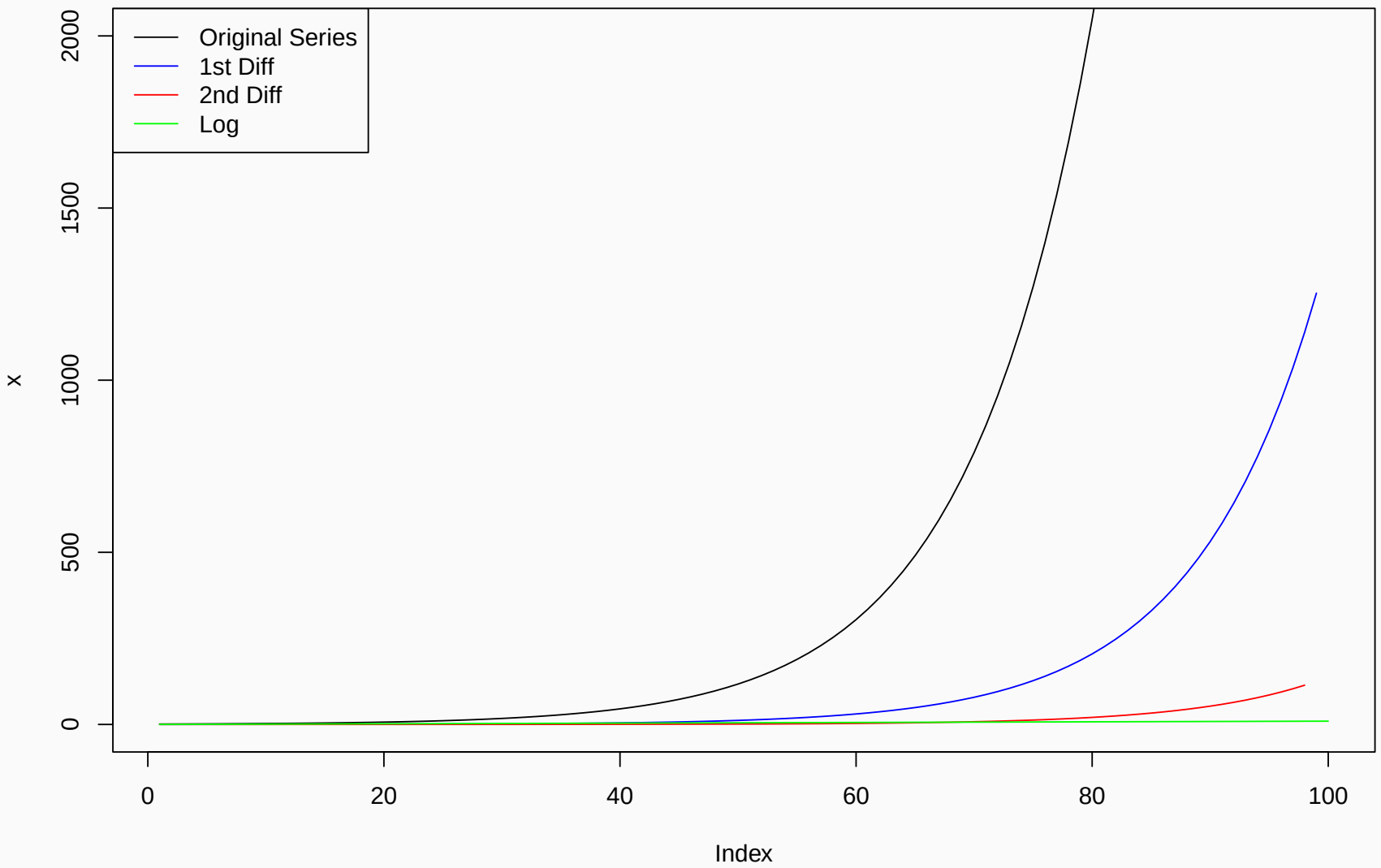
Problem: Non-stationarity (4)



How to achieve stationarity? Differencing

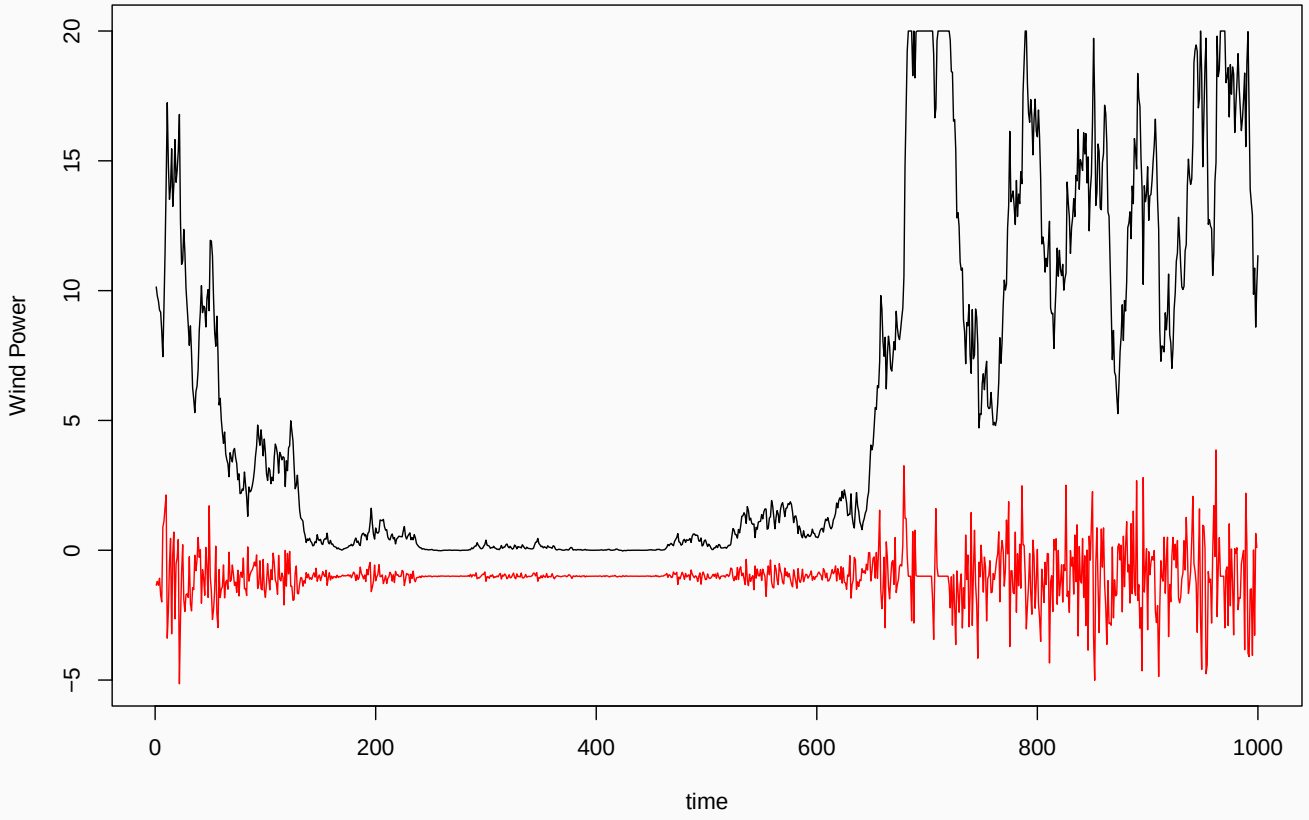
- As in Econometrics and Finance, many series are close to random walks, there, differencing is a common tool to achieve stationarity
- see ARIMA modelling earlier
- Differencing only solves some forms of non-stationarity, not others
- Losing information about the scale. Can have an additional input as the (log of) the original scale.
- Differencing can help to make ML models more robust

Differencing does not always work



Differencing loses information

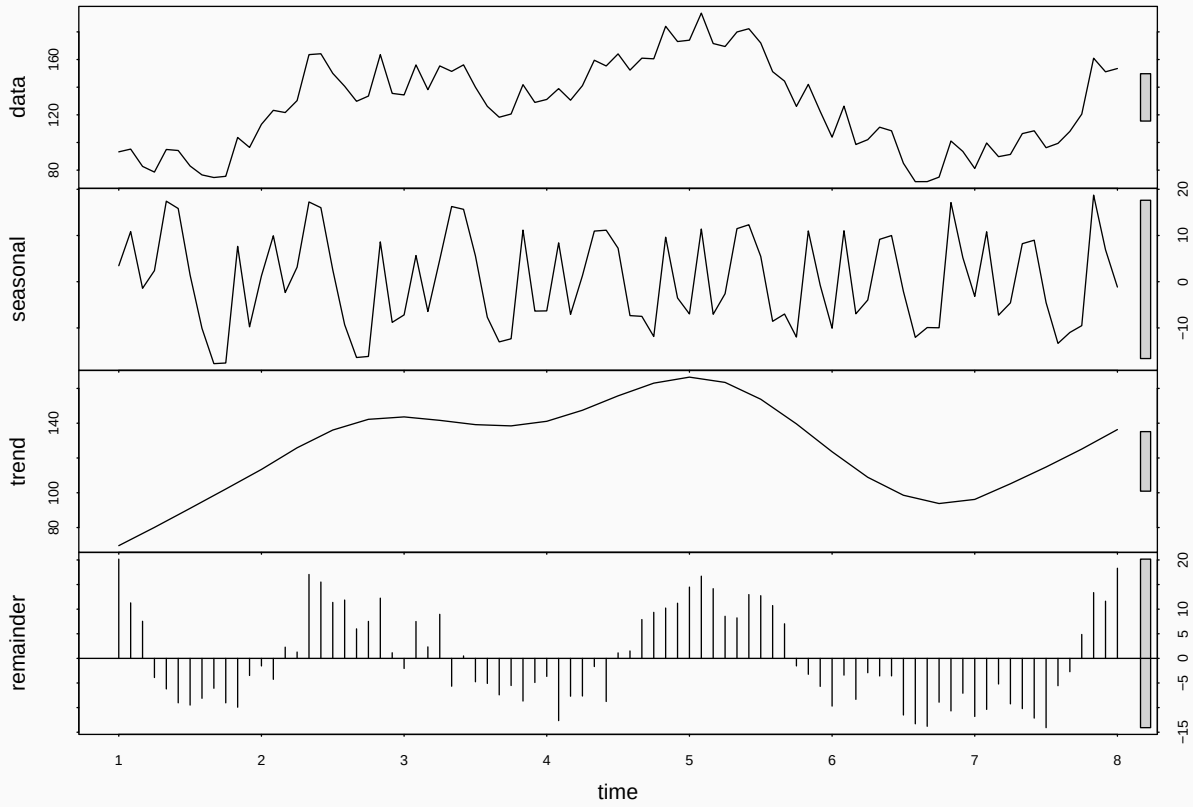
Example: Wind power forecasting



How to model trend?

Detrending

- Problem: it is not well-specified what a trend is
- Essentially just a smoothed version of the series
- We still need to forecast the trend then



How to model trend? (cont'd)

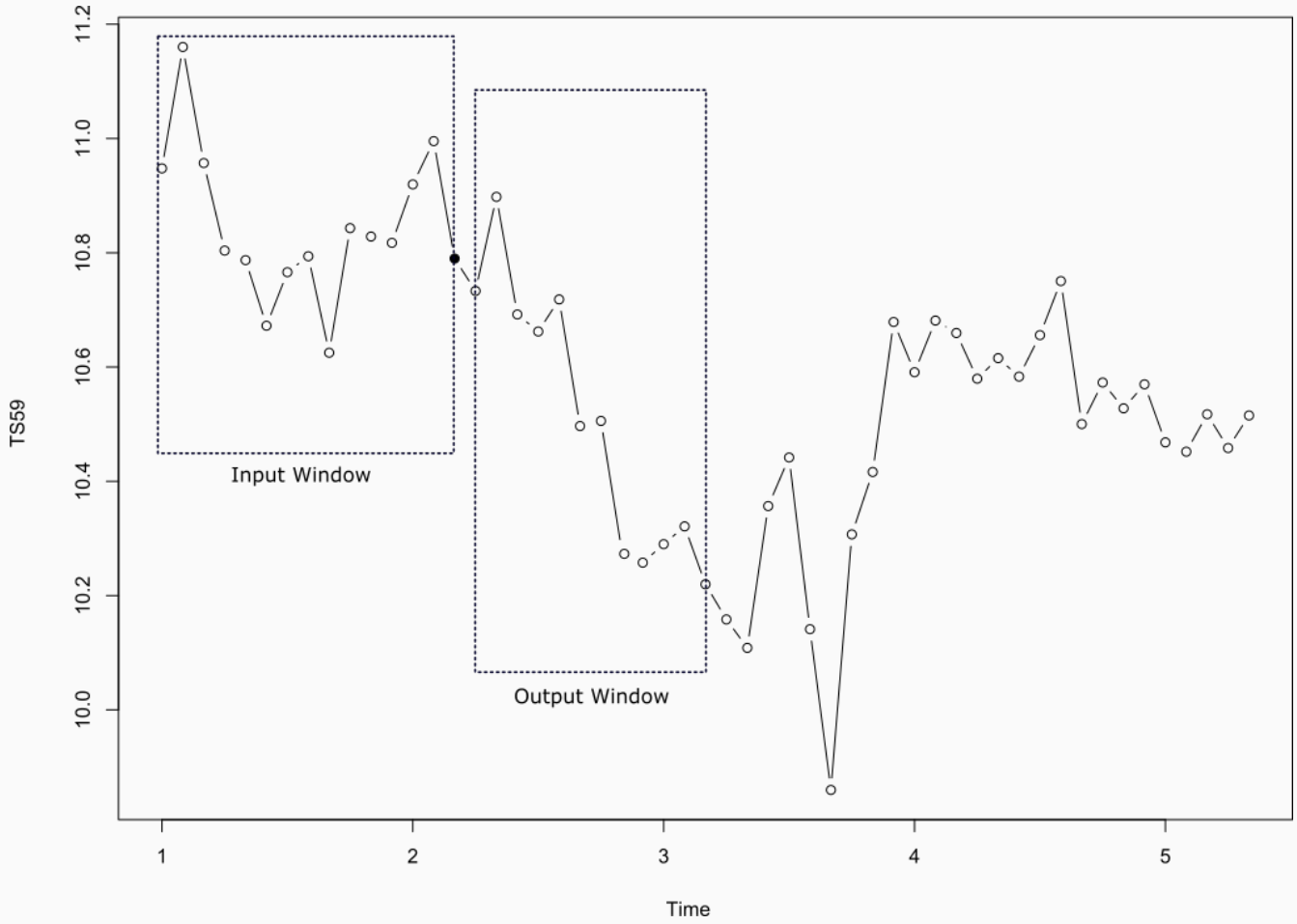
- Logarithm or Box-Cox transform
  - makes exponential trends linear
  - also stabilises the variance
  - Box-Cox Transformation

$$w_t = \begin{cases} log(y_t) & \text{if } \lambda = 0, \\ (y_t^\lambda - 1)/\lambda & \text{if } \lambda \neq 0 \end{cases}$$

  - Choice of optimal value for λ is difficult

Window-wise normalisation

Smyl and Kuber (2016);Bandara et al. (2017)

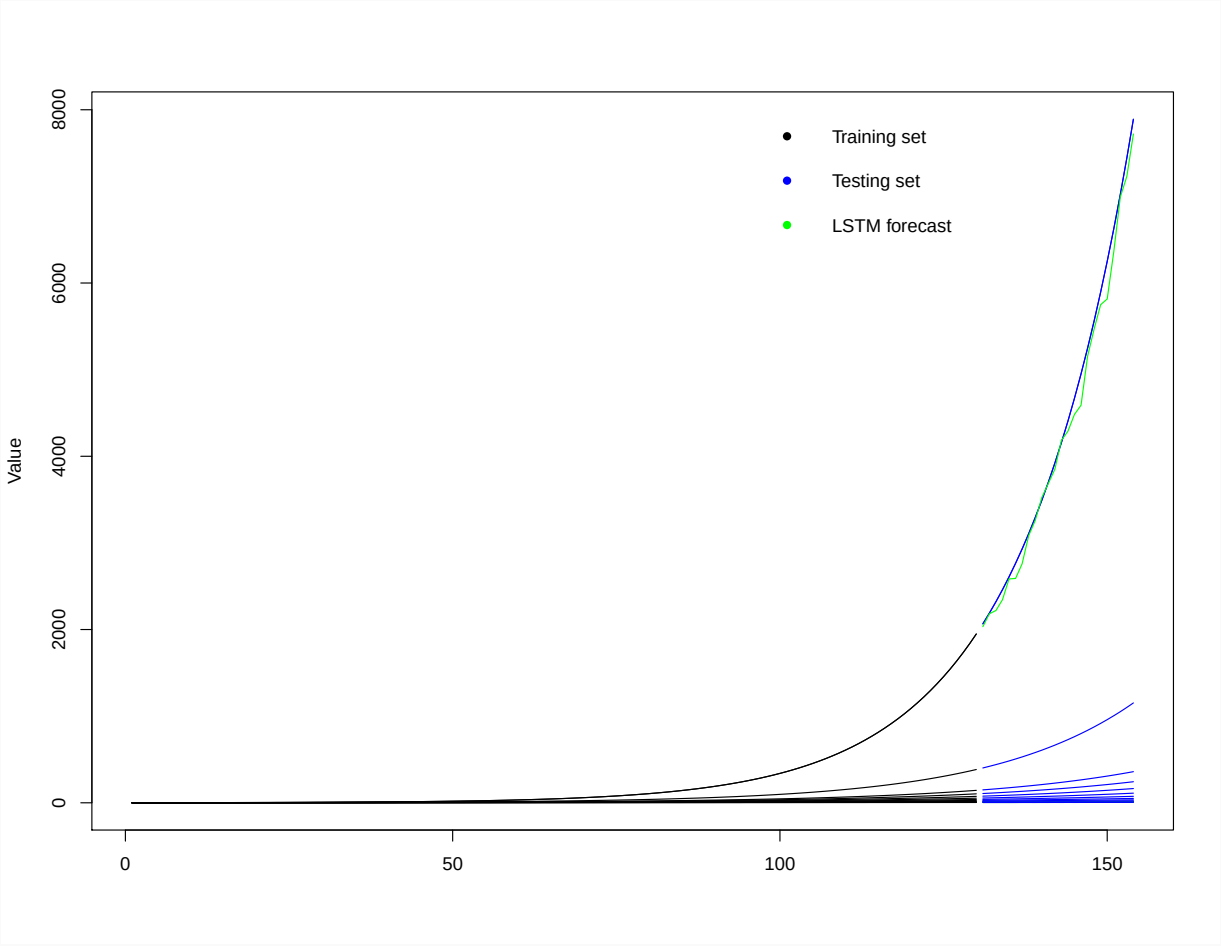


Window-wise normalisation (cont'd)

- To avoid saturation issues of sigmoid, tanh activation functions
- Similar to batch normalization
- Instead of saturating on the absolute values of the training data, it is saturating on the absolute value of the steepness of the trend in the training data
- It is usually a good idea for forecasts to be conservative.

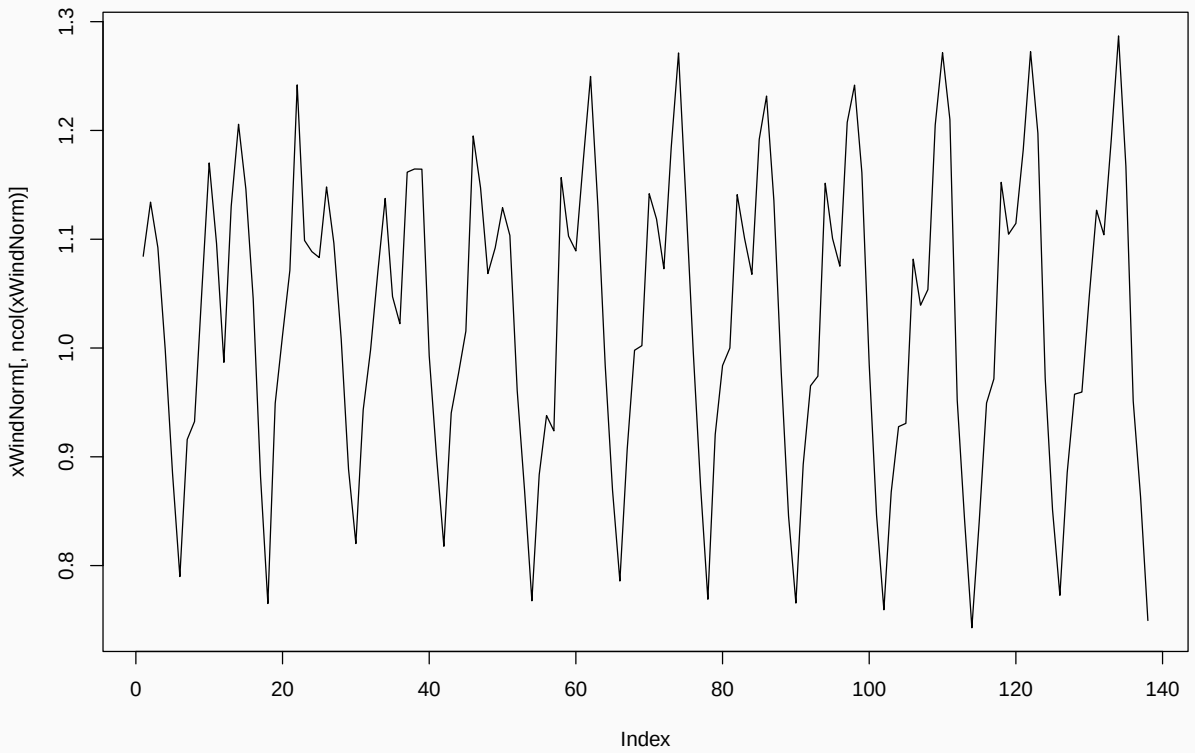
Window-wise normalisation (cont'd)

Such a system is able to predict even steep trends (Bandara et al., 2020b)



Window-wise normalisation (cont'd)

```
xEmb <- myEmb(AirPassengers, 7, 7, h=2)
xWindNorm <- t(apply(xEmb, 1, function(y) {
                    y/mean(y[1:(length(y-1))])}))
plot(xWindNorm[,ncol(xWindNorm)], type="l")
```



Window-wise normalisation (cont'd)

- Has been reinvented for deep learning under the name RevIN (Reversible Instance Normalization)
- RevIN has a couple of trainable parameters
- no ablation study as to if this actually improves the method, or if its main advantage comes from instance normalization

Expert knowledge about trends

- be careful with strong trends
- exponential trends will slow eventually
  - how much more can Facebook grow until every person on earth has 5 accounts each?
- even a linear trend is a very bold assumption if extrapolated far enough into the future
- damped trends: often not justified from the data, just to be conservative about the forecasting
- forecasts should always be conservative

How to model seasonality?

- Some discussion in the literature whether a NN can model seasonality directly or not
- Early works suggest that NNs can model seasonality (Sharda and Patil, 1992; Tang et al., 1991).
- Later, works suggest that deseasonalization is necessary (Claveria and Torra, 2014; Zhang and Qi, 2005; Zhang and Kline, 2007; Nelson et al., 1999).
- Latest findings: Machine Learning models can model seasonality well *if they have enough data* (Bandara et al., 2020a)
- Simple experiment: Generate a sine wave, let an NN learn it. How many full periods to learn it? 2 full periods, 20 full periods?

–> Assumption in forecasting is usually that we know the seasonality beforehand and that it is valid to extrapolate it infinitly into the future.

Modelling seasonality

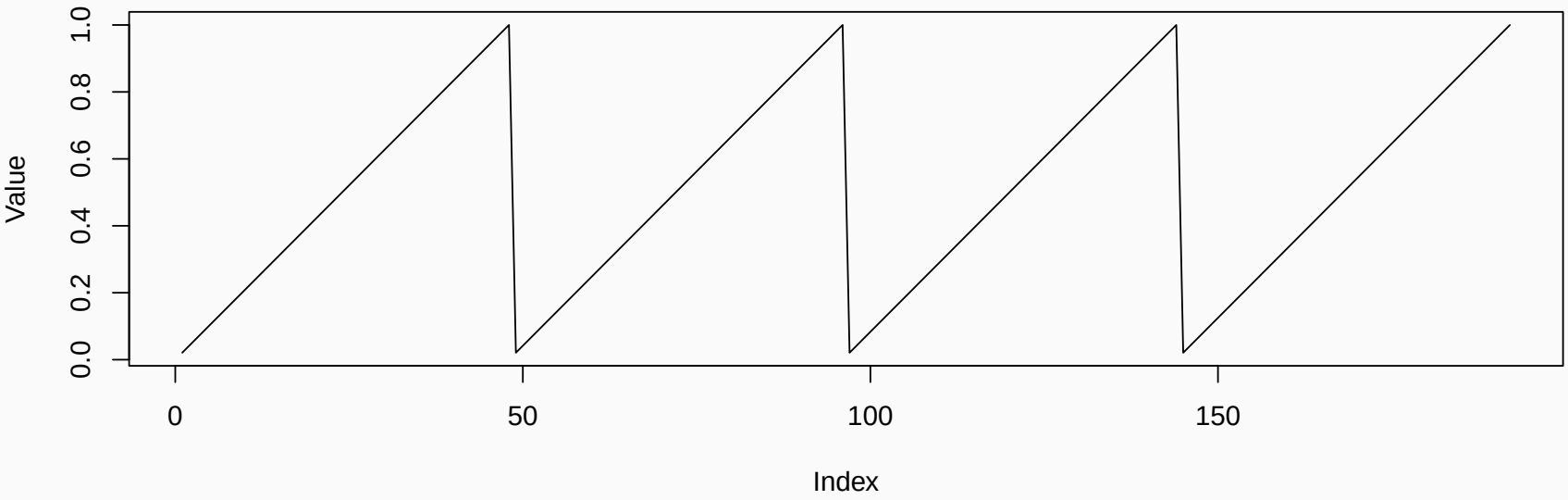
Seasonal indicator variables:

- Categorical variable: Monday, Tuesday, Wednesday, ...
- One-hot encoded version of this variable: "Seasonal dummy"

Problems if there are many seasons.

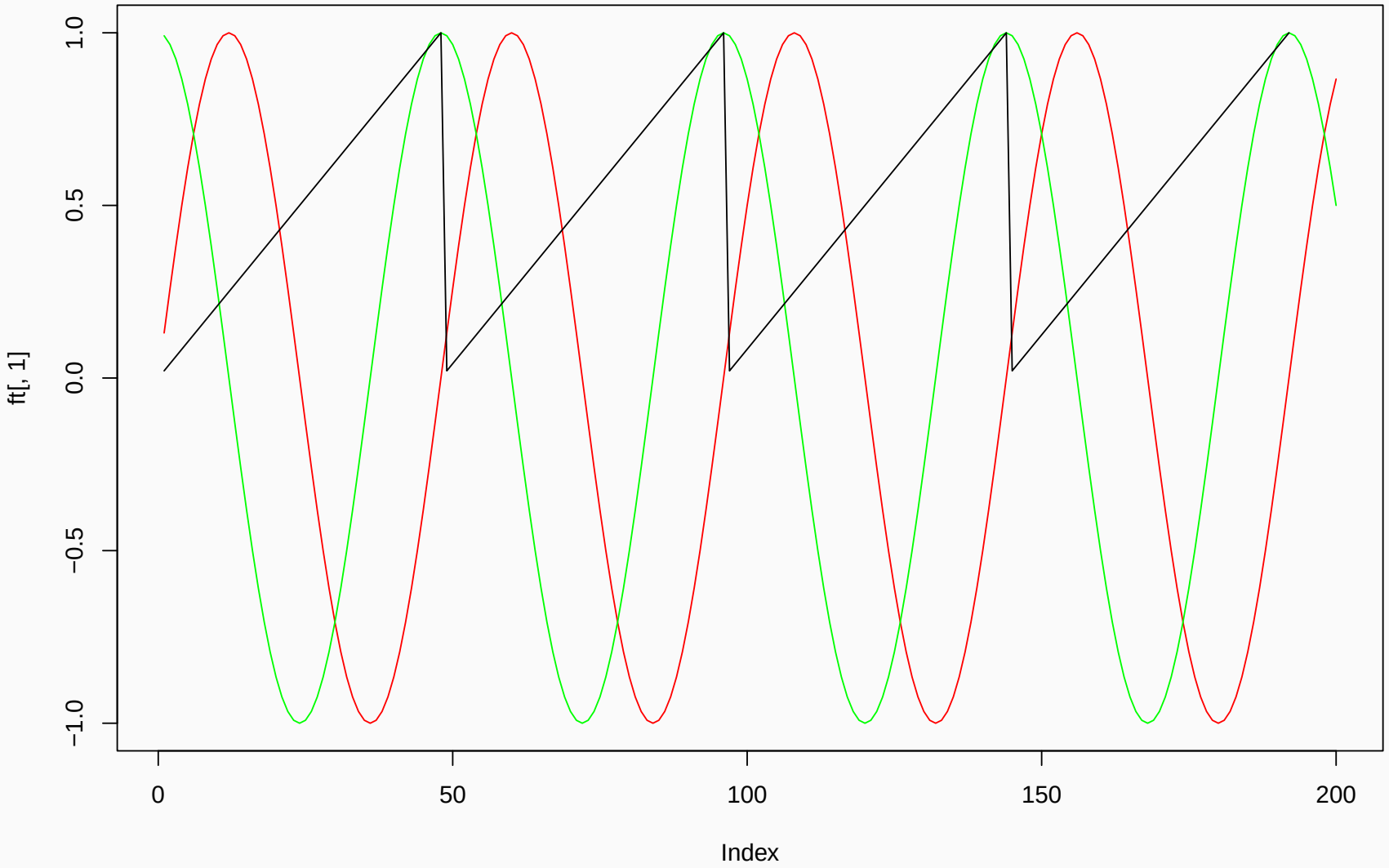
Modelling seasonality (2)

Continuous seasonal indicators



Problem: Abrupt changes. December much more distant from January than, e.g., January from February.

Modelling seasonality (3)



Fourier terms

see, e.g., Hyndman and Athanasopoulos (2018)

$$sin\left(\frac{2\pi kt}{s}\right), cos\left(\frac{2\pi kt}{s}\right)$$

t is the time point

s is the seasonal periodicity of the time series and

k is the number of sine cosine pairs used with the transformation

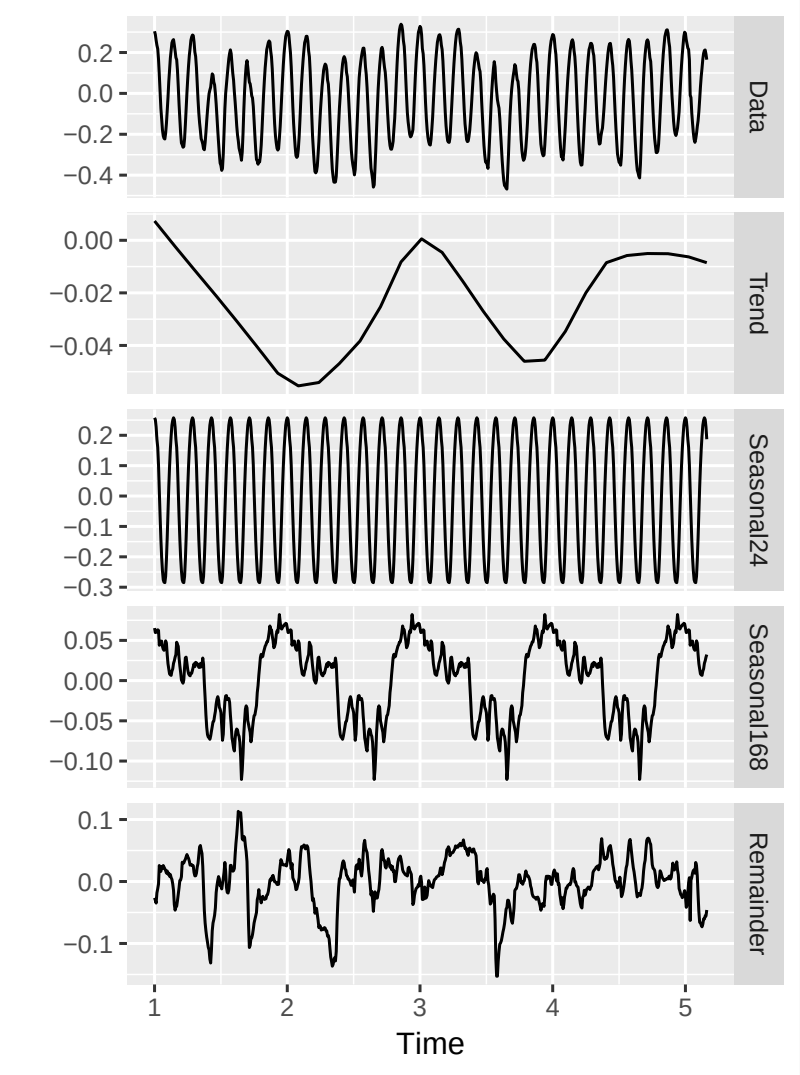
The number of Fourier terms controls the smoothness of the seasonal pattern.

Deseasonalisation

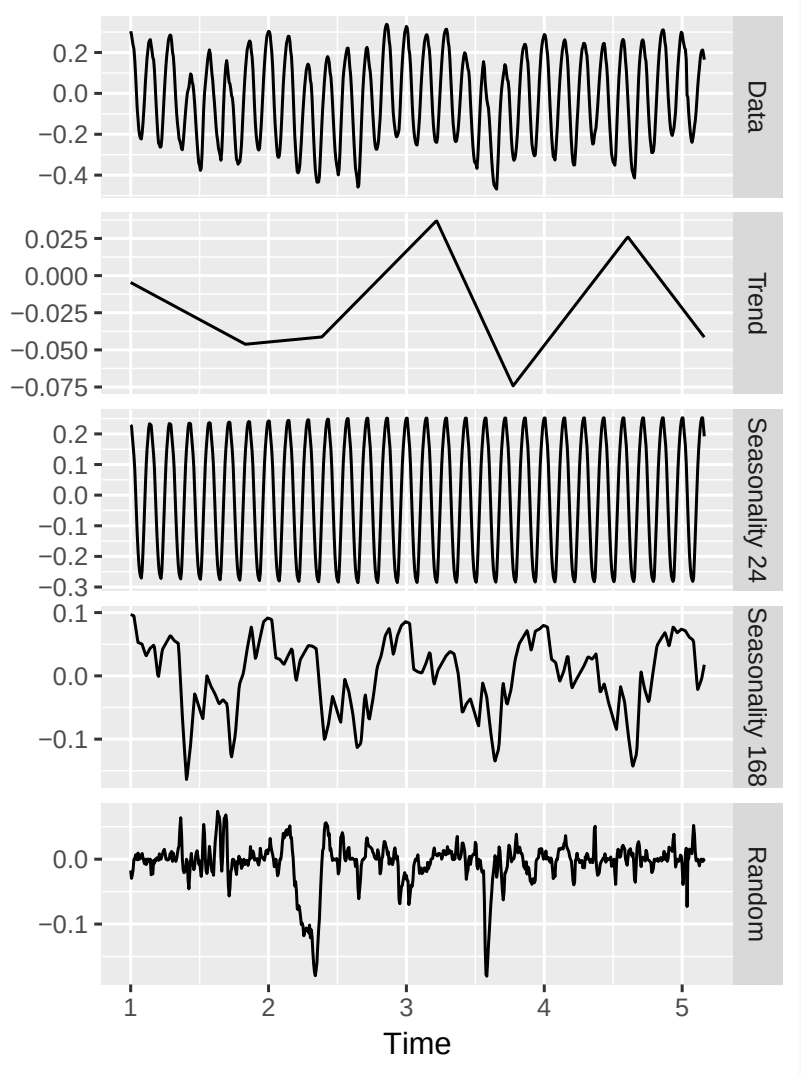
Decomposition methods such as

- STL (Cleveland et al., 1990)
- MSTL(Hyndman and Athanasopoulos, 2018)
- STR (Dokumentov and Hyndman, 2020)

Deseasonalisation (cont'd)



(a) MSTL Decomposition

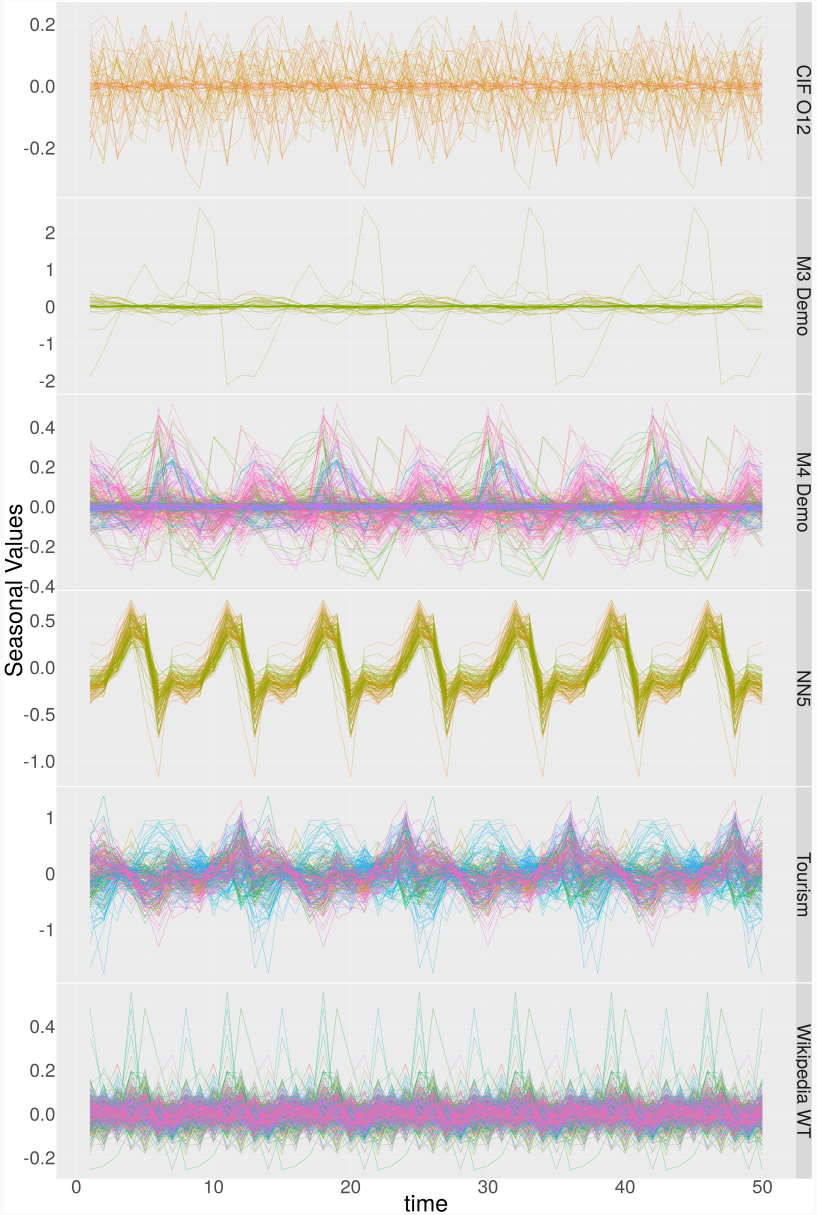


(b) STR Decomposition

Deseasonalisation (cont'd)

- we can deseasonalise and then only feed the trend and remainder component into the ML algorithm
- can be seen as a form of boosting (weak learner to model seasonality, ML model trained on residuals)
- effectively putting expert knowledge into the model
- works well if not enough data to model seasonality directly, or if seasonal components are very different between series

Deseasonalisation (cont'd)



Further considerations about seasonality

- M3, M4 datasets have different types of seasonalities mixed together, and series are not aligned
- In the M4, global models were customised for this particularity, e.g. the winner ES-RNN models seasonality for each series separately
- in real-world datasets this will normally not be the case
- seasonal indicators and Fourier terms need aligned series

–> In real-world datasets, with enough data, seasonal indicators and Fourier terms will work better than deseasonalisation

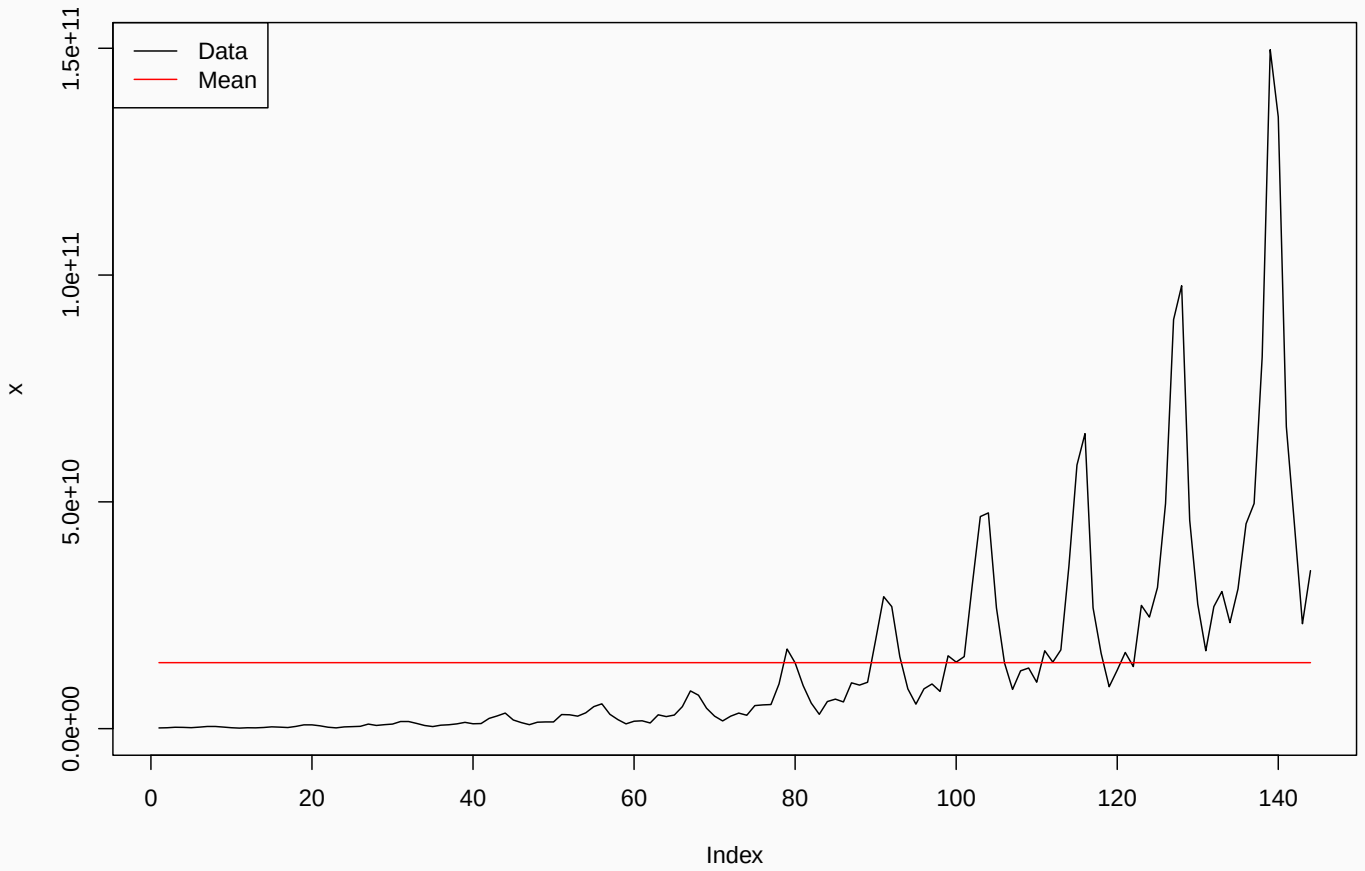
Normalisation

- in some forecasting problems, the forecasts are within a pre-specified domain (wind speed, wind power, electricity price)
- in others, they are not (share prices, web page hits, businesses that grow fast, like ride share applications, social networks, etc.)
- if the domain is limited, the scale has information (if you are at zero, you know the value will not be able to drop more)
- if the level is already high, trends tend to be less steep

–> include information about scale as additional input, if normalising

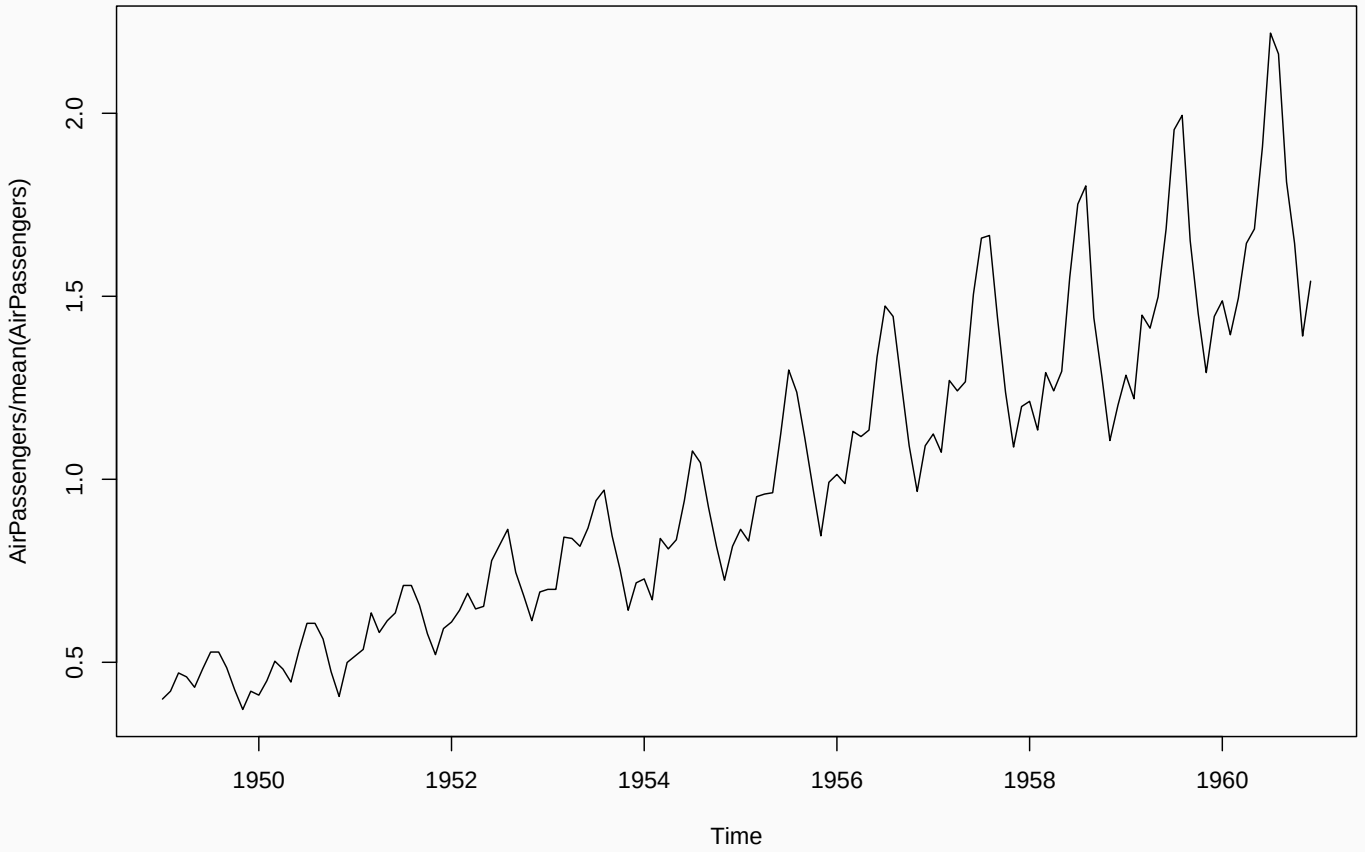
Normalisation (cont'd)

- Mean normalisation, if series don't have a strong trend
- Not a good idea if there are strong trends/level shifts:



Normalisation (cont'd)

AirPassenger series divided by its mean:



Direct output versus iterative 1-step-ahead

- Traditional methods (ETS, ARIMA) optimise 1-step-ahead accuracy
- forecasts further out are obtained by iteration, by feeding back the forecasts as inputs into the model
- this can lead to error accumulation
- usually better to predict directly all horizons needed (Ben Taieb et al., 2012; Wen et al., 2017)

Direct output versus iterative 1-step-ahead (cont'd)

- methods such as NNs can have multiple outputs
  $\Rightarrow$ output windows
- other methods, such as GBT, can produce only one output
  $\Rightarrow$ either iterate out or train a separate model per horizon
- If horizon is too long, iteration may be the only feasible option
- M5 winning method was an ensemble of direct and iterative LightGBM models

Gradient boosted trees (GBT) for forecasting

- Successful in several forecasting competitions (GEFCom 2014: Landry et al. (2016), Kaggle competitions)
- M5 competition dominated by LightGBM
- Catboost (Prokhorenkova et al., 2018) performs *ordered* gradient boosting, especially suitable for time series
- Seasonality and trend handling as discussed earlier
- build one model per horizon
- or, if dataset is small, have a feature that indicates the horizon (makes training data h times bigger)
- especially suitable with (diverse) external variables
- engineer features such as rolling means, rolling sds
- use differences as additional inputs (of different lags, e.g., lag1 difference, lag 12 difference)

Further feature engineering

- Holiday effects
  - one-hot encoded
  - as distance maps (days before/after holiday)
- Monthly series: number of trading days in the month
- Promotions
- Out-of-stock events
- Weather
- ...

Ensembling and forecast combination

- Ensembling works in forecasting just as well as in any other area
- Heavily used, e.g., in Kaggle competitions
- Ensembles of GBTs, NNs, pooled regression
- Ensembles of local and global models

Forecast combination

- Seminal paper by Bates and Granger (1969)
- Show that combining forecasts often leads to better accuracy
- Widely accepted and adopted in the forecasting field since then
- Simple average is hard to beat, though many more sophisticated methods exist

Software (selection)

MLForecast:

- `https://nixtlaverse.nixtla.io/mlforecast`
- Scales well to large datasets

skforecast:

- `https://skforecast.org/`
- Beginner-friendly

kats:

- `https://github.com/facebookresearch/Kats/blob/main/kats/models/ml_ar.py`
- A LightGBM wrapper that I implemented with all functionality discussed here

Deep learning for forecasting

Overview: Deep learning for forecasting

- Usually, recent NLP research (LSTM, attention, transformers, GPTs, ...) is adapted to the time series use case
- Main differences to NLP:
  - most recent observations are the most important ones
  - long-term dependencies are relatively simple and stable (only seasonalities: daily, weekly, yearly)
  - much more noise and uncertainty
  - much less (public) training data

Recurrent neural networks

- Overview by Hewamalage et al. (2021b)
- Have an internal state which allows them to memorize
- They have problems to learn long memory (Pascanu et al., 2013)
- LSTM mitigates that to a certain extent
- They still work better in practice if used with input and output windows (Hewamalage et al., 2021b)
  - making them more "autoregressive''
  - making the state less important
- Some literature suggests that in general RNNs that can be trained and are stable can be well approximated by feed-forward networks (Miller and Hardt, 2018; Miller, 2018)

Convolutional neural networks

- Some results suggest that CNNs work just as well as RNNs for forecasting, but are a lot faster to train (Borovykh et al., 2018)
- WaveNet (Oord et al., 2016; Sen et al., 2019): causal convolutions, dilations
- CNNs don't have a state, so windowing needs to cover everything

$\rightarrow$ Long input windows, especially with dilations

$\rightarrow$ RNNs are preferrable if input windows need to be small, e.g., across series of different lengths in a global model.

$\rightarrow$ That's also still true with Transformers

Specialised architectures

- DeepAR (Flunkert et al., 2017): Generative RNN model
- Deep state space models (Rangapuram et al., 2018): Parametrizes a linear state-space model with an RNN
- Deep Factors for forecasting (Wang et al., 2019): Combines a local probabilistic model and a global time series that is a linear combination of factors
- NBEATS (Oreshkin et al., 2019): Decomposes series into basis functions, residual stacking
  - State-of-the-art accuracy on M4
  - 2nd place in M5 (as part of an ensemble)

Transformers for forecasting

- Early works: Transformers for forecasting (Li et al., 2019), Temporal Fusion Transformers (TFT) (Lim et al., 2019)
- TFT addresses specifically common time series problems such as how to incorporate static and dynamic past and known future covariates into transformers, obtain prediction intervals etc.
- Reported to work well in practice in many situations
- TFT was already in existence at the time of the M5, but not used by any winning team. Maybe data of M5 was too intermittent?

Transformers for forecasting (2)

- Since 2019: Many variants
- Usually have plausible ideas of why they should work
- Informer: uses sparse attention to reduce quadratic complexity for long sequences
- Autoformer: integrates autocorrelation, as well as trend and seasonality decomposition
- FEDformer: uses frequency-domain transforms to model periodicity and reduce computational cost
- Pyraformer: uses a pyramidal multi-resolution strategy
- PatchTST: uses patches from the series instead of single values

Transformers for forecasting (3)

Many of them:

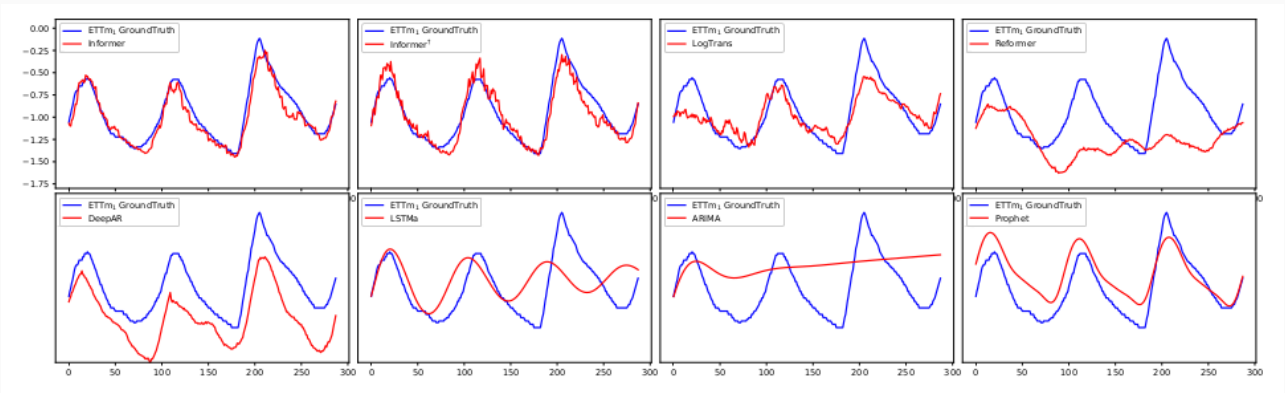
- Solve "long sequence time-series forecasting," which to me seems somewhat an invented problem
- Evaluate on the same small amount of datasets, which only represent a small subset of forecasting use cases
- Use an exchange rate dataset and fail to benchmark against naive. We have shown that they lose (Hewamalage et al., 2023)
- Fail to understand that their exchange rate dataset only contains trading days, which means the seasonality shifts around quite arbitrarily, due to the omission of bank holidays
- Lose against linear models, if those are trained directly for all horizons, as the transformers are (Zeng et al., 2023)

For details, check my talk here:

`https://cbergmeir.com/talks/neurips2024/`

Informer (AAAI outstanding paper award in 2021, 4069 citations): Benchmarks

A plot from the appendix of the original paper:



Informer: Benchmarks (2)

- The benchmarks are grotesquely misspecified
- Don't capture seasonalities at all
- What would be a reasonable benchmark?
- ARIMA and ETS only do single seasonalities and are intended mostly for monthly, quarterly, yearly data
- Let's try a linear regression onto Fourier terms: Dynamic Harmonic Regression (DHR)
- DHR with ARIMA errors, DHR-ARIMA

Informer: Benchmarks (3)

Our runs with DHR-ARIMA (a standard method for such a task and dataset):

Model	$ETTh_1$ (720)		ECL (960)	
	MSE	MAE	MSE	MAE
Informer	0.269	0.435	0.582	0.608
Informer†	0.257	0.421	0.594	0.638
LongTrans	0.273	0.463	0.624	0.645
Reformer	2.112	1.436	7.019	5.105
LSTMa	0.683	0.768	1.545	1.006
DeepAR	0.658	0.707	0.657	0.683
ARIMA	0.659	0.766	1.370	0.982
Prophet	2.735	3.253	6.901	4.264
DHR-ARIMA	**0.140**	**0.297**	**0.433**	**0.499**

Hewamalage, H., Ackermann, K., & Bergmeir, C. (2023). Forecast evaluation for data scientists: common pitfalls and best practices. Data Mining and Knowledge Discovery, 37(2), 788-832.

PatchTST and other new methods

- Qiu et al. (PVLDB, 2024): PatchTST evaluates using a "Drop Last trick"
- Seems that new methods (e.g., UniTime, iTransformer) compare against their own reruns, doing things like fixing the input window length to 96 for "fairness"

$\rightarrow$ The input window length is a hyperparameter

$\rightarrow$ Restricting to small input lengths is favouring more complex models over simpler ones

Transformers for forecasting (4)

Zalando has reported a transformer architecture they have in production (Kunz et al., 2023). Works well, trains on over 350k series.

$\rightarrow$ If you have large amounts of data, Transformers should work.

$\rightarrow$ If you have small amounts of data, there are strong competitors that will require less computation.

Multivariate forecasting

- Machine learners have invented the name "channel-dependent" for this
- Not the same as global models: series need to be aligned, and often have the same length (see also forecastingdata.org)
- Series can influence each other: Cannibalisation and substitution effects, etc.
- The holy grail in retail demand forecasting?
- Difficult and very active research area. Main problems:
  - How to scale methods to thousands of time series?
  - How to have series (products) come and go?
  - How to model changing and complex relationships?
- Usually sparseness constraints: Each series can interact only with few other series

Multivariate forecasting (cont'd)

- Yu et al. (2016): Matrix factorization with temporal regularization
- Lai et al. (2018), LSTNet: CNN feeding into RNN with skip connections
- Sen et al. (2019): Matrix factorization, causal convolutions and attention
- Salinas et al. (2019a): Gaussian copulas
- Wu et al. (2020), Sriramulu et al. (2023): Graph neural networks

Multivariate forecasting (cont'd)

Newer works:

- CrossFormer: multivariate, explicitly models both temporal (cross-time) and variable-wise (cross-dimension) dependencies via a two-stage attention mechanism
- Zhou et al. (2024): transformer that uses sparse relation matrices, a reindexing training scheme, and 2D attention to manage noise and computational cost while capturing inter-channel dependencies
- iTransformer: "inverts" the usual transformer structure: applies attention over variates instead of time tokens, enabling better multivariate correlation modeling and scalability

Software

NeuralForecast:

- `https://nixtlaverse.nixtla.io/neuralforecast`
- Library from Nixtla with implementations of many deep-learning architectures

GluonTS:

- `https://ts.gluon.ai/`
- Python library from Amazon that implements many deep-learning architectures

Data repositories and benchmark runs

ForecastingData.org:

- `https://forecastingdata.org/`
- Data repository with many forecasting datasets
- Code and results of statistic, traditional machine learning, and deep learning methods

GIFT-Eval:

- `https://huggingface.co/spaces/Salesforce/GIFT-Eval`
- Data repository that includes the Monash one and adds some more
- Leaderboard

fev-bench:

- `https://huggingface.co/spaces/autogluon/fev-bench`
- Another large repository and benchmark from Amazon

Summary: Deep Learning models

- There are some good methods out there, like PatchTST, TIDE, TFT, and others.
- They are certainly worth exploring

But:

- Don't believe everything you read in these papers
- It depends greatly on the dataset characteristics if the models will work (large amounts of series and/or large amounts of observations per series).
- Check our own evaluations at forecastingdata.org where we ran many of these methods on different datasets

Foundational models

Main idea

- Idea foundational models: pretrain on a large time series dataset

$\rightarrow$ Once pretrained, they work on single time series, dataset size is not an issue anymore

- Global and foundational models are conceptually the same, the difference is in the evaluation regime
- Evaluate on all series you trained with, vs evaluate only on a small subset

$\rightarrow$ Any global model can be used as a foundational model

$\rightarrow$ Global model guarantees do not hold, series need to be "relevant/related" for it to work

Foundational models (2)

- Some train a time series model from scratch, some use an LLM directly
- Most train on our repository from forecastingdata.org
- Using LLM directly: tokenize, convert time series to text (e.g., TimeLLM)

Foundational models (3)

Many of the earlier models have again not great evaluations:

- For example TimeLLM loses against the most simple methods like versions of naive on the M3 quarterly data
- TimesNet, TimeMixer and many others claim SOTA on the M4, when they do not outperform the original competition winners

Bergmeir, C. (2024) LLMs and Foundational Models: Not (Yet) as Good as Hoped. In: Foresight: The International Journal of Applied Forecasting, (73)

(paper is available from my personal web page, a first version was published as a LinkedIn post)

Foundational models (4)

NeurIPS 2024 spotlight paper: "Are Language Models Actually Useful for Time Series Forecasting?"

- "popular LLM-based time series forecasters perform the same or worse than basic LLM-free ablations, yet require orders of magnitude more compute"
- Basically an ablation study that papers like TimeLLM should have done in the first place

Foundational models (5)

But evaluation is getting better. Newer models evaluate on GIFT-Eval

- A large dataset, contains all of the Monash repository plus some more
- 23 datasets covering 144,000 time series and 177 million data points across 7 domains and 10 frequencies
- Supports prediction tasks ranging from short-term to long-term forecasting, with both univariate and multivariate settings
- hosted on Huggingface, and new methods routinely submit forecasts
- `https://huggingface.co/spaces/Salesforce/GIFT-Eval`

Foundational models (6)

- TimeGPT: proprietary from Nixtla, Transformer-based
- Chronos: encoder-decoder transformer, tokenizes series
- Chronos-Bolt: uses patching, predicts directly future quantile distributions
- Chronos 2: Can incorporate external variables
- TimesFM: purely decoder-based transformer
- TimesFM2.5: fewer parameters, longer context length
- Moirai: masked-encoder transformer
- Lag-llama: combines LLaMA's architecture with time series features
- TinyTimeMixers (TTM): MLP-based model, combines adaptive patching, resolution prefix tuning, and multi-level channel modeling

Fundamental Problems in Foundational models

- Global models: Average performance is improved at the expense of performance on particular series
- If a paper evaluates broadly an average performance across many domains, it is effectively a global model and covered by the theory
- If you later use this model for your particular use case, you don't have these guarantees
- We are back to the idea that the series in training need to be related to your series in testing
- Depending on the dataset, you can fine-tune a foundational model or you may be better off just building your own model from scratch.

$\rightarrow$ Some of these methods achieve quite competitive results, not beating everybody, but giving you a good baseline with minimal needed forecasting knowledge.

$\rightarrow$ Need to evaluate the model **on your data** with a hold-out set, comparing to benchmarks

Let's say I have a single yearly time series, which model should I use?

Let's look into the Chronos paper to find out:

	Pretrained Models (Zero Shot)							Pretrained Models (Other)			Task Specific Models									
	Chronos-T5 (Large)	Chronos-T5 (Base)	Chronos-T5 (Small)	Chronos-T5 (Mini)	Chronos-GPT2	LLMTime	ForecastPFN	Lag-Llama	Moirai-1.0-R (Base)	Moirai-1.0-R (Large)	PatchTST	DeepAR	WaveNet	TFT	DLinear	N-HiTS	N-BEATS	GPT4TS	SCUM	AutoETS
Australian Electricity	1.333	1.319	1.399	1.114	1.310	1.186	2.158	1.635	1.258	1.009	0.871	1.473	0.997	**0.810**	1.278	**0.794**	0.828	1.161	1.427	2.391
Car Parts	0.906	0.899	0.887	0.891	0.881	-	2.657	0.816	1.735	1.542	0.803	**0.798**	0.817	**0.799**	0.879	0.803	0.803	0.891	1.157	1.185
CIF 2016	0.986	0.981	0.989	1.051	1.046	1.384	3.588	2.235	1.197	1.160	1.537	1.363	1.309	1.553	1.145	1.389	1.440	0.960	**0.907**	**0.957**
Covid Deaths	42.550	42.687	42.670	43.621	48.215	32.143	91.515	78.456	33.062	33.108	36.465	38.203	102.457	**30.635**	40.418	31.771	31.730	75.909	33.595	38.114
Dominick	0.818	0.816	0.819	0.833	0.820	-	3.274	1.250	0.879	0.845	0.867	0.851	0.812	0.800	0.880	**0.782**	**0.782**	1.813	0.891	0.885
ERCOT Load	0.617	**0.550**	0.573	0.588	0.561	1.319	3.975	0.834	0.583	0.667	**0.553**	1.197	0.780	0.690	0.651	0.615	0.648	0.558	1.308	2.826
ETT (15 Min.)	0.741	0.739	0.710	0.792	0.796	1.042	1.138	0.967	0.981	0.753	0.652	0.874	1.339	0.962	0.724	0.643	0.659	**0.574**	0.673	1.183
ETT (Hourly)	0.735	0.789	0.789	0.797	0.768	1.232	1.833	1.002	0.902	0.845	**0.729**	0.814	1.509	0.875	**0.695**	0.811	0.782	0.768	0.850	1.139
Exchange Rate	2.375	2.433	2.252	2.030	2.335	1.743	7.583	3.087	**1.507**	1.909	1.540	1.615	3.105	2.361	**1.459**	2.041	2.149	2.709	1.749	1.643
FRED-MD	0.500	0.486	0.496	0.483	**0.468**	0.513	2.621	2.283	0.607	0.593	0.745	0.621	0.849	0.929	0.713	0.696	0.635	0.693	0.492	0.544
Hospital	0.810	0.810	0.815	0.817	0.831	0.861	1.775	0.939	0.821	0.826	0.859	0.804	0.857	0.799	0.940	0.781	**0.760**	0.793	**0.748**	0.760
M1 (Monthly)	1.090	1.117	1.169	1.174	1.182	1.415	2.172	1.875	1.272	1.238	1.208	1.122	1.266	1.326	1.369	1.333	1.236	1.198	**1.023**	**1.072**
M1 (Quarterly)	1.713	1.739	1.764	1.785	1.785	1.802	9.931	3.036	1.896	1.840	1.920	1.741	1.904	2.144	1.943	2.061	2.043	1.958	**1.602**	1.710
M1 (Yearly)	4.301	4.624	4.659	4.958	4.751	4.077	23.089	7.149	4.623	4.708	4.042	3.685	4.727	4.316	11.565	5.568	6.212	**3.675**	**3.571**	4.110
M3 (Monthly)	**0.857**	0.868	0.885	0.900	0.930	0.996	2.240	1.846	0.946	0.924	1.225	0.943	0.950	0.916	1.161	0.899	0.883	0.950	**0.827**	0.869
M3 (Quarterly)	1.181	1.199	1.256	1.289	1.241	1.450	10.176	2.886	1.428	1.429	1.264	1.209	1.257	1.160	1.572	1.202	1.147	1.448	1.135	**1.125**
M3 (Yearly)	3.106	3.209	3.276	3.385	3.158	3.140	18.728	5.114	3.661	3.822	2.949	2.827	3.026	2.860	3.435	3.432	3.547	3.418	2.703	**2.696**
M4 (Quarterly)	1.216	1.231	1.246	1.271	1.312	-	6.927	2.663	1.286	1.259	1.150	1.254	1.241	1.248	1.229	1.157	**1.129**	1.215	**1.145**	1.188
M4 (Yearly)	3.606	3.678	3.651	3.743	3.933	-	-	5.866	3.599	4.175	**3.072**	3.178	3.221	3.119	3.295	-	-	3.374	**3.013**	3.374
M5	0.944	0.939	0.940	0.944	0.969	-	1.530	0.965	1.442	0.929	0.919	0.956	0.959	**0.909**	1.027	**0.917**	0.917	0.935	1.096	1.101
NN5 (Daily)	0.573	0.585	0.615	0.642	0.601	0.953	1.375	0.992	0.698	0.625	0.575	0.585	0.585	**0.556**	0.604	**0.571**	0.571	0.720	1.052	1.039
NN5 (Weekly)	0.940	0.938	0.944	0.947	0.963	0.968	1.349	1.141	0.980	1.009	**0.877**	0.920	1.034	**0.896**	0.966	0.919	1.014	1.268	0.974	0.978
Tourism (Monthly)	1.761	1.828	1.900	1.950	1.783	2.139	4.348	3.030	2.039	1.910	1.572	1.529	1.629	1.686	1.551	1.514	**1.486**	1.573	**1.441**	1.497
Tourism (Quarterly)	1.677	1.717	1.730	1.829	1.828	1.916	5.595	3.695	2.722	2.281	1.723	1.586	1.769	1.729	1.690	**1.585**	1.618	1.750	**1.501**	1.590
Tourism (Yearly)	3.755	3.900	3.901	4.048	3.862	3.309	12.093	3.755	**3.047**	3.301	3.138	3.702	4.130	**3.047**	3.406	3.448	3.564	-	3.276	3.138
Traffic	0.804	0.828	0.837	0.850	0.818	0.973	1.909	0.829	**0.726**	0.759	0.790	**0.737**	0.797	0.880	0.821	0.927	0.968	0.787	-	1.685
Weather	**0.822**	0.824	0.836	0.853	0.858	-	2.003	1.001	0.831	**0.807**	0.860	0.911	0.945	0.913	0.997	0.910	0.888	0.972	0.933	1.079
Agg. Relative Score	**0.823**	0.832	0.841	0.850	0.852	0.962	2.450	1.291	0.907	0.876	**0.810**	0.843	0.951	0.847	0.894	0.830	0.835	0.895	0.838	0.953
Avg. Rank	8.481	9.296	10.593	12.037	11.630	16.593	23.204	19.667	13.037	12.444	**8.222**	9.111	14.074	9.778	12.704	9.463	9.648	12.111	**8.204**	10.704

SCUM: Fotios Petropoulos and Ivan Svetunkov. A simple combination of univariate models. International journal of forecasting, 36(1):110–115, 2020. 10, 43

Let's have a look at Gift-Eval

- Gift-Eval is dominated by high-frequent data
- ARIMA, ETS do not handle multiple seasonalities or long seasonal periods, are not designed for high-frequent data
- Also, they optimise towards the mean of the forecast distribution (maximum likelihood under Gaussian assumption, effectively MSE), but Gift-Eval evaluates for the median (MASE, thus MAE)
- What if we look at the data they are designed for?

Let's have a look at Gift-Eval (2)

T	model	Organization	Test Leak.	Replication Code	Monthly (MASE)	Q
🟢	timesfm_2_0_500m (code)	Google Research	Yes	Yes	0.7	0
🔷	xLSTM-Mixer	AIML Lab @ TU Darmsta	No	No	0.701	0
🟣	TTM-R2-Finetuned (code)	IBM Research	Yes	Yes	0.74	0
🔴	DeOSAlphaTimeGPTPredictor-2025	vencortex®	No	No	0.748	0
🟢	Chronos-2 (code)	AWS	No	Yes	0.756	0
🟡	Kairos-1.0	ContinualIST	No	No	0.759	0
🔶	Auto_Arima		No	No	0.759	0
🟡	TimeCopilot (code)		No	Yes	0.76	0
🔴	TimesFM-2.5 (code)	Google Research	No	Yes	0.76	0
🔴	TiRex (code)	NXAI	No	Yes	0.762	0
🟣	TSOrchestra-test	Melady Lab @ USC	Yes	No	0.766	0

Prior-fitted Networks (PFNs)

- A new exciting development: Pre-train on synthetic data
- Has a Bayesian motivation why this will work
- TabPFN (Hollmann et al.) for classification and regression
- TabICL (Jingang et al.)

PFNs for forecasting

- TabPFN-TS (Hoo et al., 2024): Regular TabPFN trained for regression, with time series specific inference
- ForecastPFN (Dooley et al., 2024): Time-series-specific DGP, again quite dodgy evaluation (only evaluate on one time series per dataset)
- TimePFN (Taga et al., 2025)
- Chronos 2

$\Rightarrow$ Achieve decent results on many forecasting tasks, no risk of data leakage

The future will likely be multimodal

- If we provide textual context for each series, the fundamental problems are addressed
- Do we want to replace the model or the modeller?

$\rightarrow$ Can ask already now ChatGPT to run ETS or ARIMA for me, replacing the modeller

$\rightarrow$ LLMs are not good with numerical computations out of the box (that's why they often automatically run Python code for numeric requests)

Again, beware: Some recent papers claim they do multimodal, but just use LLMs to extract features from numeric time series, without any additional information from the text.

Forecasting agents:

TimeCopilot: `https://timecopilot.dev/`

Chronulus.AI: `https://www.chronulus.com/`

Still more like LLM interfaces to traditional algorithms, but more integration is likely in the future.

Summary Foundation models

- Foundational models have seen an explosion of research over the recent past
- They are promising, and can be used as fully automatic baselines
- We are usually still able to outperform them.
- This is currently a problem that has no easy fix, it is inherent to their construction
- A possible fix are multimodal models and forecasting agents
- *You always need to do rigorous evaluations.*

Conclusions

- Machine learning models work well if you have long (high-frequent) series or many series from a similar source, and/or external regressors
- If you train Deep Learning models from scratch, you need usually a large (enough) dataset
- Foundational models are now a great fully-automatic baseline, but we can still beat them with bespoke modelling
- Different series need different types of forecasting (or at least different preprocessing)
- It is paramount that you perform an evaluation on your data with the right error metrics and reasonable conventional benchmarks, to be able to assess whether ML and DL should be used
- If you have low-frequent data (monthly, quarterly, years), beating the classics like ARIMA, ETS, Theta (and ensembles of them) on their home turf can still be a challenge

Thank You

https://www.cbergmeir.com

References i

K. Bandara, C. Bergmeir, and S. Smyl. Forecasting across time series databases using long short-term memory networks on groups of similar series. *arXiv preprint arXiv:1710.03222*, 8:805–815, 2017.

K. Bandara, C. Bergmeir, and H. Hewamalage. LSTM-MSNet: Leveraging forecasts on sets of related time series with multiple seasonal patterns. *IEEE Transactions on Neural Networks and Learning Systems*, (forthcoming), 2020a. URL `https://arxiv.org/pdf/1909.04293`.

References ii

K. Bandara, C. Bergmeir, and S. Smyl. Forecasting across time series databases using recurrent neural networks on groups of similar series: A clustering approach. *Expert Systems with Applications*, 140:112896, 2020b.

J. M. Bates and C. W. Granger. The combination of forecasts. *Journal of the Operational Research Society*, 20(4):451–468, 1969.

References iii

S. Ben Taieb, G. Bontempi, A. F. Atiya, and A. Sorjamaa. A review and comparison of strategies for multi-step ahead time series forecasting based on the NN5 forecasting competition. *Expert Syst. Appl.*, 39(8):7067–7083, June 2012.

C. S. Bojer and J. P. Meldgaard. Kaggle forecasting competitions: An overlooked learning opportunity. *International Journal of Forecasting*, 2020.

References iv

A. Borovykh, S. Bohte, and C. W. Oosterlee. Dilated convolutional neural networks for time series forecasting. *Journal of Computational Finance*, 2018. doi: 10.21314/jcf.2019.358. URL `https://doi.org/10.21314/jcf.2019.358`.

O. Claveria and S. Torra. Forecasting tourism demand to catalonia: Neural networks vs. time series models. *Econ. Model.*, 36:220–228, Jan. 2014.

R. B. Cleveland, W. S. Cleveland, J. McRae, and I. Terpenning. STL: A seasonal-trend decomposition procedure based on loess. *Journal of Official Statistics*, 6:3–73, 1990.

References v

A. Dokumentov and R. J. Hyndman. Str: A seasonal-trend decomposition procedure based on regression. *arXiv preprint arXiv:2009.05894*, 2020.

S. Dooley, G. S. Khurana, C. Mohapatra, S. V. Naidu, and C. White. Forecastpfn: Synthetically-trained zero-shot forecasting. *Advances in Neural Information Processing Systems*, 36, 2024.

G. T. Duncan, W. L. Gorr, and J. Szczypula. Forecasting analogous time series. In *Principles of forecasting*, pages 195–213. Springer, 2001.

References vi

V. Flunkert, D. Salinas, and J. Gasthaus. Deepar: Probabilistic forecasting with autoregressive recurrent networks. *CoRR*, abs/1704.04110, 2017. URL `http://arxiv.org/abs/1704.04110`.

A. Gelman, P. Gelman, and J. Hill. *Data Analysis Using Regression and Multilevel/Hierarchical Models*. Analytical Methods for Social Research. Cambridge University Press, 2007. ISBN 9780521686891.

References vii

H. Hewamalage, C. Bergmeir, and K. Bandara. Global models for time series forecasting: A simulation study. *Pattern Recognition*, 2021a. URL `https://arxiv.org/abs/2012.12485`.

H. Hewamalage, C. Bergmeir, and K. Bandara. Recurrent neural networks for time series forecasting: Current status and future directions. *International Journal of Forecasting*, 37(1):388–427, 2021b.

References viii

H. Hewamalage, K. Ackermann, and C. Bergmeir. Forecast evaluation for data scientists: common pitfalls and best practices. *Data Mining and Knowledge Discovery*, 37(2): 788–832, 2023. doi: 10.1007/s10618-022-00894-5.

N. Hollmann, S. Müller, K. Eggensperger, and F. Hutter. Tabpfn: A transformer that solves small tabular classification problems in a second. In *The Eleventh International Conference on Learning Representations*.

References ix

S. B. Hoo, S. Müller, D. Salinas, and F. Hutter. The tabular foundation model tabPFN outperforms specialized time series forecasting models based on simple features. In *NeurIPS Workshop on Time Series in the Age of Large Models*, 2024. URL `https://openreview.net/forum?id=ho8Yx5YfiM`.

R. J. Hyndman and G. Athanasopoulos. *Forecasting: principles and practice, 2nd edition*. OTexts, otexts.com/fpp2, 2018.

References x

T. Januschowski, J. Gasthaus, Y. Wang, D. Salinas, V. Flunkert, M. Bohlke-Schneider, and L. Callot. Criteria for classifying forecasting methods. *International Journal of Forecasting*, 36(1):167–177, 2020.

Q. Jingang, D. Holzmüller, G. Varoquaux, and M. Le Morvan. Tabicl: A tabular foundation model for in-context learning on large data. In *Forty-second International Conference on Machine Learning*.

References xi

M. Kunz, S. Birr, M. Raslan, L. Ma, Z. Li, A. Gouttes, M. Koren, T. Naghibi, J. Stephan, M. Bulycheva, et al. Deep learning based forecasting: a case study from the online fashion industry. *arXiv preprint arXiv:2305.14406*, 2023.

G. Lai, W.-C. Chang, Y. Yang, and H. Liu. Modeling long- and Short-Term temporal patterns with deep neural networks. In *The 41st International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '18, pages 95–104, New York, NY, USA, 2018. ACM.

References xii

M. Landry, T. P. Erlinger, D. Patschke, and C. Varrichio. Probabilistic gradient boosting machines for gefcom2014 wind forecasting. *International Journal of Forecasting*, 32 (3):1061–1066, 2016.

S. Li, X. Jin, Y. Xuan, X. Zhou, W. Chen, Y.-X. Wang, and X. Yan. Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. In *Advances in Neural Information Processing Systems*, pages 5243–5253, 2019.

References xiii

B. Lim, S. O. Arik, N. Loeff, and T. Pfister. Temporal fusion transformers for interpretable multi-horizon time series forecasting. *arXiv preprint arXiv:1912.09363*, 2019.

S. Makridakis, E. Spiliotis, and V. Assimakopoulos. The m5 accuracy competition: Results, findings and conclusions. 10 2020.

J. Miller. When recurrent models don't need to be recurrent, 2018. URL `http://www.offconvex.org/2018/07/27/approximating-recurrent/`. Accessed 10 November 2020.

References xiv

J. Miller and M. Hardt. Stable recurrent models. *arXiv preprint arXiv:1805.10369*, 2018.

P. Montero-Manso and R. J. Hyndman. Principles and algorithms for forecasting groups of time series: Locality and globality. *arXiv preprint arXiv:2008.00444*, 2020.

M. Nelson, T. Hill, W. Remus, and M. O'Connor. Time series forecasting using neural networks: should the data be deseasonalized first? *J. Forecast.*, 18(5):359–367, 1999.

References xv

A. v. d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

B. N. Oreshkin, D. Carpov, N. Chapados, and Y. Bengio. N-beats: Neural basis expansion analysis for interpretable time series forecasting. *arXiv preprint arXiv:1905.10437*, 2019.

R. Pascanu, T. Mikolov, and Y. Bengio. On the difficulty of training recurrent neural networks. In *International conference on machine learning*, pages 1310–1318, 2013.

References xvi

L. Prokhorenkova, G. Gusev, A. Vorobev, A. V. Dorogush, and A. Gulin. Catboost: unbiased boosting with categorical features. In *Advances in neural information processing systems*, pages 6638–6648, 2018.

S. S. Rangapuram, M. W. Seeger, J. Gasthaus, L. Stella, Y. Wang, and T. Januschowski. Deep state space models for time series forecasting. In *Advances in neural information processing systems*, pages 7785–7794, 2018.

References xvii

D. Salinas, M. Bohlke-Schneider, L. Callot, R. Medico, and J. Gasthaus. High-dimensional multivariate forecasting with low-rank gaussian copula processes. In *Advances in Neural Information Processing Systems*, pages 6827–6837, 2019a.

D. Salinas, V. Flunkert, J. Gasthaus, and T. Januschowski. Deepar: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting*, 2019b. ISSN 0169-2070.

References xviii

R. Sen, H.-F. Yu, and I. S. Dhillon. Think globally, act locally: A deep neural network approach to high-dimensional time series forecasting. In *Advances in Neural Information Processing Systems*, pages 4837–4846, 2019.

R. Sharda and R. B. Patil. Connectionist approach to time series prediction: an empirical test. *J. Intell. Manuf.*, 3(5): 317–323, Oct. 1992.

S. Smyl and K. Kuber. Data preprocessing and augmentation for multiple short time series forecasting with recurrent neural networks. In *36th International Symposium on Forecasting*, 2016.

References xix

A. Sriramulu, N. Fourrier, and C. Bergmeir. Adaptive dependency learning graph neural networks. *Information Sciences*, 625:700–714, 2023.

M. Štěpnička and M. Burda. Computational intelligence in forecasting (CIF) 2016 time series forecasting competition. In *IEEE WCCI 2016, JCNN-13 Advances in Computational Intelligence for Applied Time Series Forecasting (ACIATSF)*, 2016.

References xx

E. O. Taga, M. E. Ildiz, and S. Oymak. Timepfn: Effective multivariate time series forecasting with synthetic data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 20761–20769, 2025.

Z. Tang, C. de Almeida, and P. A. Fishwick. Time series forecasting using neural networks vs. box- jenkins methodology. *Simulation*, 57(5):303–310, Nov. 1991.

References xxi

J. R. Trapero, N. Kourentzes, and R. Fildes. On the identification of sales forecasting models in the presence of promotions. *Journal of the Operational Research Society*, 66(2):299–307, Feb 2015. ISSN 1476-9360. doi: 10.1057/jors.2013.174.

Y. Wang, A. Smola, D. C. Maddix, J. Gasthaus, D. Foster, and T. Januschowski. Deep factors for forecasting. *arXiv preprint arXiv:1905.12417*, 2019.

R. Wen, K. Torkkola, B. Narayanaswamy, and D. Madeka. A Multi-Horizon quantile recurrent forecaster. In *Neural Information Processing Systems*, Nov. 2017.

References xxii

Z. Wu, S. Pan, G. Long, J. Jiang, X. Chang, and C. Zhang. Connecting the dots: Multivariate time series forecasting with graph neural networks. *arXiv preprint arXiv:2005.11650*, 2020.

H.-F. Yu, N. Rao, and I. S. Dhillon. Temporal regularized matrix factorization for high-dimensional time series prediction. In *Advances in neural information processing systems*, pages 847–855, 2016.

References xxiii

A. Zeng, M. Chen, L. Zhang, and Q. Xu. Are transformers effective for time series forecasting? In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pages 11121–11128, 2023.

G. P. Zhang and D. M. Kline. Quarterly Time-Series forecasting with neural networks. *IEEE Trans. Neural Netw.*, 18(6):1800–1814, Nov. 2007.

G. P. Zhang and M. Qi. Neural network forecasting for seasonal and trend time series. *Eur. J. Oper. Res.*, 160(2): 501–514, 2005.

X. Zhou, W. Wang, W. Buntine, S. Qu, A. Sriramulu, W. Tan, and C. Bergmeir. Scalable transformer for high dimensional multivariate time series forecasting. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pages 3515–3526, 2024.