ADVANCED TOPICS IN DEEP LEARNING: THE RISE OF TRANSFORMER

POLITECNICO DI MILANO





Quick Intro

Prof. Manuel Roveri, Ing. Alessandro Falcetta, Ing. Diego Riva

Prof. Manuel Roveri

- **Full Professor**

Dipartimento di Elettronica, Informazione e Bioingegneria (DEIB), Politecnico di Milano, Italy
Email: **manuel.roveri@polimi.it**
Web: http://roveri.faculty.polimi.it

- **Research interests**: **Embedded and edge Artificial intelligence, forecasting as-a-service and privacy-preserving machine and deep learning**
- **Lecturer of « Computing Infrastructures» and «Hardware Architecture for Embedded and edge AI»**
- **Associate Editor** of IEEE Trans. on Artificial Intelligence, Neural Networks, IEEE Trans. on Emerging Tecnologies in Computational Intelligence, IEEE Trans. on Neural Networks and Learning Systems
- Chair of the IEEE CIS **Technical Activities** strategic planning committee and IEEE CIS **Neural Network** Technical Committee
- **Co-Founder of DHIRIA**, a Spin-Off of Politecnico di Milano

The research team

- **Alessandro Falcetta (PhD Student)**
- Massimo Pavan (PhD Student)
- Luca Colombo (PhD Student)
- Matteo Gambella (PhD Student)
- **Diego Riva (Research Assistant)**
- Gabriele Viscardi (Research Assistant)
- Simone Disabato (Former Post-doc researcher)
- Giuseppe Canonaco (Former Post-doc researcher)
- …

The WemSy Lab @ Lecco Campus of Politecnico di Milano

The research activity

Cyber-physical Systems

Artificial Intelligence (Machine learning and deep learning)

The research activity

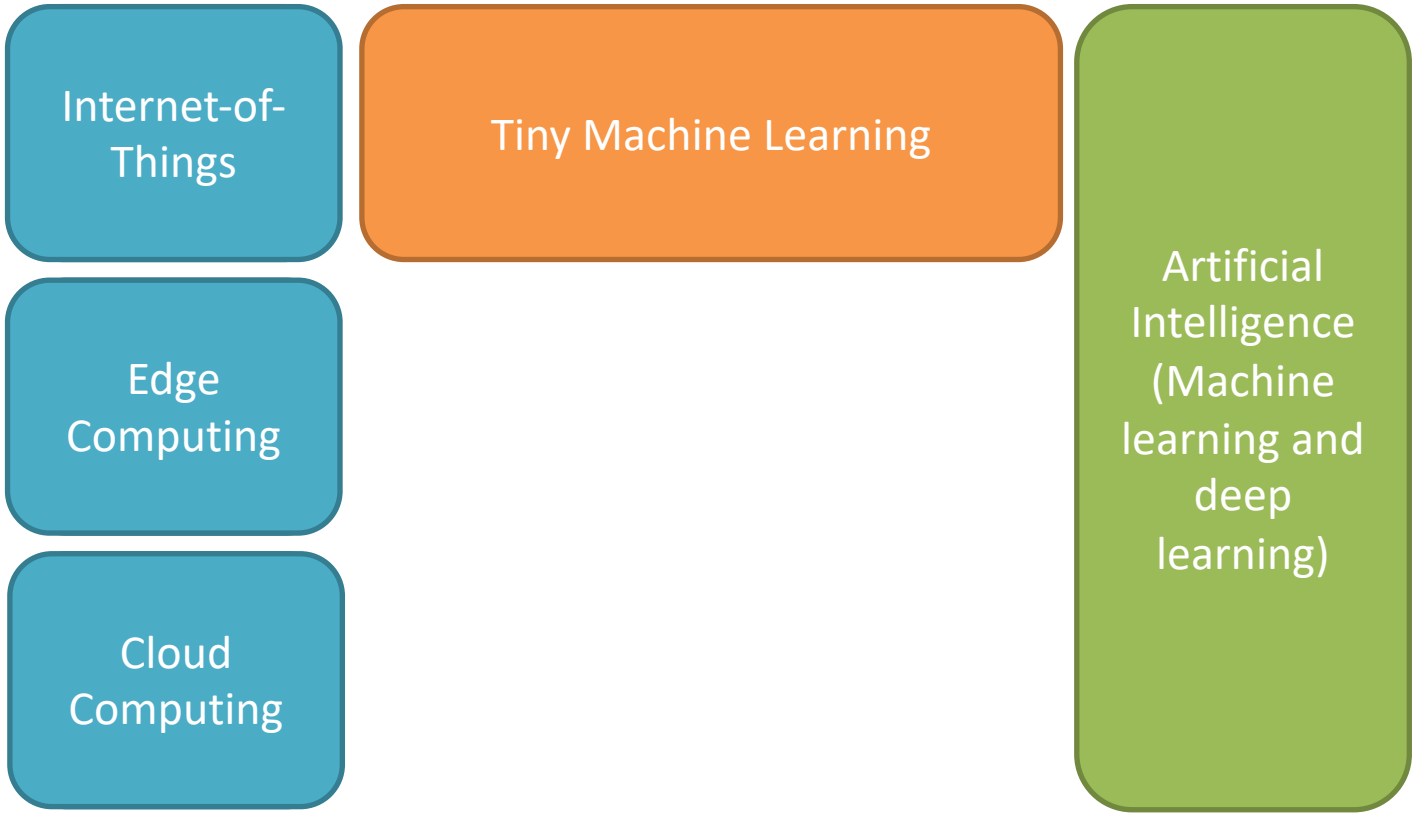
Internet-of-Things

Edge Computing

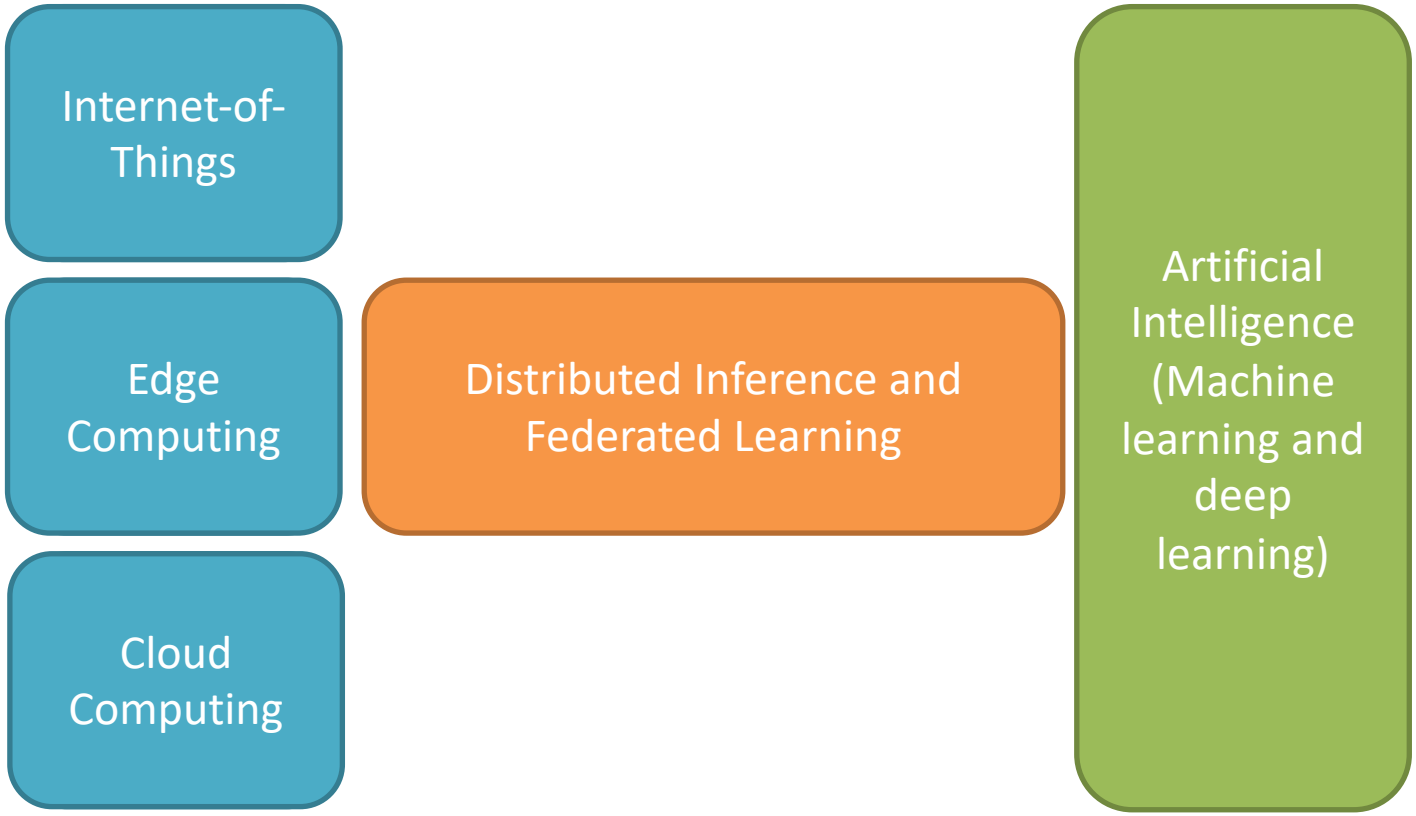
Cloud Computing

Artificial Intelligence (Machine learning and deep learning)

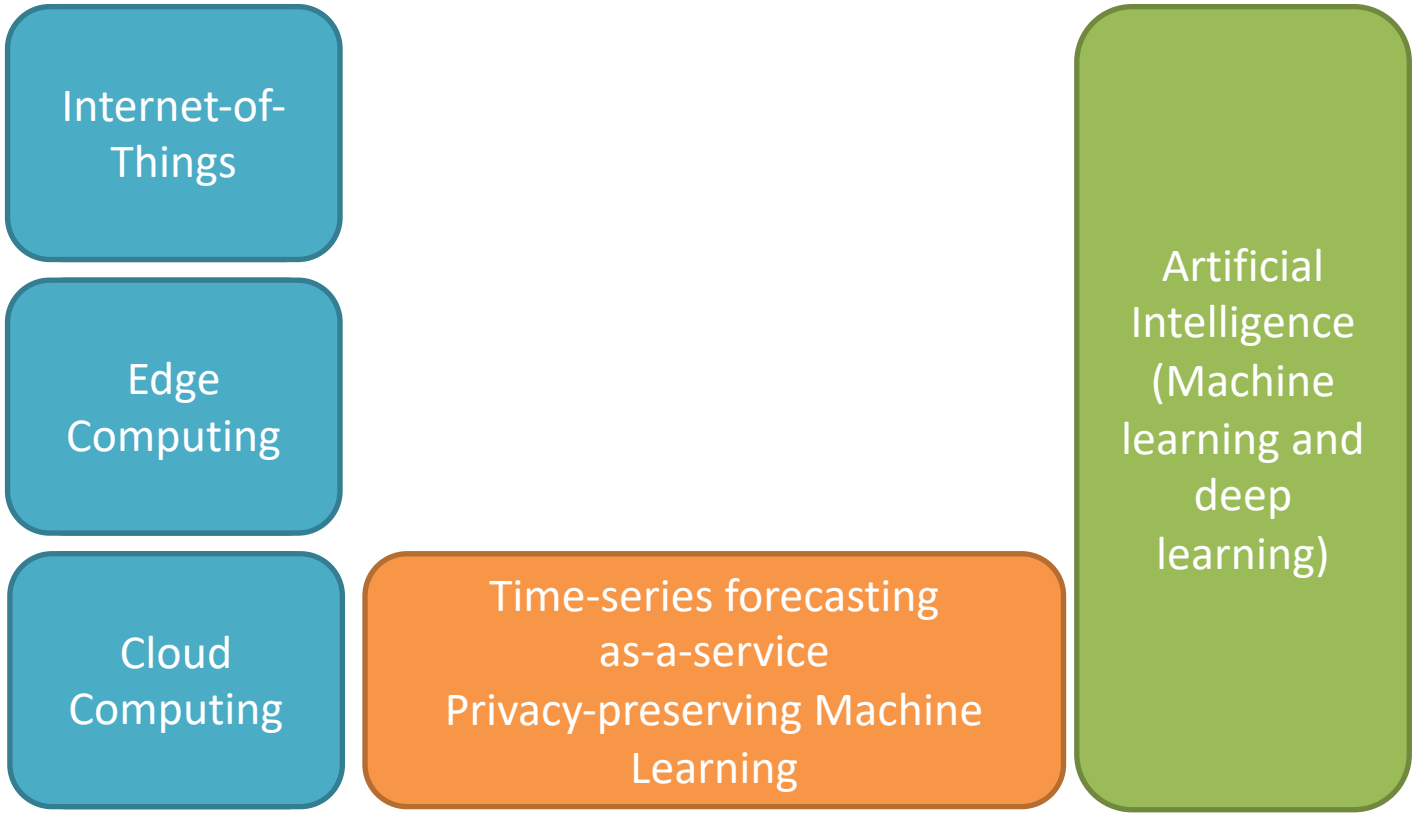
The research activity



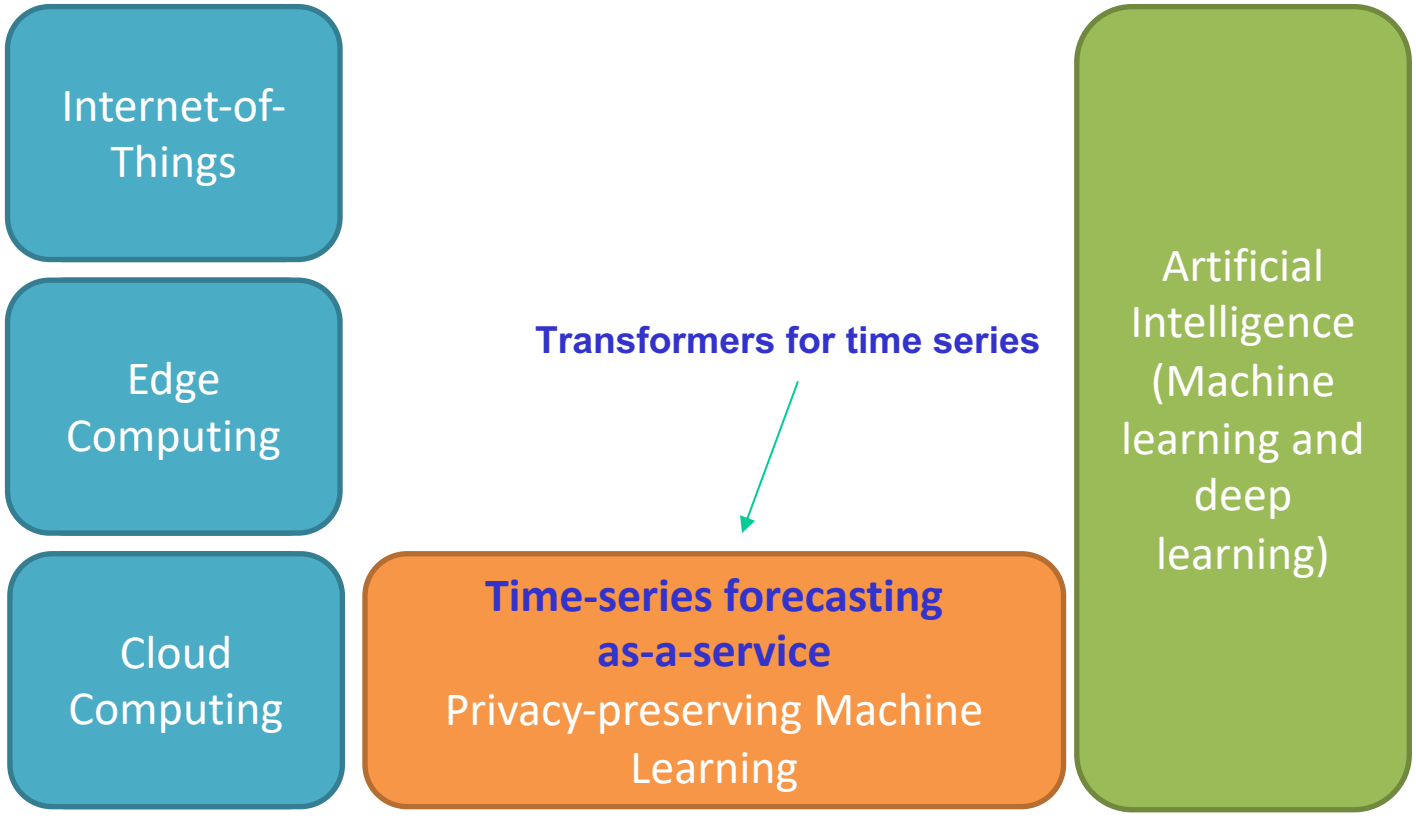
The research activity



The research activity



The research activity



POLITECNICO DI MILANO

ADVANCED TOPICS IN DEEP LEARNING: THE RISE OF TRANSFORMER

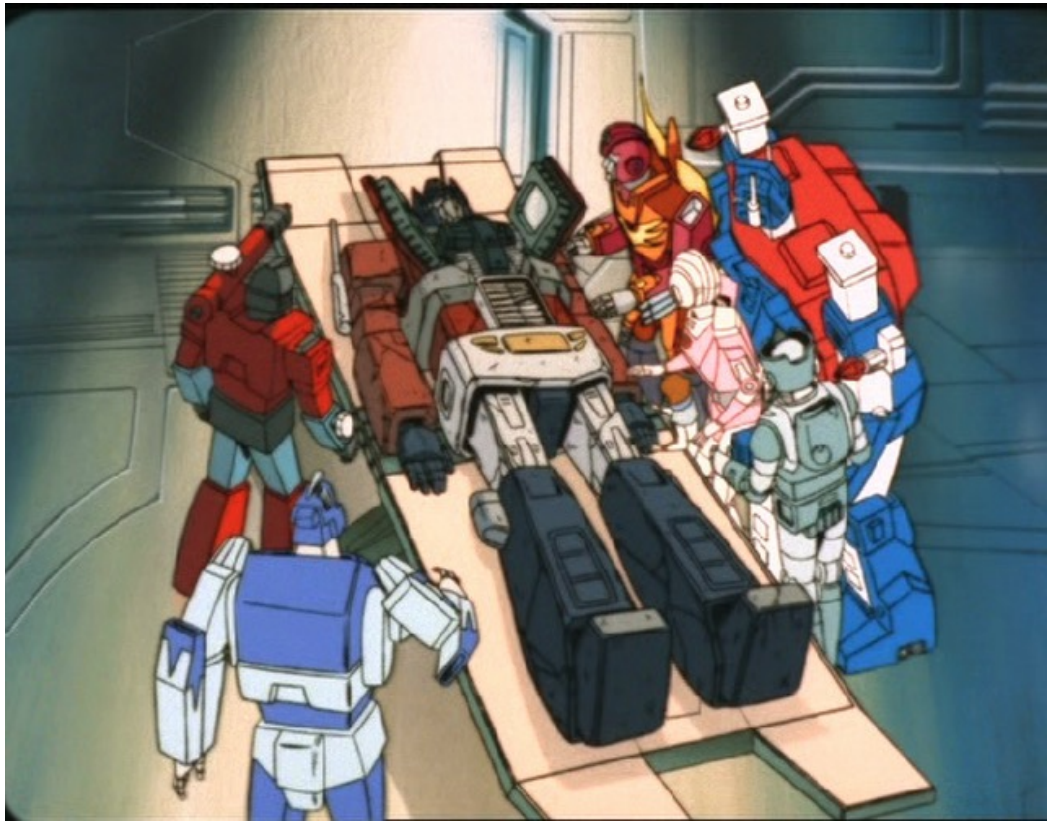


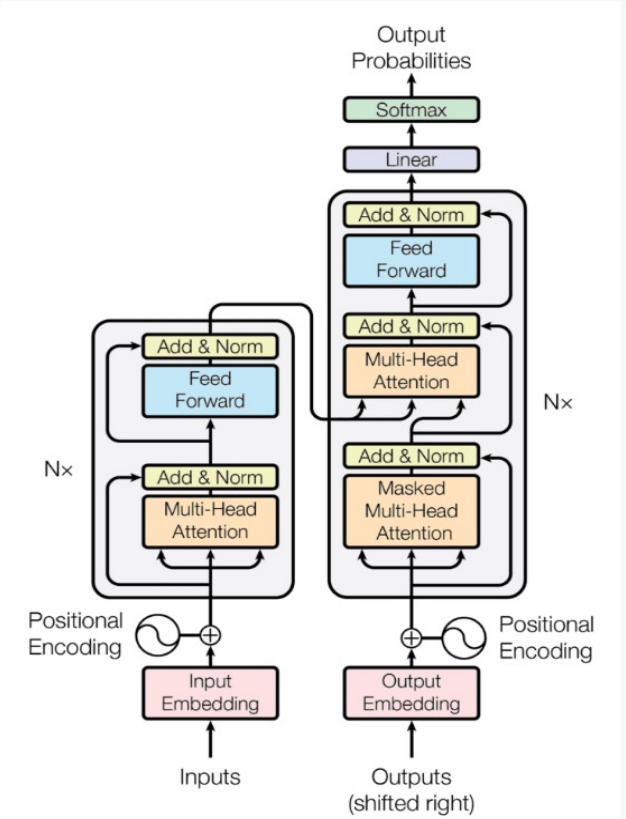


The rise (and the fall) of Transformers

Prof. Manuel Roveri, Ing. Alessandro Falcetta, Ing. Diego Riva

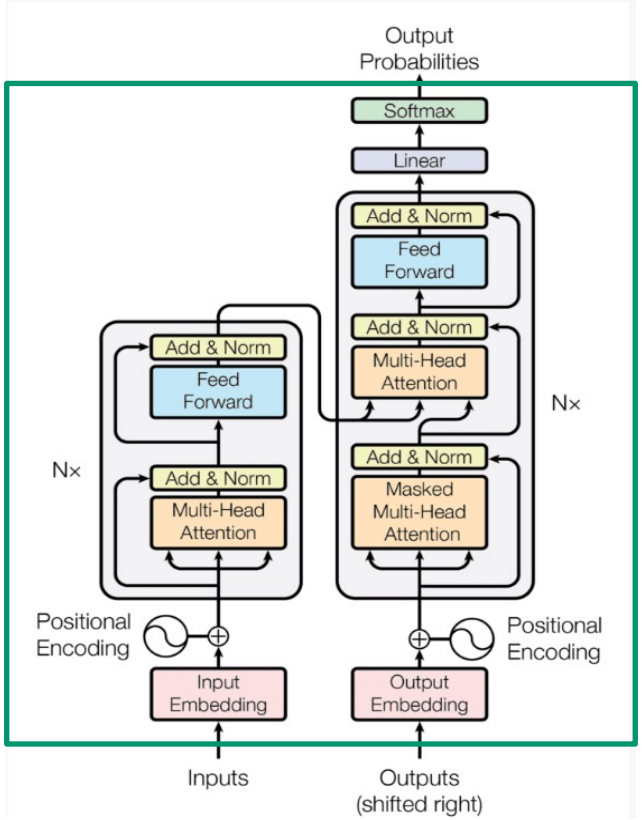
… the fall of Transformers





… the fall of Transformers. Why?

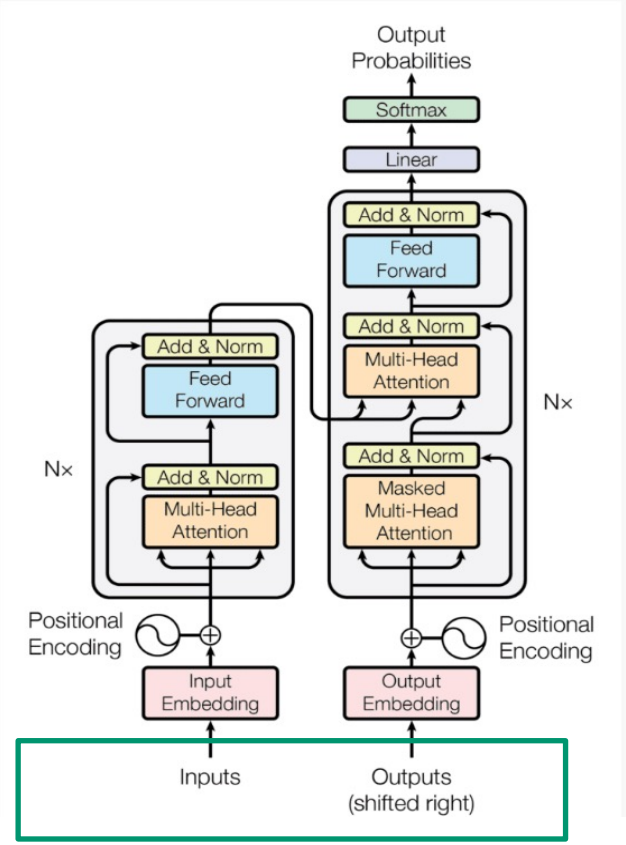
#1 Transformer is not suitable for time series prediction



… the fall of Transformers. Why?

#1 Transformer is not suitable for time series prediction

#2 Not just positional but temporal encoding is needed

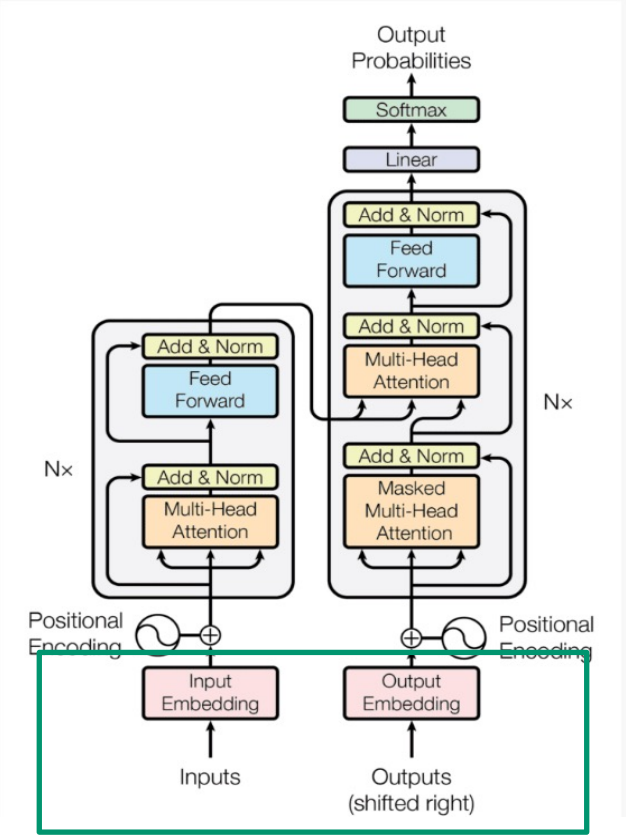


… the fall of Transformers. Why?

#1 Transformer is not suitable for time series prediction

#2 Not just positional but temporal encoding is needed

#3 Time series comprise real values



The (motivated) fall of Transformers

Methods	Metric	Electricity 96	192	336	720	Exchange-Rate 96	192	336	720	Traffic 96	192	336	720	Weather 96	192	336	720	ILI 24	36	48	60
DLinear-S*	MSE	0.194	0.193	0.206	0.242	**0.078**	0.159	0.274	**0.558**	0.650	**0.598**	**0.605**	0.645	0.196	0.237	0.283	0.345	**2.398**	**2.646**	2.614	2.804
	MAE	0.276	0.280	0.296	0.329	0.197	0.292	0.391	**0.574**	0.396	**0.370**	0.373	0.394	0.255	0.296	0.335	0.381	**1.040**	1.088	1.086	1.146
DLinear-I*	MSE	**0.184**	**0.184**	**0.197**	**0.234**	0.084	**0.157**	**0.236**	0.626	0.647	0.602	0.607	0.646	**0.164**	**0.209**	**0.263**	**0.338**	3.015	2.737	**2.577**	2.821
	MAE	**0.270**	**0.273**	**0.289**	**0.323**	0.216	0.298	**0.379**	0.634	0.403	0.375	0.377	0.398	**0.237**	**0.282**	**0.327**	**0.380**	1.192	**1.036**	1.043	**1.091**
FEDformer	MSE	0.193	0.201	0.214	0.246	0.148	0.271	0.460	1.195	**0.587**	0.604	0.621	**0.626**	0.217	0.276	0.339	0.403	3.228	2.679	2.622	2.857
	MAE	0.308	0.315	0.329	0.355	0.278	0.380	0.500	0.841	**0.366**	0.373	0.383	**0.382**	0.296	0.336	0.380	0.428	1.260	1.080	1.078	1.157
Autoformer	MSE	0.201	0.222	0.231	0.254	0.197	0.300	0.509	1.447	0.613	0.616	0.622	0.660	0.266	0.307	0.359	0.419	3.483	3.103	2.669	**2.770**
	MAE	0.317	0.334	0.338	0.361	0.323	0.369	0.524	0.941	0.388	0.382	**0.337**	0.408	0.336	0.367	0.395	0.428	1.287	1.148	1.085	1.125
Informer	MSE	0.274	0.296	0.300	0.373	0.847	1.204	1.672	2.478	0.719	0.696	0.777	0.864	0.300	0.598	0.578	1.059	5.764	4.755	4.763	5.264
	MAE	0.368	0.386	0.394	0.439	0.752	0.895	1.036	1.310	0.391	0.379	0.420	0.472	0.384	0.544	0.523	0.741	1.677	1.467	1.469	1.564
Pyraformer*	MSE	0.386	0.378	0.376	0.376	1.748	1.874	1.943	2.085	0.867	0.869	0.881	0.896	0.622	0.739	1.004	1.420	7.394	7.551	7.662	7.931
	MAE	0.449	0.443	0.443	0.445	1.105	1.151	1.172	1.206	0.468	0.467	0.469	0.473	0.556	0.624	0.753	0.934	2.012	2.031	2.057	2.100
LogTrans	MSE	0.258	0.266	0.280	0.283	0.968	1.040	1.659	1.941	0.684	0.685	0.734	0.717	0.458	0.658	0.797	0.869	4.480	4.799	4.800	5.278
	MAE	0.357	0.368	0.380	0.376	0.812	0.851	1.081	1.127	0.384	0.390	0.408	0.396	0.490	0.589	0.652	0.675	1.444	1.467	1.468	1.560
Reformer	MSE	0.312	0.348	0.350	0.340	1.065	1.188	1.357	1.510	0.732	0.733	0.742	0.755	0.689	0.752	0.639	1.130	4.400	4.783	4.832	4.882
	MAE	0.402	0.433	0.433	0.420	0.829	0.906	0.976	1.016	0.423	0.420	0.420	0.423	0.596	0.638	0.596	0.792	1.382	1.448	1.465	1.483
Repeat-C*	MSE	1.588	1.595	1.617	1.647	0.081	0.167	0.305	0.823	2.723	2.756	2.791	2.811	0.259	0.309	0.377	0.465	6.587	7.130	6.575	5.893
	MAE	0.946	0.950	0.961	0.975	**0.196**	**0.289**	0.396	0.681	1.079	1.087	1.095	1.097	0.254	0.292	0.338	0.394	1.701	1.884	1.798	1.677

- Methods* are implemented by us; Other results are from FEDformer [29].

Linear

Transformers

Zeng, A., Chen, M., Zhang, L., & Xu, Q. (2022). Are Transformers Effective for Time Series Forecasting?. *arXiv preprint arXiv:2205.13504*.

What is a time-series?

(aka are time series data just related to time?)

The importance of time series analysis

Smart Cities

E-health

Smart Car

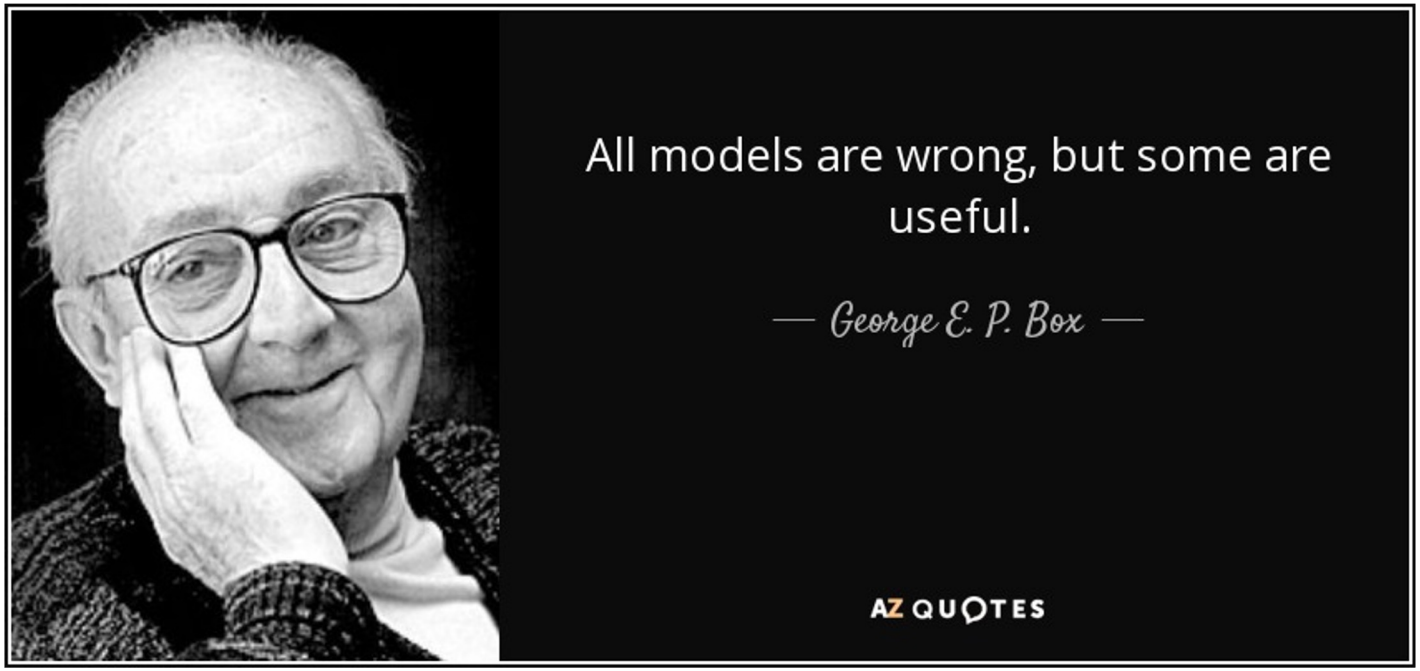
Massive production of time-series data over time

Smart Home/Building

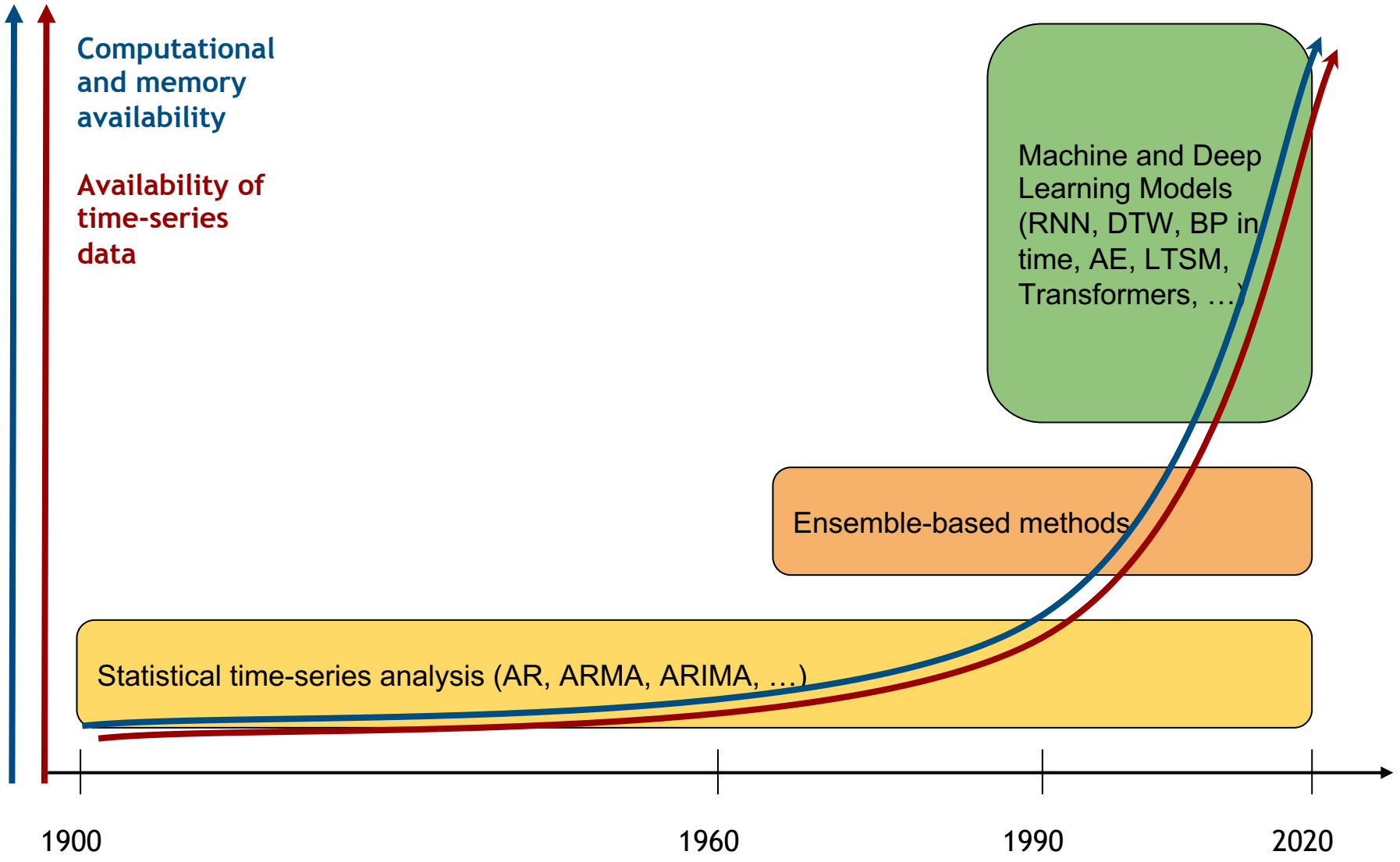
Industry 4.0

Time series analysis aims at extracting meaningful summary and statistical information from points arranged in chronological order

So, are we interested in forecasting the future ...



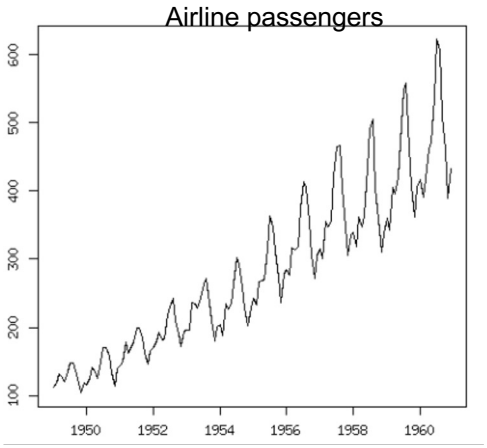
The rise of time series analysis and forecasting



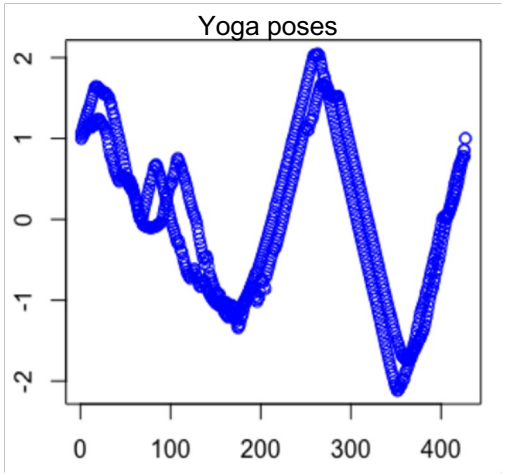
The definition of time series

A time series is a sequence of data points indexed in chronological order: $X = (x_1, x_2, \ldots, x_N, \ldots)$

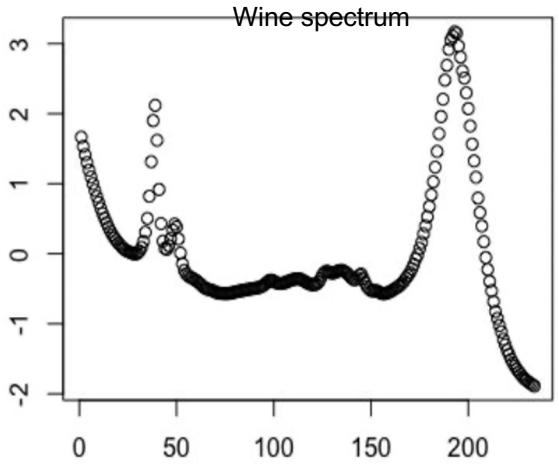
In other words, a time series is a series of data points taken at successive equally spaced points ~~in time~~.



Time

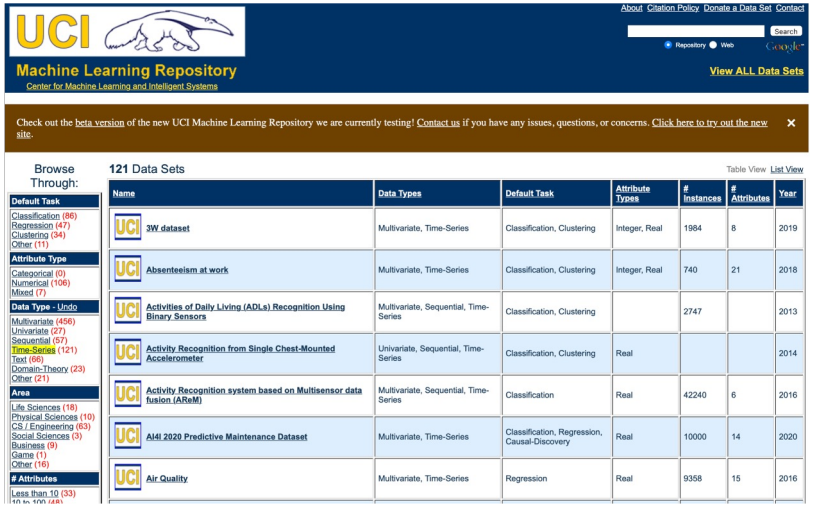


A person performed a series of transitions between yoga poses while images were recorded. The images were converted to a one-dimensional series.

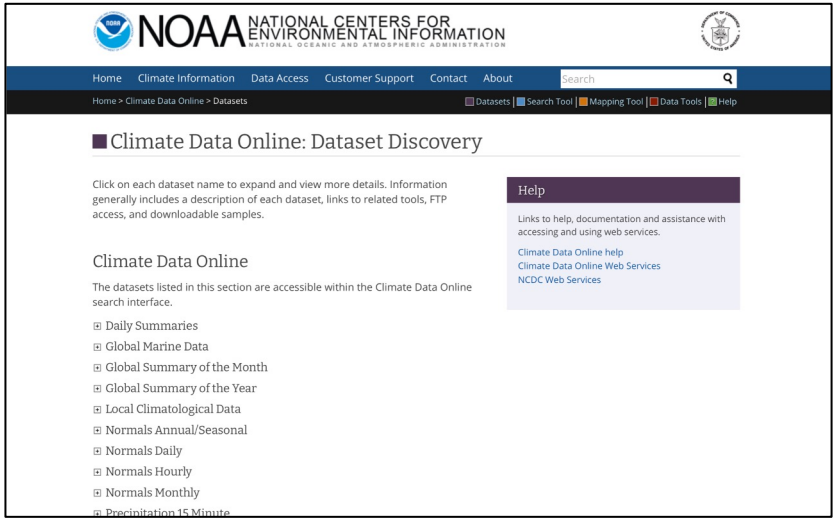


A *spectrum* is a plot of light wavelength versus intensity

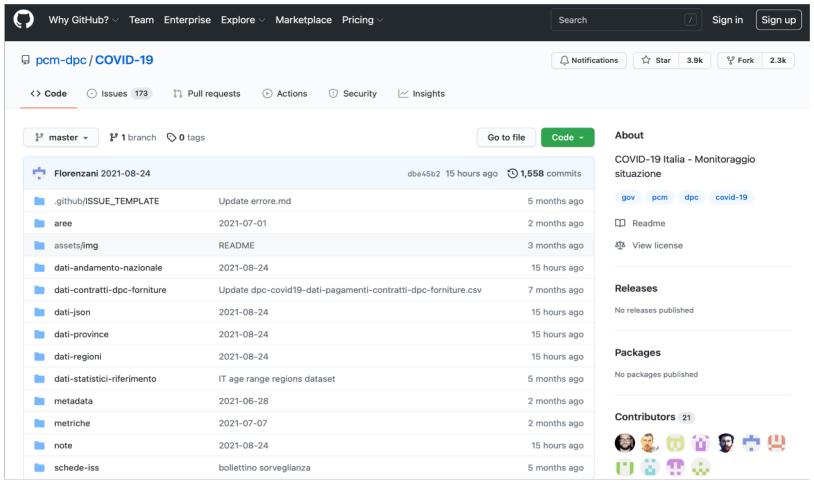
Where can we find time series data?



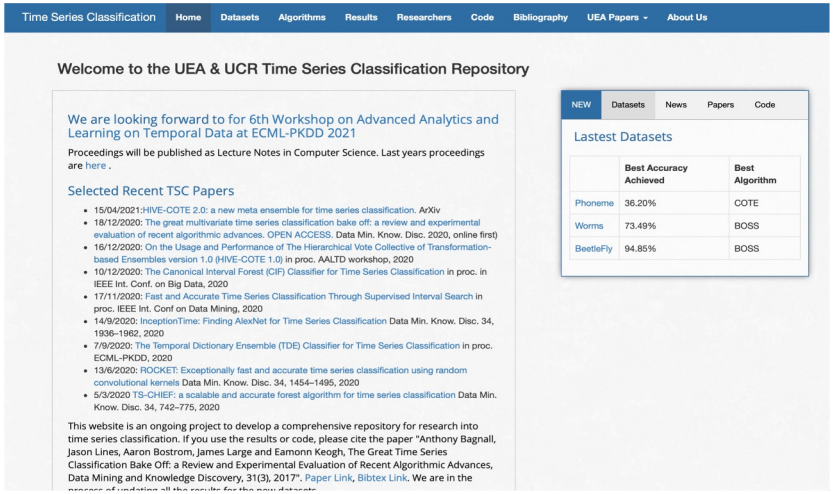
http://archive.ics.uci.edu/ml/index.php



https://www.ncdc.noaa.gov/cdo-web/datasets



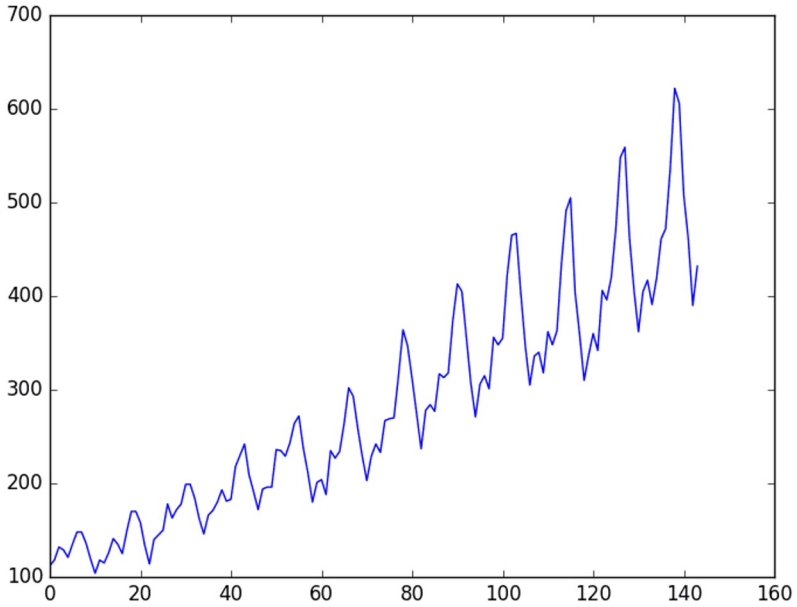
https://github.com/pcm-dpc/COVID-19



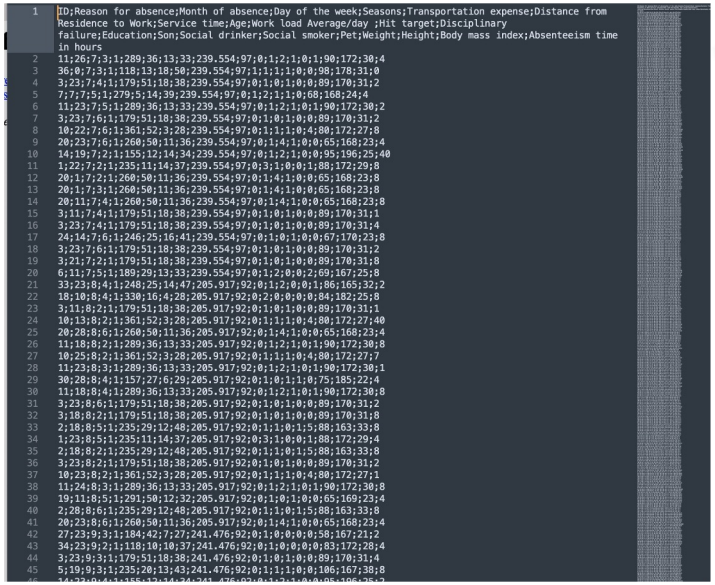
http://www.timeseriesclassification.com

Working with time series …

What we expect …



… what we get!



- Irregular or missing data
- No time stamp
- No explicit time series
- …

Wrangling time-series data (aka garbage in, garbage out)

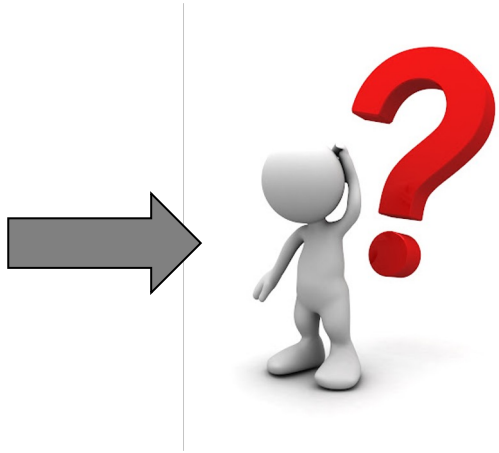
What to do with that?

- Timestamping Troubles
- Handling missing data
- Changing the frequency of the time series
- Smoothing data
- Addressing seasonality in data

1) Timestamping Troubles

- Timestamps are quite helpful for time series analysis.
- From timestamps, we can extrapolate a number of interesting features, such as time of day or day of the week.
- But.. what process generated the timestamp, how, and when?
  - Often an event happening is not coincident with an event being recorded
  - E.g., the sample meal diary from a weight loss app

Time	Intake
Mon, April 7, 11:14:32	pancakes
Mon, April 7, 11:14:32	sandwich
Mon, April 7, 11:14:32	pizza



Did the user specify this time or was it automatically created? Does the interface perhaps offer an automatic time that the user can adjust or choose to ignore? Where in the world was it 11:14?

2) Handling missing data

- Missing data is surprisingly common in real-world datasets (e.g., communication problems, faults in sensor/actuators, software bugs, etc..)
- How to deal with that?
  - **“Global” filling methods**
    - When we fill in missing data based on observations about the entire data set.
  - **“Local” filling methods**
    - When we use neighboring data points to estimate the missing value (Forward Fill, Moving average, local Interpolation)
  - **Deletion of affected time periods**
    - When we choose not to use time periods that have missing data at all.

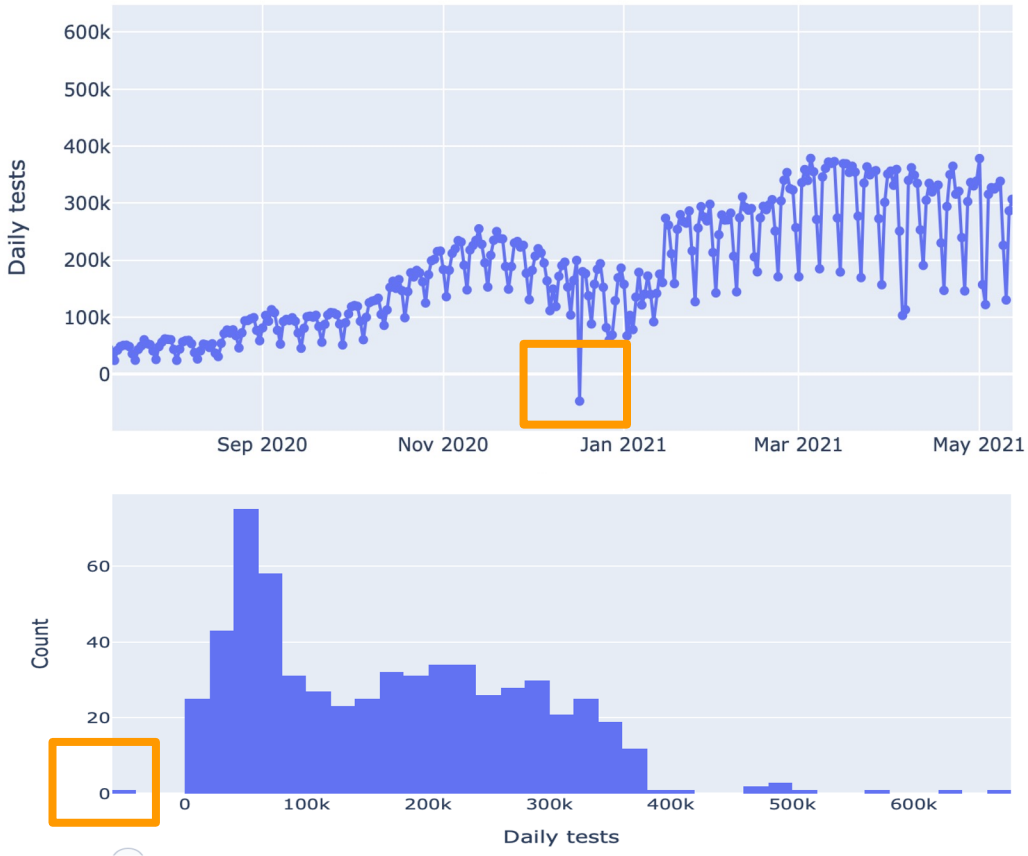
3) Changing the frequency of the time series: upsampling and downsampling

- Dealing with **time series having different sampling frequencies**
- **Change the sampling frequency** of your data:
  - We **cannot change the actual rate** at which information was measured
  - We **can change the frequency of the timestamps** in your data collection
- **Upsampling**: increasing the timestamp frequency (generating)
- **Downsampling**: decreasing the timestamp frequency (subsampling)

4) Smoothing data

Why are we smoothing data?

- Data should be smoothed to eliminate measurement spikes or errors of measurement, e.g., exponential smoothing



- Data preparation
- Feature generation
- Prediction
- Visualization

5) Addressing seasonality in data

Seasonality in data is any kind of recurring behavior in which the frequency of the behavior is stable

It can occur at many different frequencies at the same time.

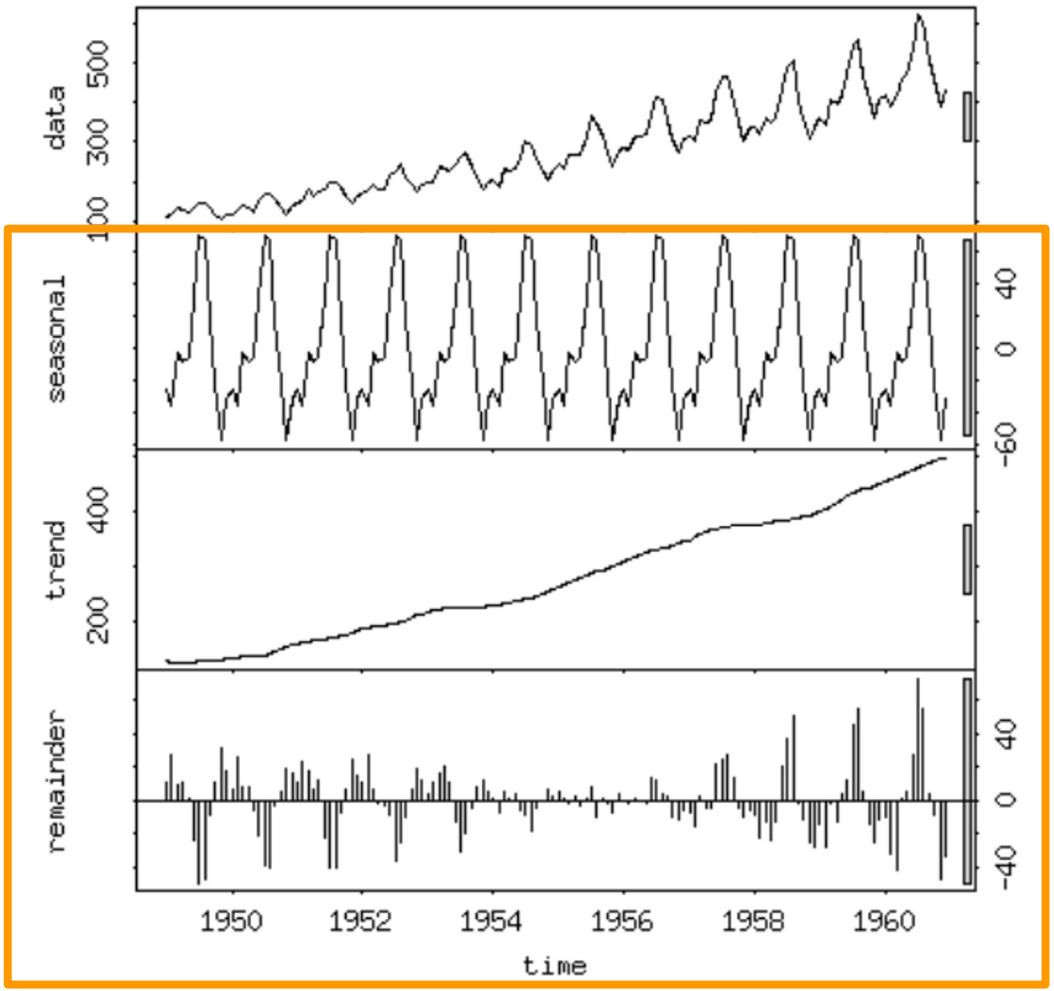
For example, human behavior tends to have:

- a daily seasonality (lunch at the same time every day),
- a weekly seasonality (Mondays are similar to other Mondays),
- a yearly seasonality (New Year's Day has low traffic)

Physical systems also demonstrate seasonality, such as the period the Earth takes to revolve around the sun

Identifying and dealing with seasonality is part of the modeling process

Time series decomposition: seasonal, trend, reminder



1. **The seasonal component** is found by LOESS smoothing ("locally estimated scatter plot smoothing") the seasonal subseries (the series of all January values, ...).
2. The seasonal values are removed, the remainder smoothed to **find the trend**.
3. This process is iterated a few times.
4. **The remainder component is the residuals f**rom the seasonal plus trend fit.

Taken from "Practical Time Series Analysis", Aileen Nielsen, O'Reilly 2019.

Exploratory Data Analysis for Time Series

Data methods for time series

(histograms, plotting, and group-by operations applied to time series data)

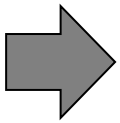
Temporal methods for time series analysis

(nonstationary, self-correlation, spurious correlation)

Two main approaches

Data methods for time series

(histograms, plotting, and group-by operations applied to time series data)



Let's start with these methods …

Data methods for time series

How to apply commonly used data exploration techniques to time series data sets?

- Are any of the columns strongly correlated with one another?
- What is the overall mean of an interesting variable? What is its variance?

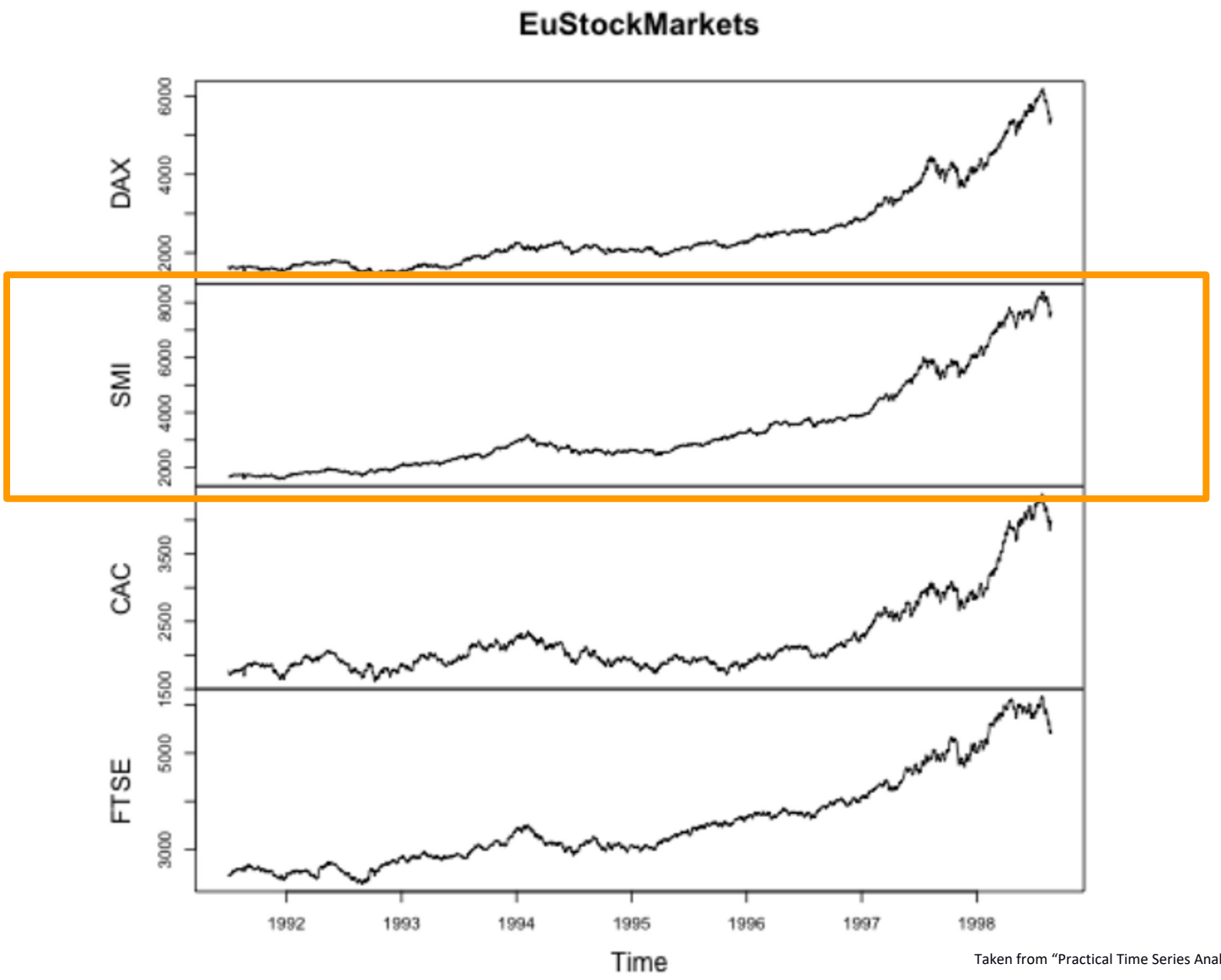
Plotting, computing summary statistics, applying histograms, and using targeted scatter plots

Explicitly time-oriented questions:

- What is the range of values you see, and do they vary by time period or some other logical unit of analysis?
- Does the data look consistent and uniformly measured, or does it suggest changes in either measurement or behavior over time?

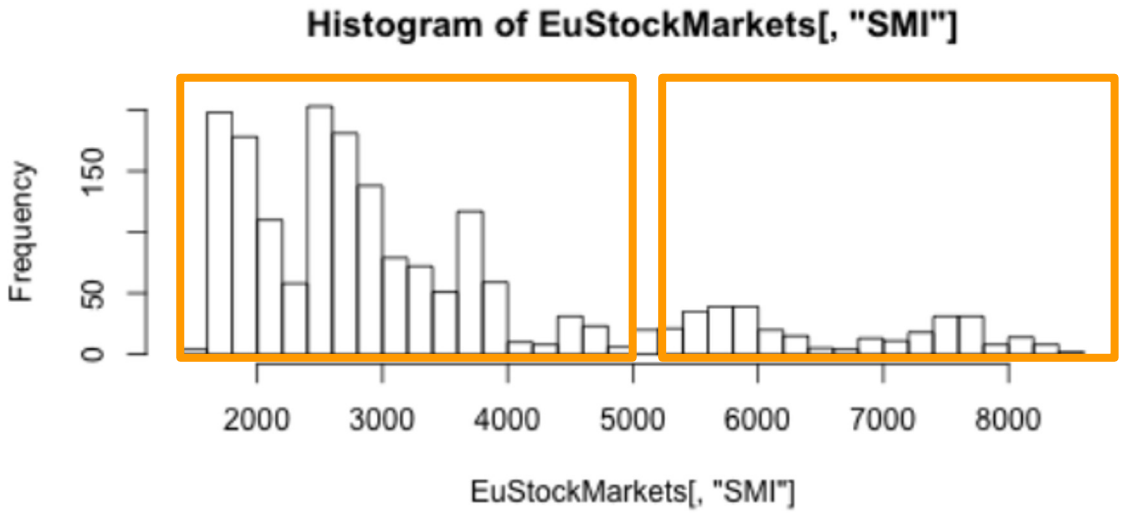
Incorporate time into our statistics, as an axis in our graphs or as a group in group-by operations

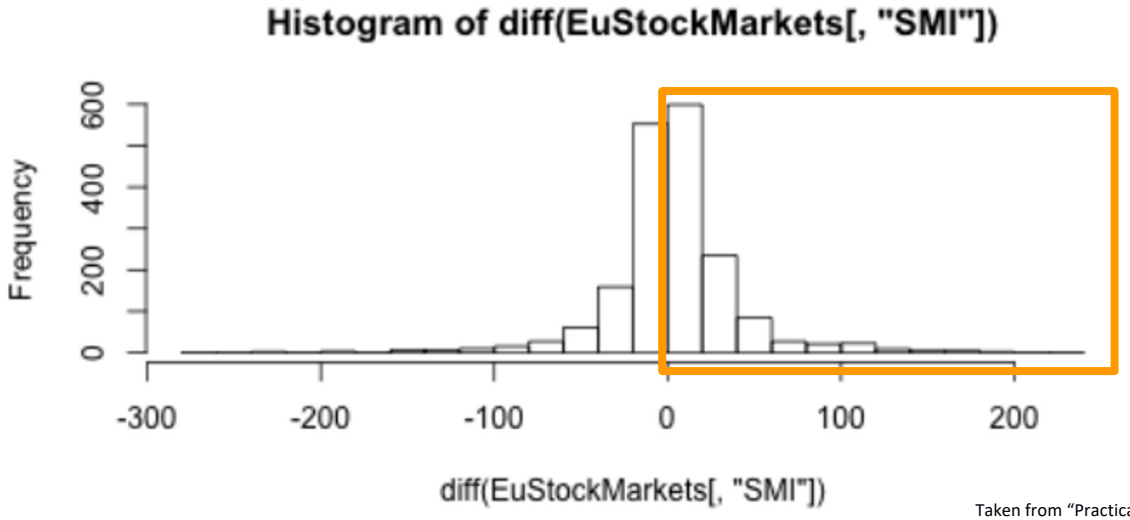
Plotting



Taken from "Practical Time Series Analysis", Aileen Nielsen, O'Reilly 2019.

Histograms “of the data” and histograms “of the difference”

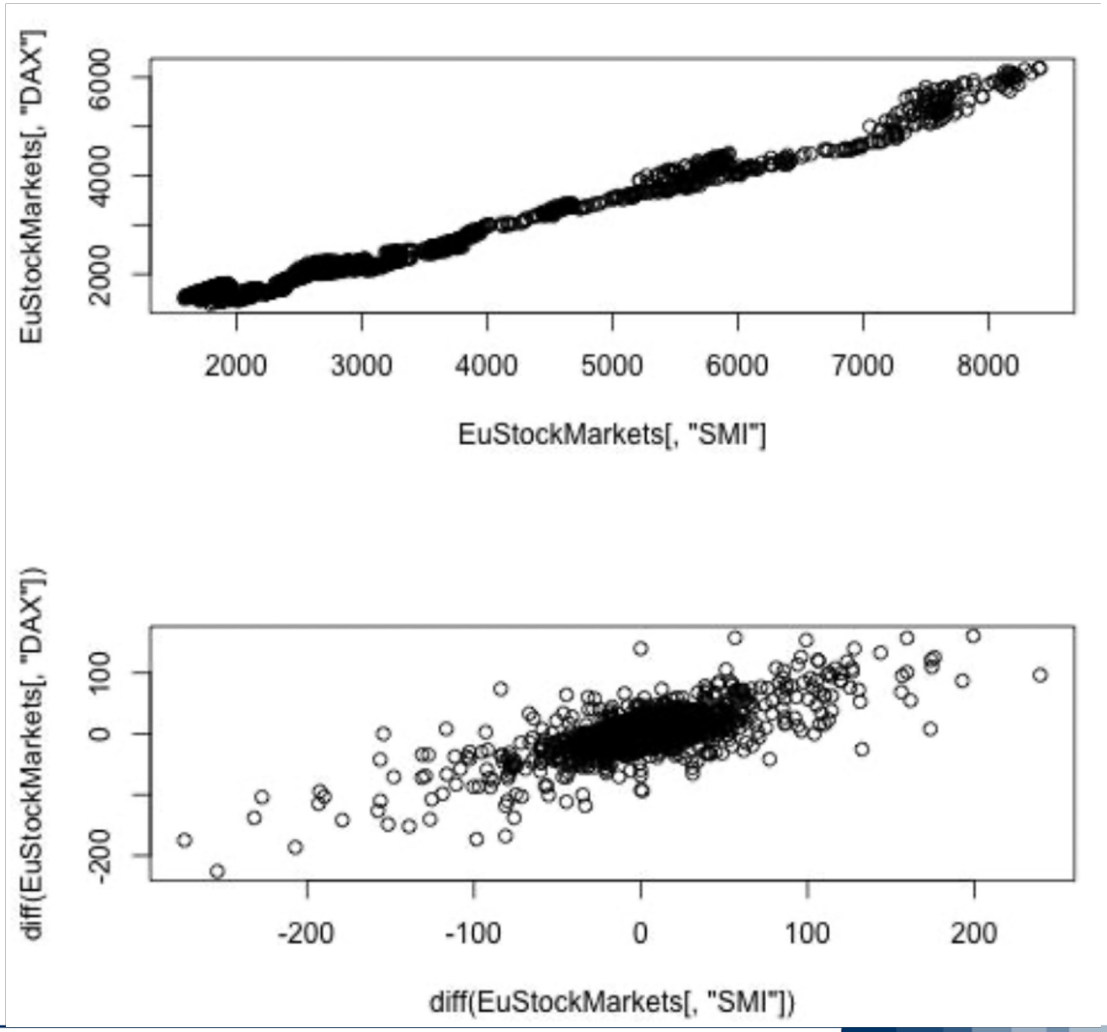




Taken from “Practical Time Series Analysis”, Aileen Nielsen, O’Reilly 2019.

Scatter Plots

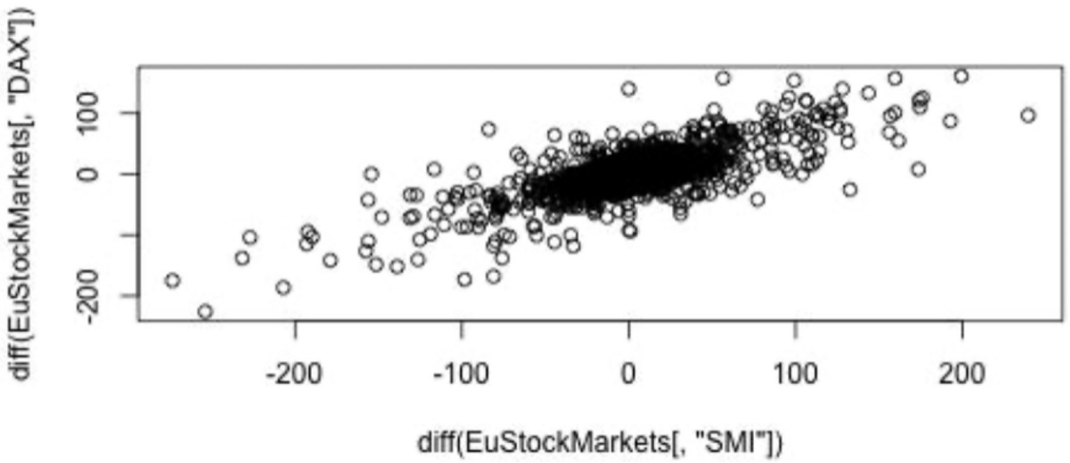
Evaluate how two stocks are correlated at a specific time:



These “apparent” correlations are interesting. Even if they are true correlations these are not correlations we can monetize as stock traders.

Taken from “Practical Time Series Analysis”, Aileen Nielsen, O’Reilly 2019.

Why not making money with scatter plot ???

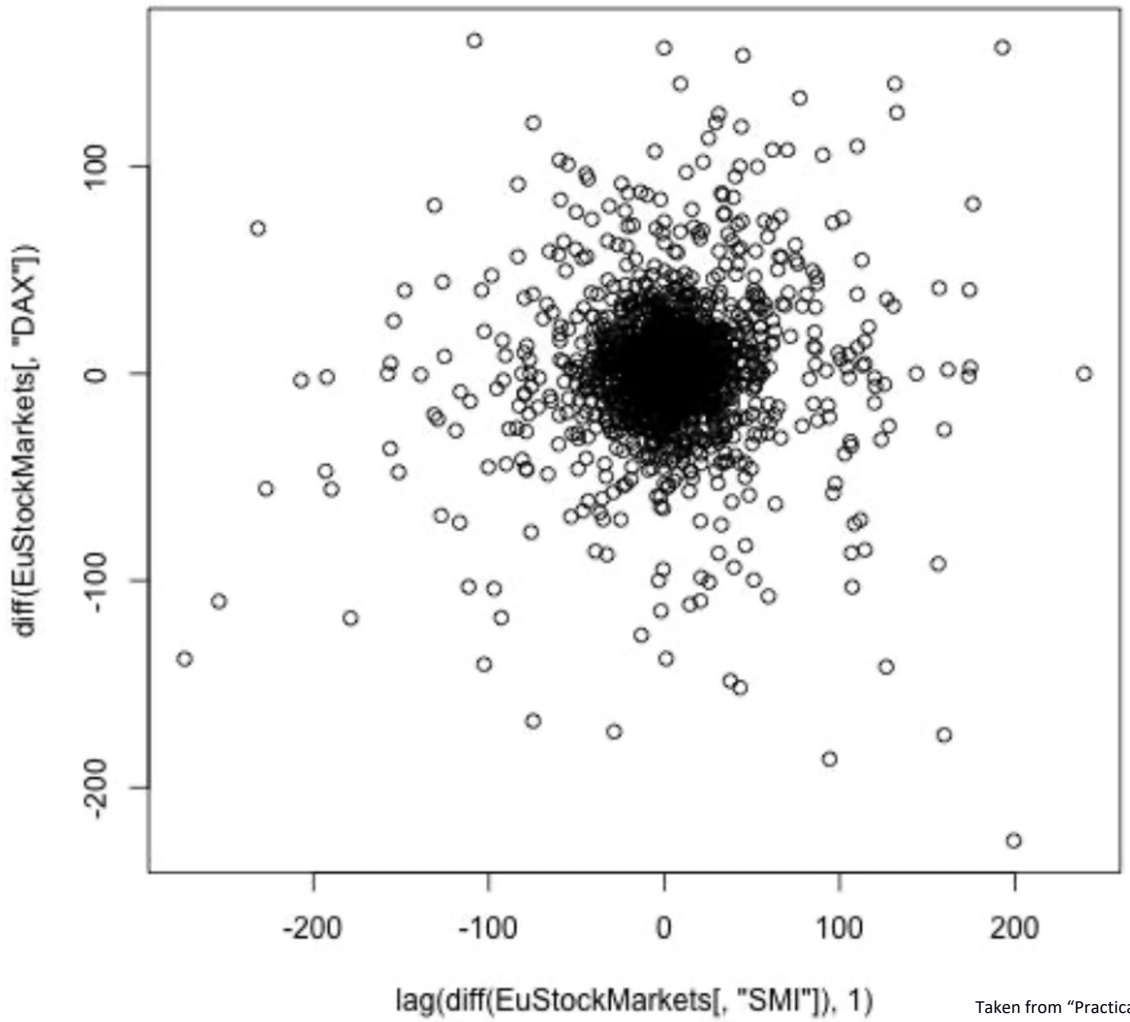


- By the time we know whether a stock is going up or down, the stocks it is correlated with will have also gone up or down, since we are taking correlations of values at **identical time points**.
- What we need to do is find out whether the change in one stock earlier in time can **predict the change in another stock later in time**.
- To do this, we shift one of the **differences of the stocks back by 1 before looking at the scatter plot**.

Taken from "Practical Time Series Analysis", Aileen Nielsen, O'Reilly 2019.

Scatter plot of “diff” and “diff + 1”

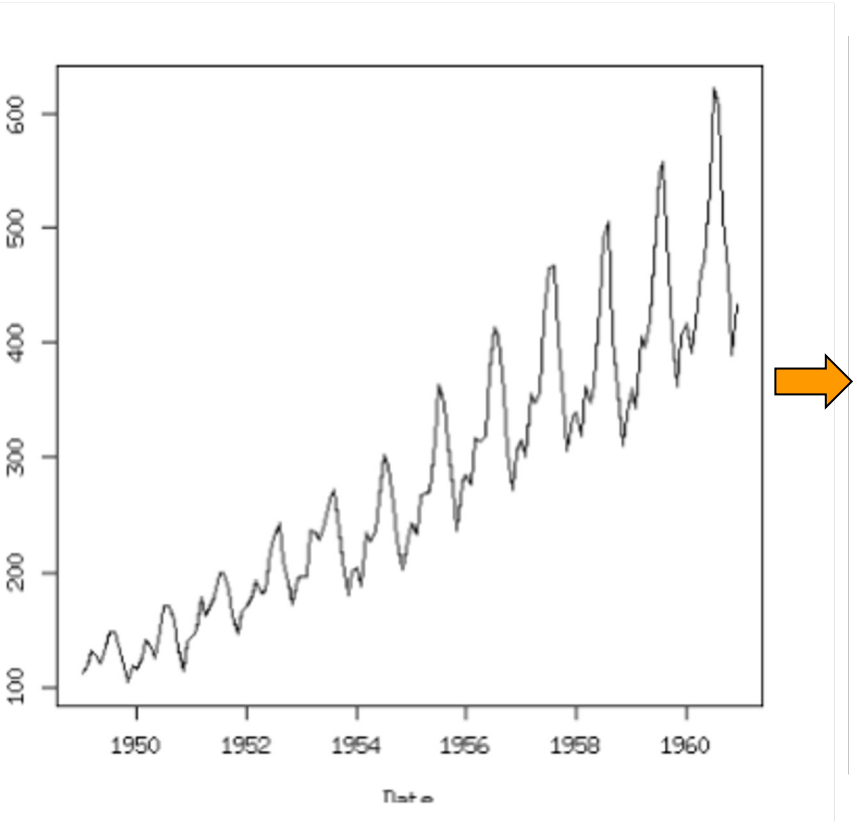
Evaluate how stock price are related at different time instants

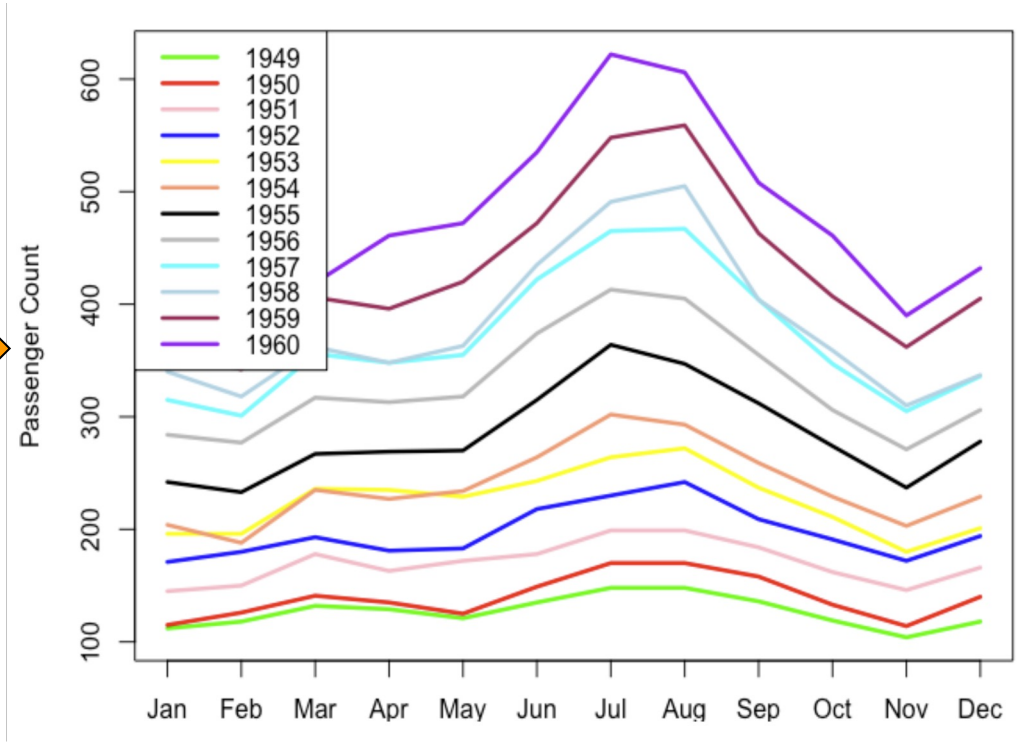


Extremely difficult for a stock trader!!!

Taken from “Practical Time Series Analysis”, Aileen Nielsen, O’Reilly 2019.

Group-by operations on time series (#1)

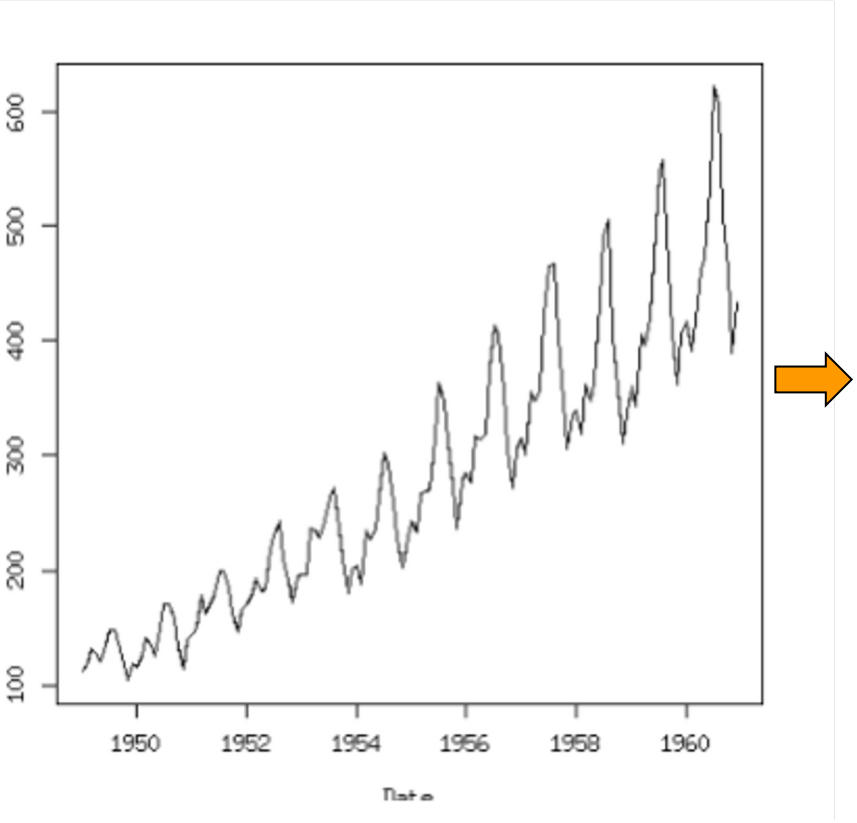


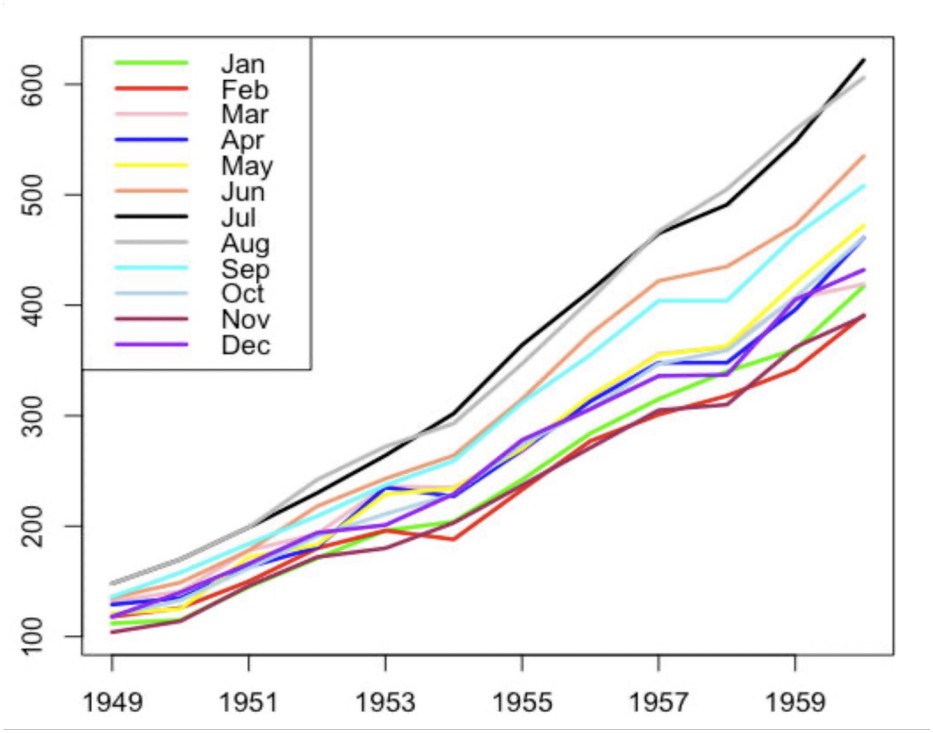


*Per-year **month-by-month** counts of airline passenger (here I fix the year and I plot the values of the different months per year)*

Taken from "Practical Time Series Analysis", Aileen Nielsen, O'Reilly 2019.

Group-by operations on time series (#2)



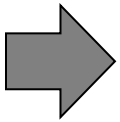


*Per-month curves of **year-to-year** time serie (here I fix the month and I plot the value of the month for the different years)*

Taken from "Practical Time Series Analysis", Aileen Nielsen, O'Reilly 2019.

Two main approaches

… and now let's have a look at these methods …



Temporal methods for time series analysis

Time Series-Specific Exploratory Methods

Stationarity:

- What is a stationary time series and how can we measure it?

Self-correlation

- How to evaluate and determine a time series is correlated with itself? How can it be useful for understanding the underlying dynamics?

Spurious correlations

- What is a spurious correlation and how can we deal with that?

What is the “stationary” for a time series?

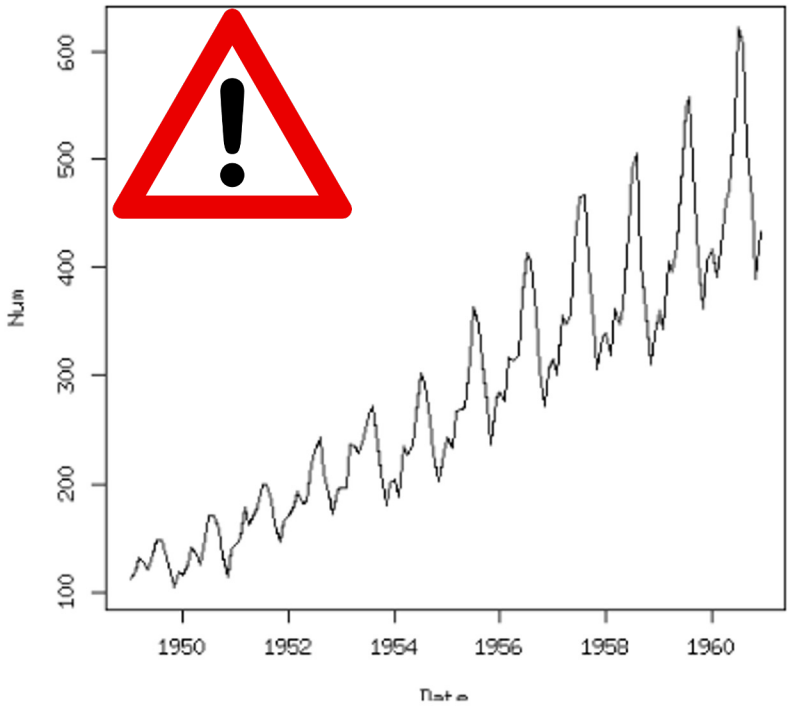
- The stationary **measures the “stability” of a time series**: how the (possibly long) past of a system would reflect for the (possibly long) future!
- Given that the time series is stationary we **can compute relevant figures of merit**, such as:
  - Seasonal changes
  - Self-correlations
  - Etc..

(Generally speaking) a time series is stationary when its statistical properties (e.g., mean and variance) remain stable over time

Understanding stationary (more formally)

A **time series is stationary when system generating the measurement is in a "steady state"**: i.e., for all possible lags, k, the distribution of y_t, y_{t+1},..., y_{t+k}, does not depend on t.

Sometimes it's easier to understand what "it is not stationary"..



- The mean value increases over time
- The variance increases over time

Why stationary is relevant?

- Many time series models assume the stationary of the data generating the process (e.g., AR, ARMA, etc..)
- A nonstationary time series could vary its accuracy over time:
  - the “mean” testing accuracy of our prediction model we can estimate in validation could be far from what we will have in the future
- Models could become obsolete over time… dealing with concept drift!
  - Adaptive mechanisms are required…

Time Series-Specific Exploratory Methods

Stationarity:

- What is a stationary time series and how can we measure it?

Self-correlation

- How to evaluate and determine a time series is correlated with itself? How can it be useful for understanding the underlying dynamics?

Spurious correlations

- What is a spurious correlation and how can we deal with that?

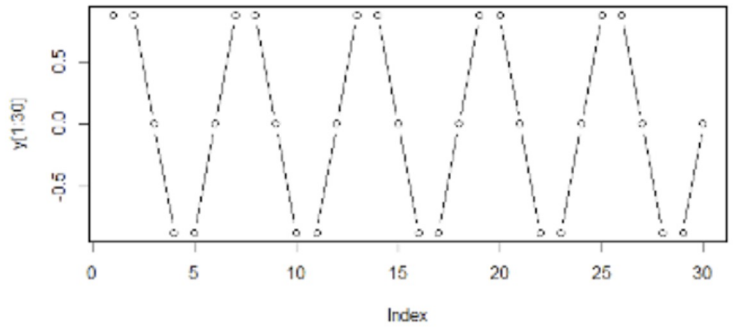
The importance of Self-Correlation

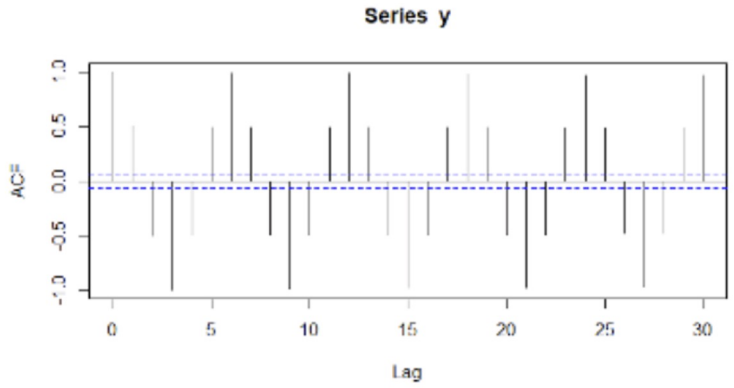
- Self-correlation in time series is crucial to **understand how a value** of the time series **at a given time instant is correlated to another value** of the time series **at a different time instant**
- Self-correlation is able to **characterize the dynamics of a time series over time**
- Human behaviours and environmental conditions tend to be self-correlated over time (e.g., the daily behaviour of people and the yearly weather conditions)
- **Auto-correlation extends self-correlations by considering all the possible time-delays between time points**

The autocorrelation function (ACF)

This is the Wikipedia's definition:

> Autocorrelation, also known as serial correlation, is the correlation of a signal with a delayed copy of itself as a function of the delay. Informally, it is the similarity between observations as a function of the time lag between them.

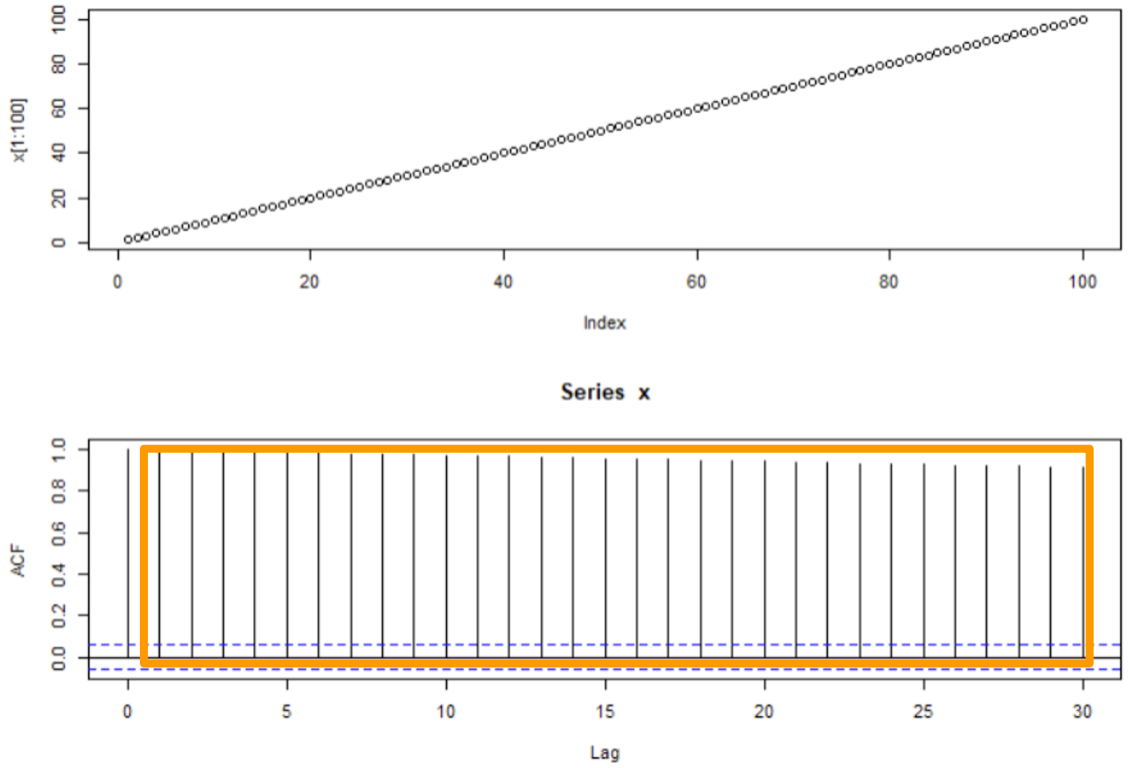




- From the ACF, we see that points that have a lag between them of 0 have a correlation of 1 (this is true for every time series),
- Points separated by 1 lag have a correlation of 0.5.
- Points separated by 2 lags have a correlation of –0.5, and so on.

Taken from "Practical Time Series Analysis", Aileen Nielsen, O'Reilly 2019.

What's the problem with ACF?

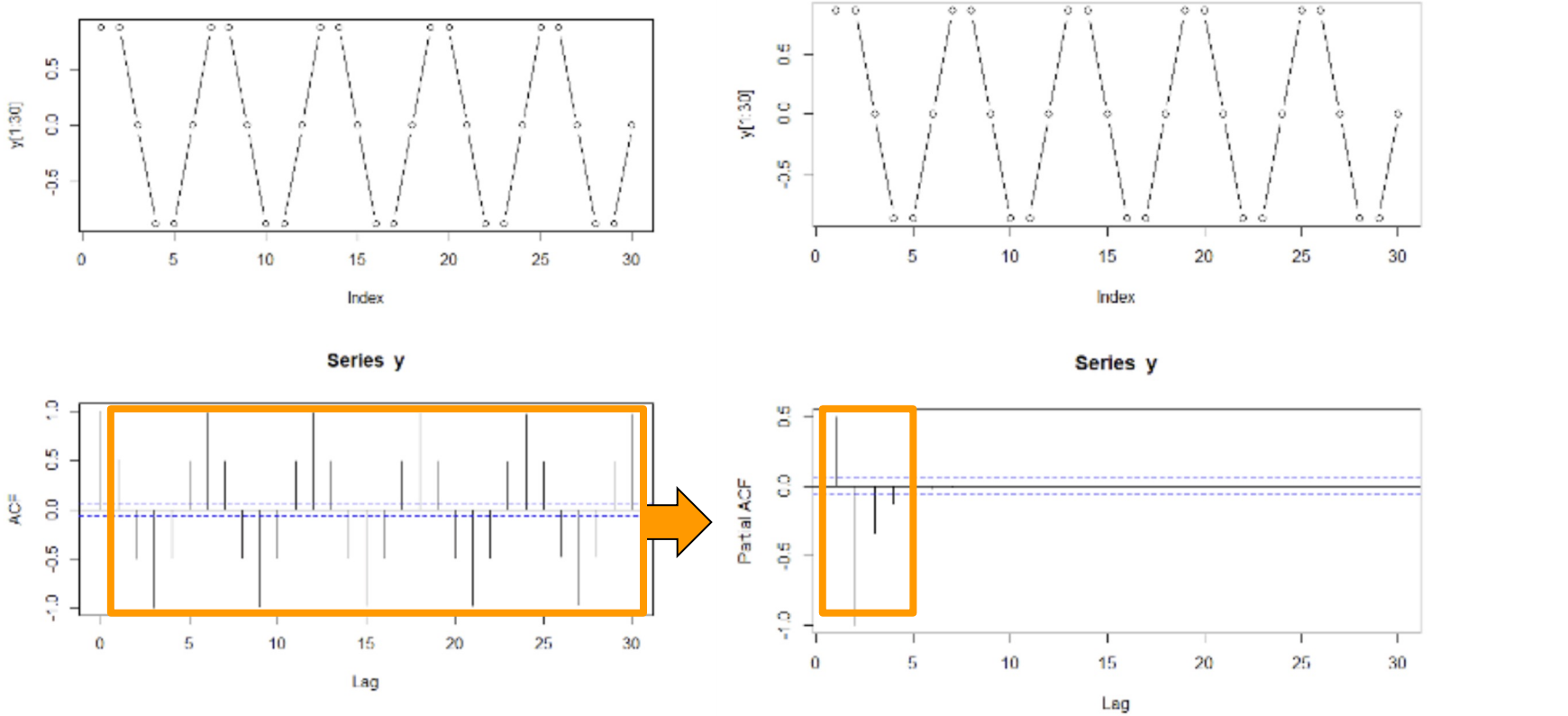


The ACF is not informative. It has a similar value for every lag, seemingly implying that all lags are equally correlated to the data

Taken from "Practical Time Series Analysis", Aileen Nielsen, O'Reilly 2019.

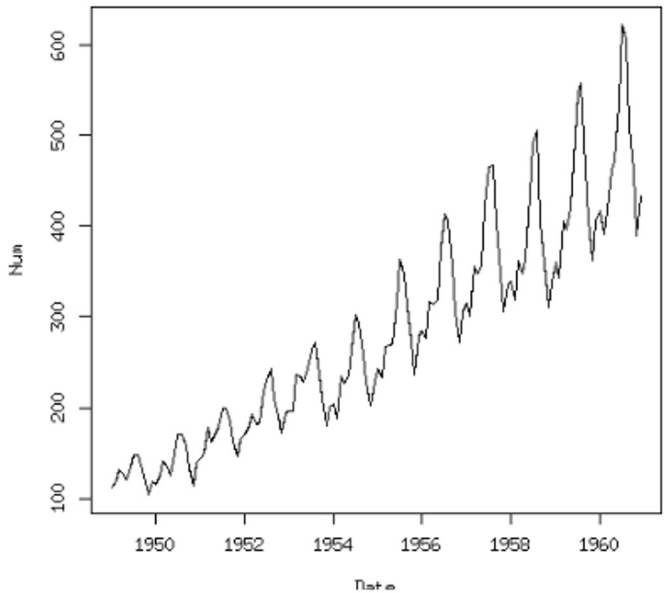
The partial autocorrelation function (PACF)

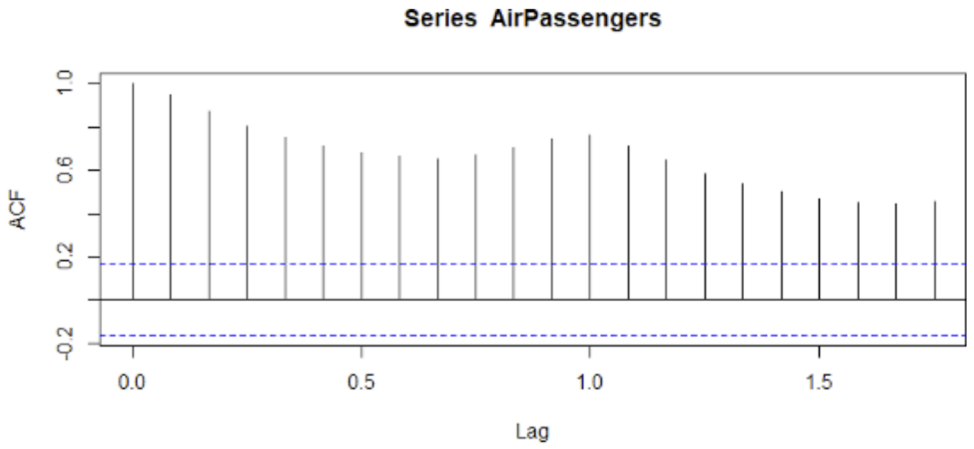
The partial autocorrelation of a time series for a given lag is the partial correlation of the time series with itself at that lag given all the information between the two points in time.

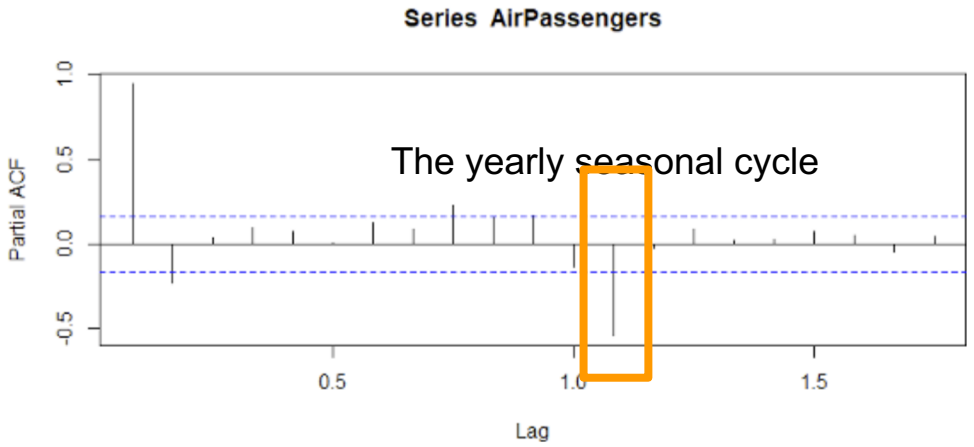


Taken from "Practical Time Series Analysis", Aileen Nielsen, O'Reilly 2019.

ACF and PACF in the AirPassenger time series







Taken from "Practical Time Series Analysis", Aileen Nielsen, O'Reilly 2019.

Time Series-Specific Exploratory Methods

Stationarity:

- What is a stationary time series and how can we measure it?

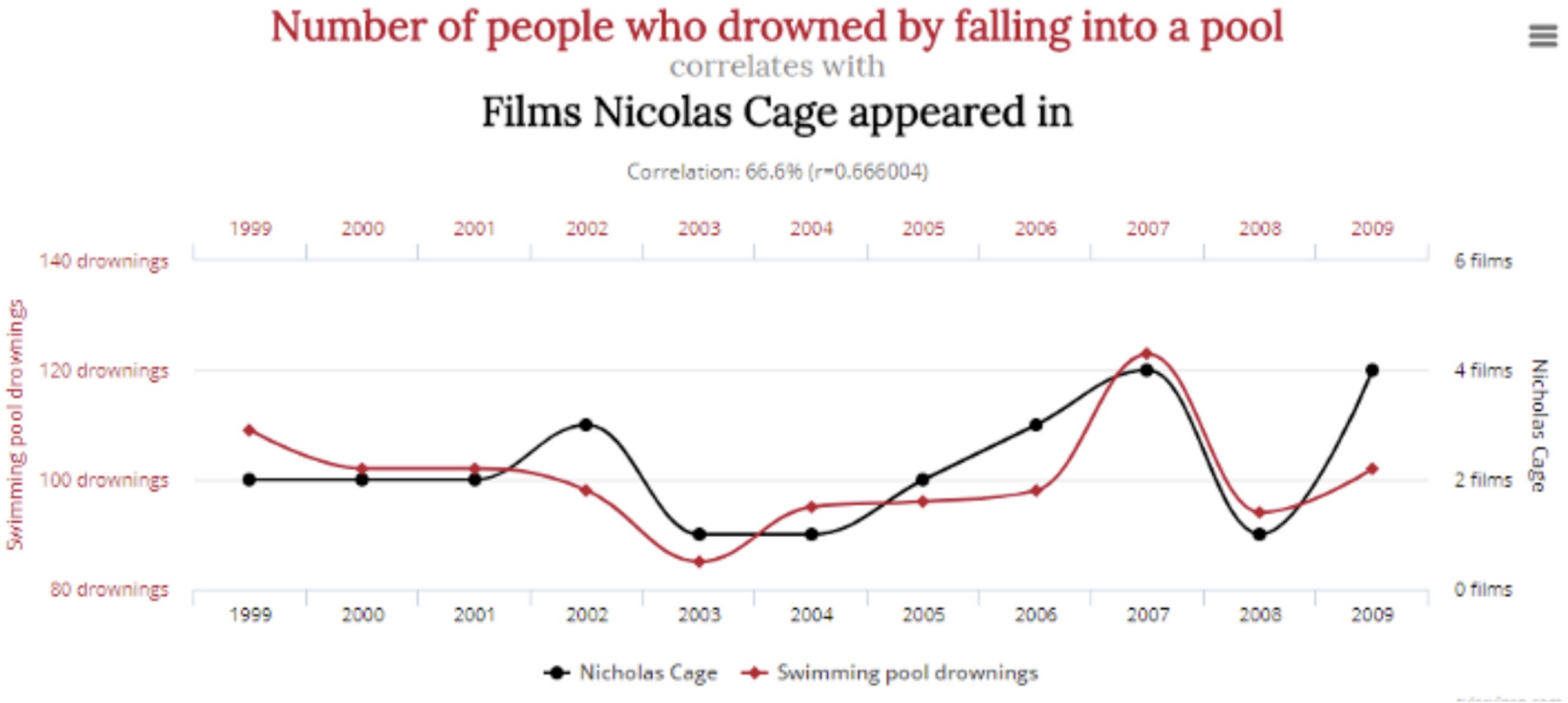
Self-correlation

- How to evaluate and determine a time series is correlated with itself? How can it be useful for understanding the underlying dynamics?

Spurious correlations

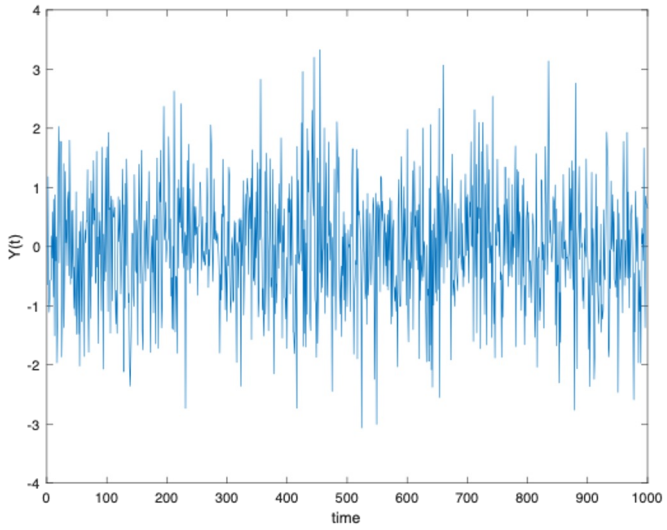
- What is a spurious correlation and how can we deal with that?

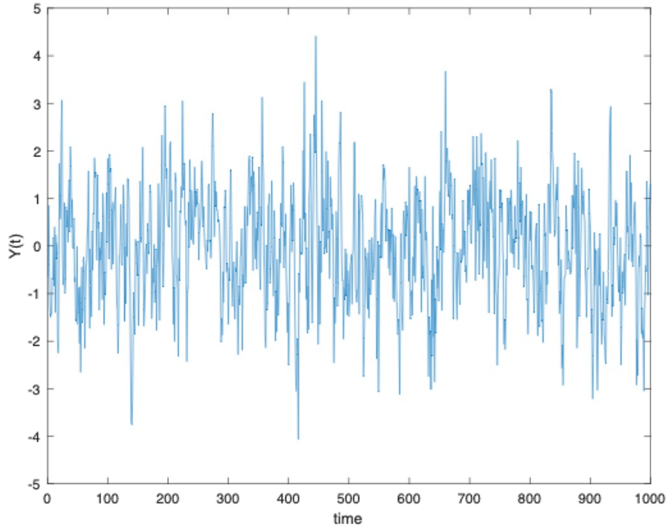
An example of spurious correlation



This plot was taken from Tyler Vigen's website of spurious correlations:
https://www.tylervigen.com/spurious-correlations

Breakout session

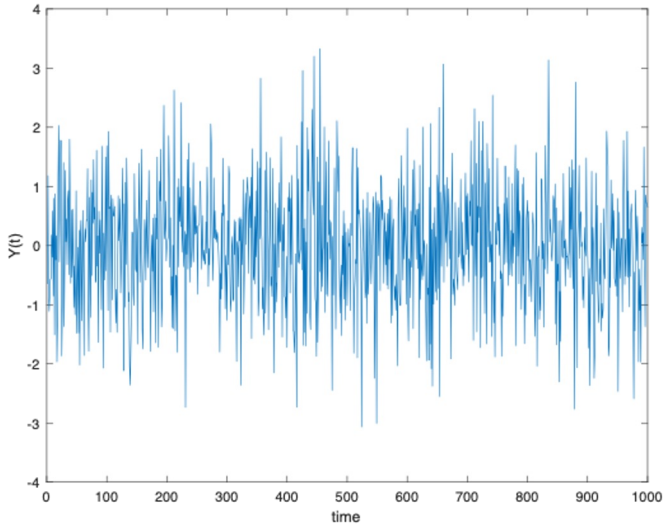


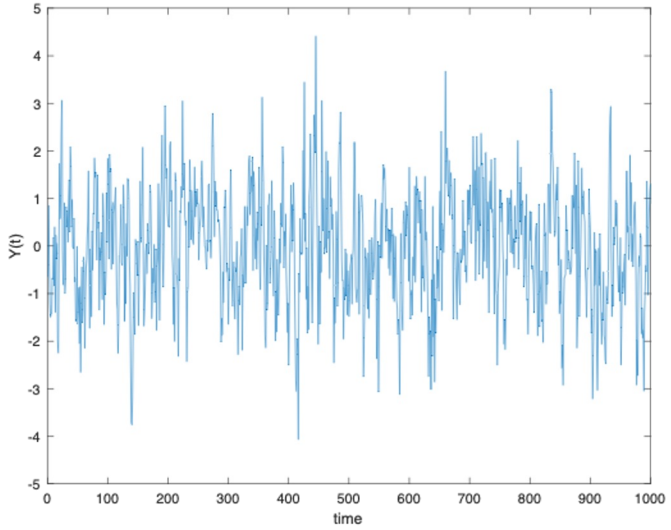


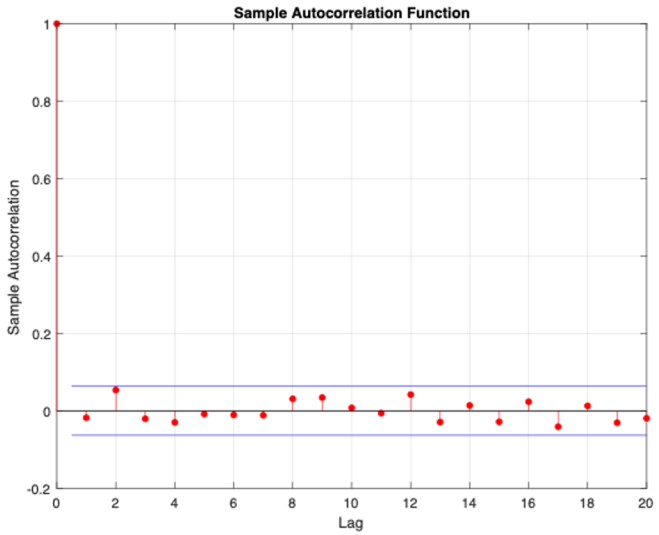
Look at the role of ACT for “mining” time series ...

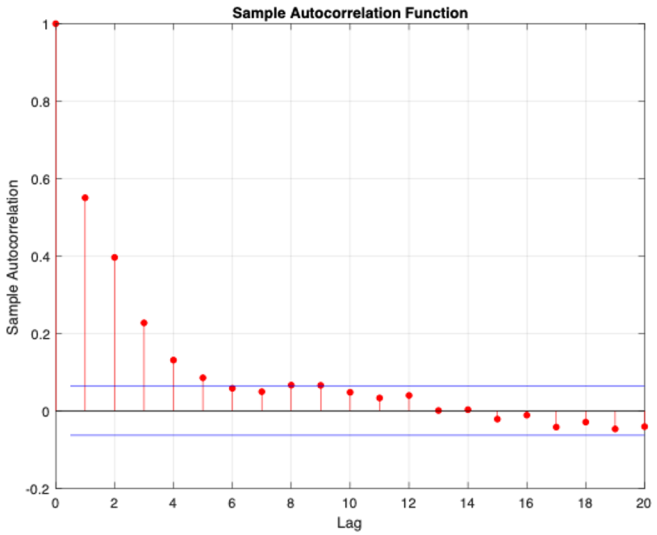
1. Which is the meaningful time series here?

Breakout session



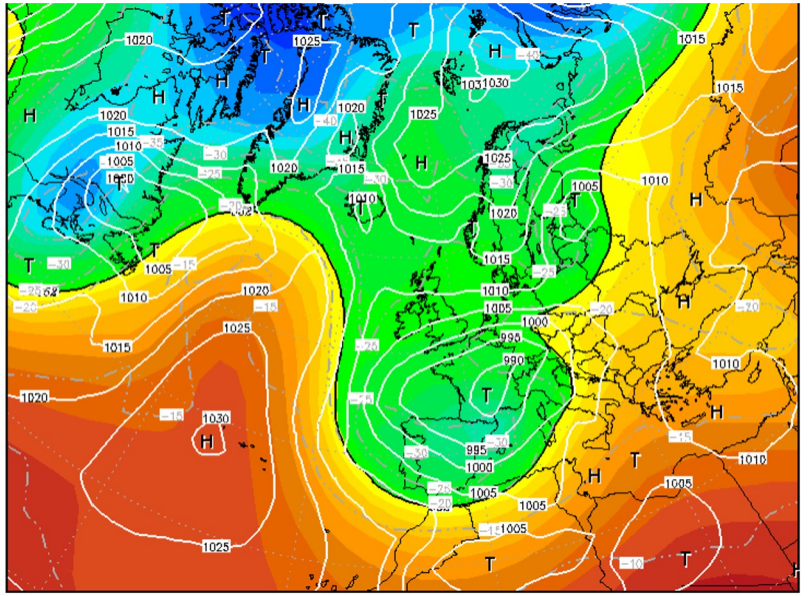






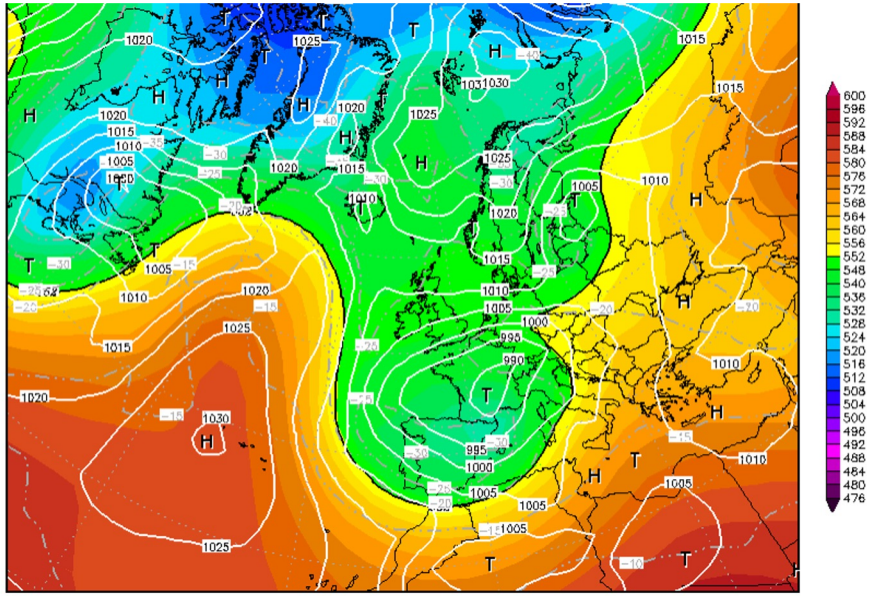
Statistical Models for Time Series: what does it mean “modelling”?

$Y = g(X) + e$



$Y = f(X)$

What does it mean “modelling”?



$Y = g(X) + e$ *Unavailable*

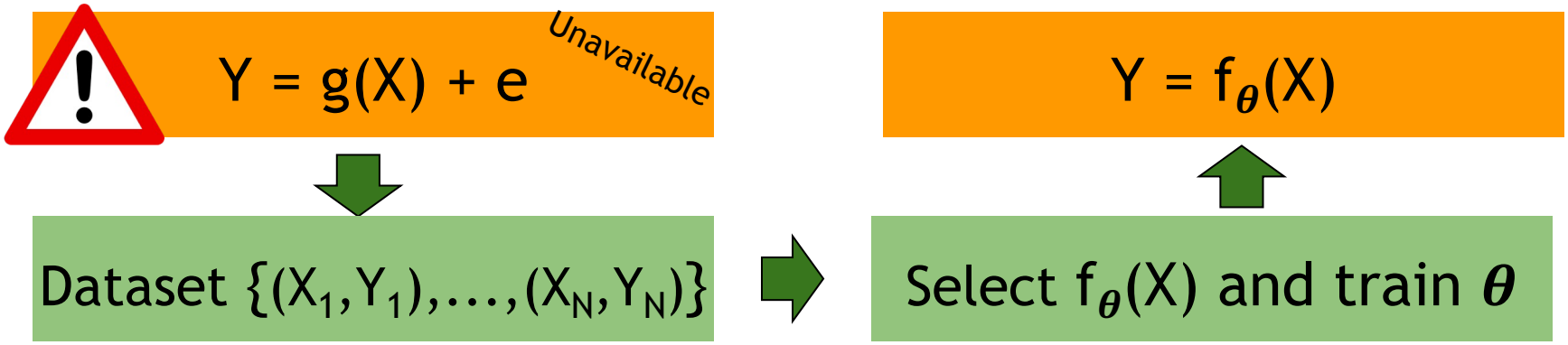
$Y = f_{\boldsymbol{\theta}}(X)$

Dataset $\{(X_1, Y_1), \ldots, (X_N, Y_N)\}$

Select $f_{\boldsymbol{\theta}}(X)$ and train $\boldsymbol{\theta}$

Two major issues with “modelling”

- Properly select $f_{\boldsymbol{\theta}}(X)$:
  - What if $f_{\boldsymbol{\theta}}(X)$ is linear while $g(X)$ is non linear?
- Have enough data to train $\boldsymbol{\theta}$:
  - What if data are not enough for the training?



Linear regression models for time series

- Linear models able to deal with time-dependent data (in contrast to traditional linear models assuming i.i.d. data)
- Specific models:
  - Autoregressive Models (AR), Moving Average (MA) models, Autoregressive Integrated Moving Average (ARIMA) models
  - Vector autoregression models (VAR)
  - Hierarchical models

The need for linear regression

$$Y = f(X) = a\,X + b$$

- Linear models assumes to have i.i.d. data, i.e., couples (X, Y) are independent and identically distributed.
- This is not true in time series where points in time are correlated

?

Linear regression for time series

System: $X_t = a_1 X_{t-1} + a_2 X_{t-2} + \ldots + a_N X_{t-N} + e$

Model: $X_t = a_1 X_{t-1} + a_2 X_{t-2} + \ldots + a_N X_{t-N}$

Assumptions with respect to the behavior of the time series

- The time series has a linear response to its predictors.
- No input variable is constant over time or perfectly correlated with another input variable. This simply extends the traditional linear regression requirement of independent variables to account for the temporal dimension of the data.

When these assumptions hold (together with the ones on the errore), one could resort on the Ordinary Least Squares (OLS) regression to estimate the model, i.e., to get an unbiased estimator of the coefficients (a_1,...a_N)

Linear regression models for time series

- Linear models able to deal with time-dependent data (in contrast to traditional linear models assuming i.i.d. data)
- Specific models:
  - Autoregressive Models (AR), Moving Average (MA) models, Autoregressive Integrated Moving Averaga (ARIMA) models
  - Vector autoregression models (VAR)
  - Hierarchical models

Autoregressive Models (AR)

- The future is predicted by past data: **the value of the time series at time t depends on the value of the time series at the earlier time instants.**

- How many points in the past? **This is the order p of the Autoregressive Model AR(p)**

- In this case we have AR(1):

$$X_t = a_1 X_{t-1} + a_0$$

The value of X at time t depends on the value of X at time t-1 multiplied for a coefficient plus a constant term

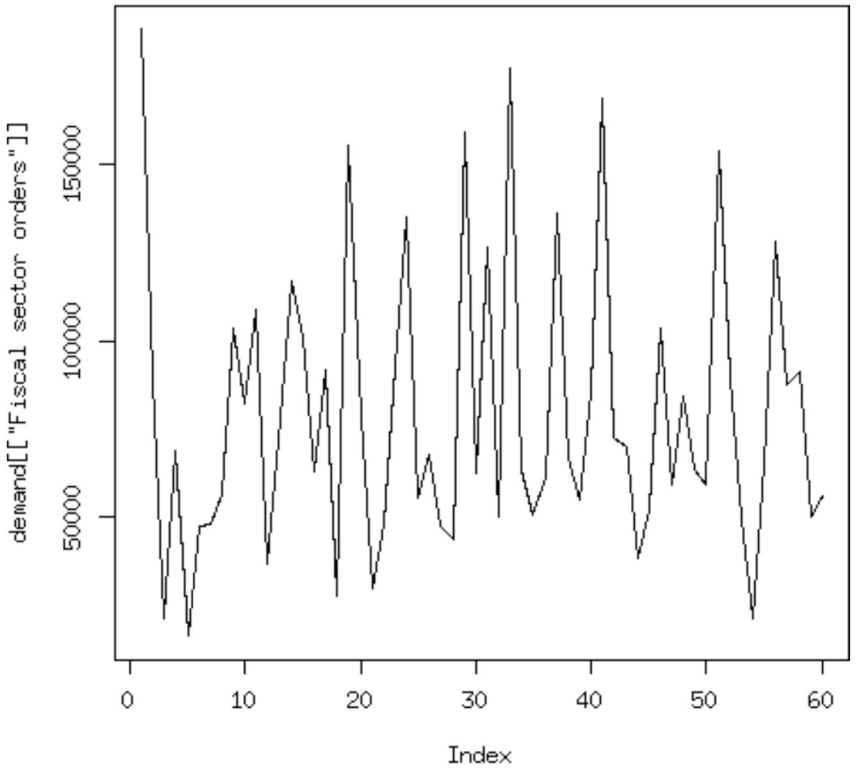
Autoregressive Model AR(p)

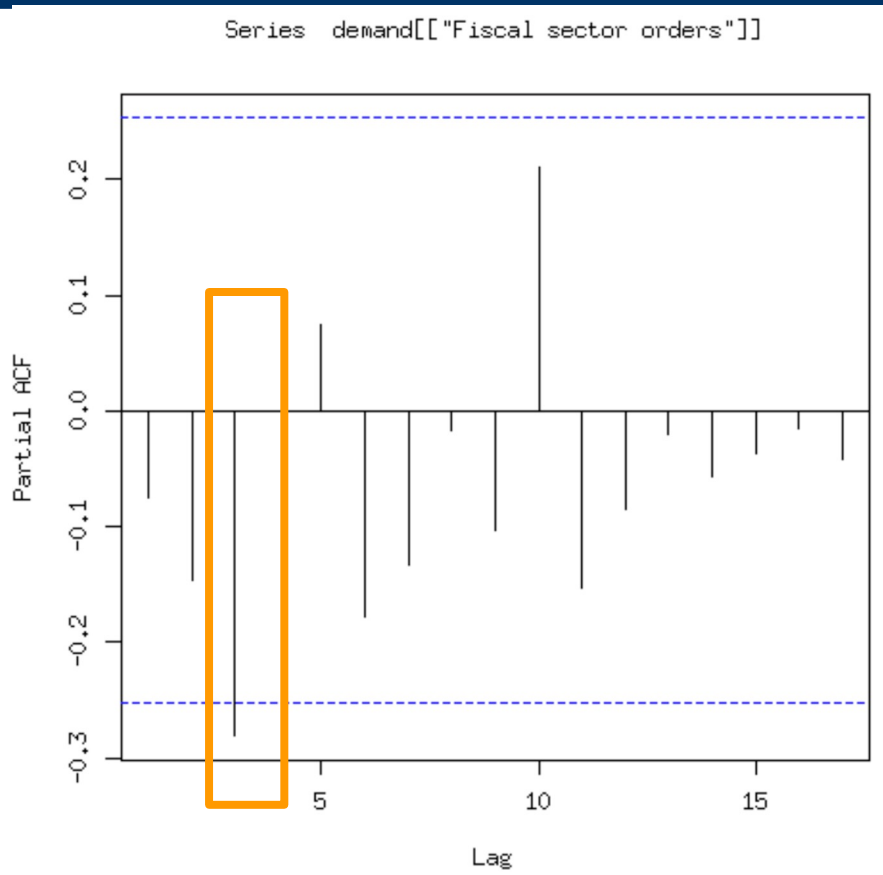
$$X_t = a_0 + a_1 X_{t-1} + a_2 X_{t-2} + \ldots + a_p X_{t-p}$$

How can we select p of AR(p)?

Let's have a look at the partial autocorrelation function (PACF)

The PACF of an AR process should cut off to zero beyond the order *p* of an AR(*p*) process, giving a concrete and visual indication of the order of an AR process empirically seen in the data





The value of the PACF crosses the 5% significance threshold at lag 3.

Taken from "Practical Time Series Analysis", Aileen Nielsen, O'Reilly 2019.

Compute the residual to verify the modeling phase

Let assume that our **"true" system** is

$$x(t) = a\,x(t-1) + b\,x(t-2) + e$$

where e is the noise (e.g., Gaussian and iid) and that **our model** is

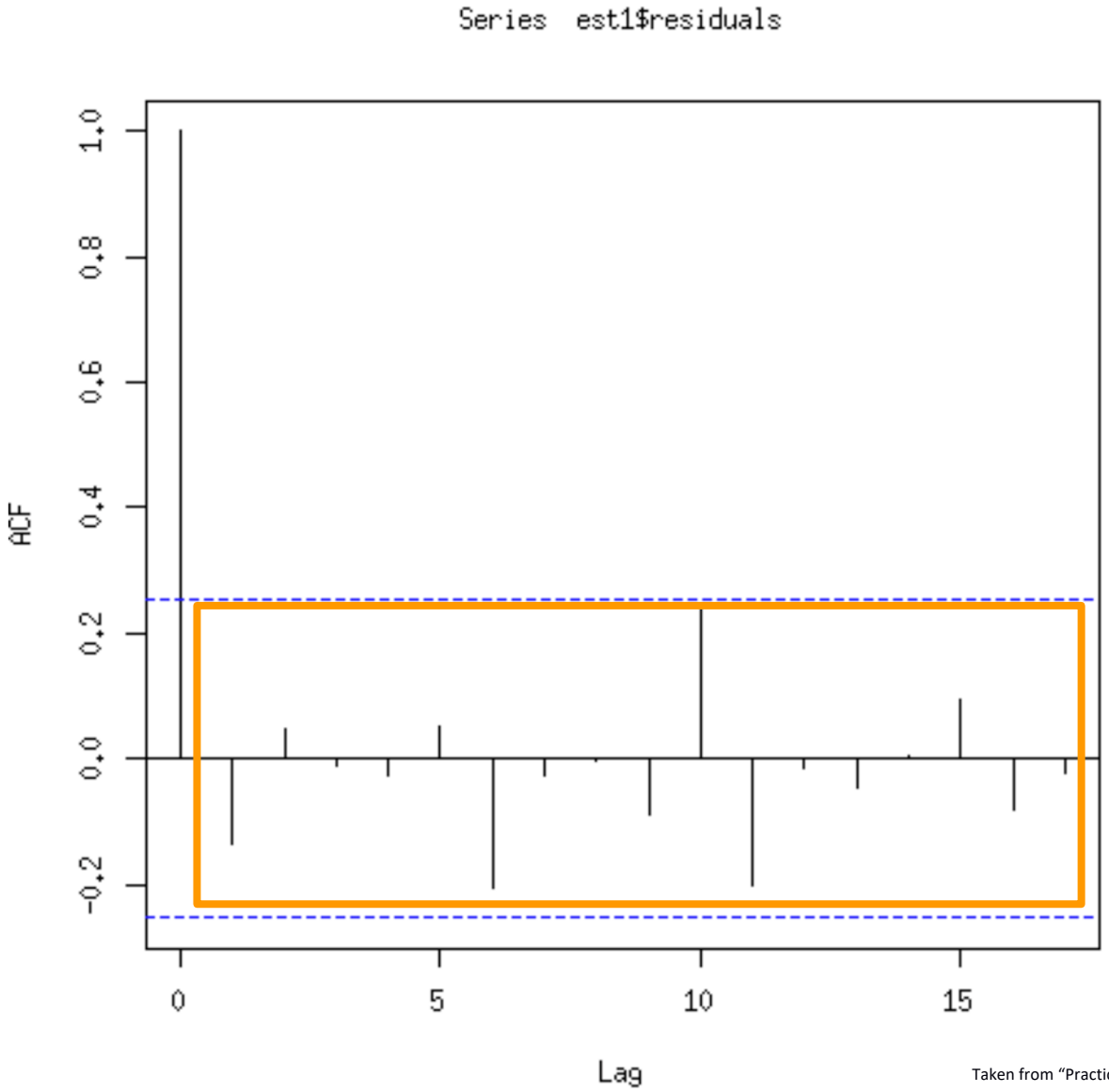
$$\overline{x}(t) = c\,x(t-1) + d\,x(t-2)$$

Let assume that **we have data to estimate c and d. If the true system is linear and we have enough data**, by using OLS, we get $c \backsimeq a$ and $d \backsimeq b$. Hence, **we can compute the residual:**

$$r(t) = x(t) - \overline{x}(t) \backsimeq e$$

The **analysis of the residual could tell us much of the modelling!**

Analysing the residual: mean, std and ACF



Taken from "Practical Time Series Analysis", Aileen Nielsen, O'Reilly 2019.

Moving Average (MA) Models

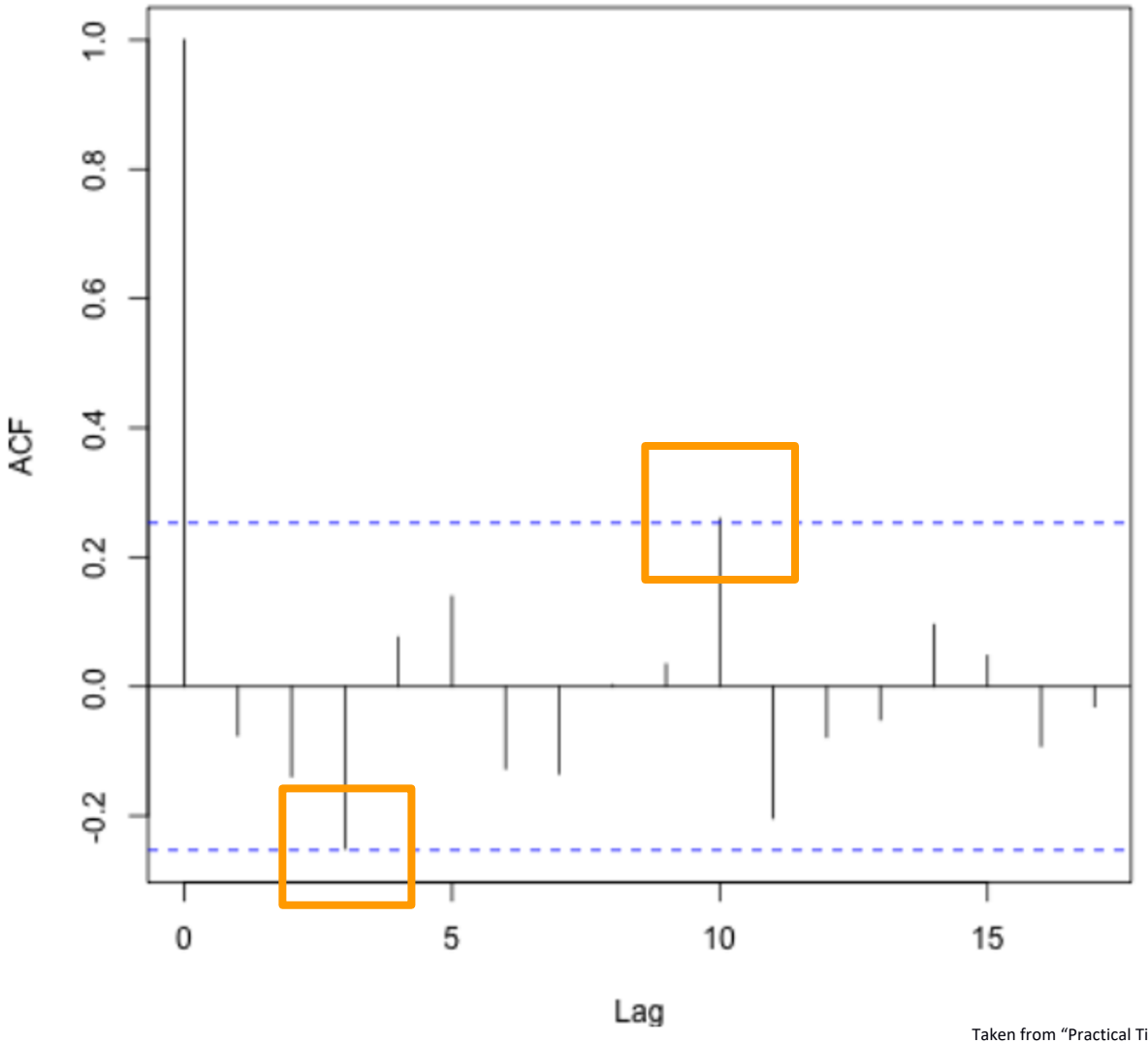
- Moving average (MA) models rely on the assumption that the **value of the time series at a given time instant depends on the recent past value "error" terms**

- MA model of order q: MA(q)

$$X_t = a_0 + a_1 e_{t-1} + a_2 e_{t-2} + \dots + a_q e_{t-q}$$

- MA models can be expressed as an infinite order AR processes
- In economy e_{t-1} , e_{t-2} …. e_{t-q} are often called "**shocks**" to the system, while in electrical engineering they represent as a **series of impulses** and the **MA model is finite impulse response filter** (i.e., the effects of a given impulse remain only is limited in time)

Selecting the q parameter of MA(q) by means of the ACF

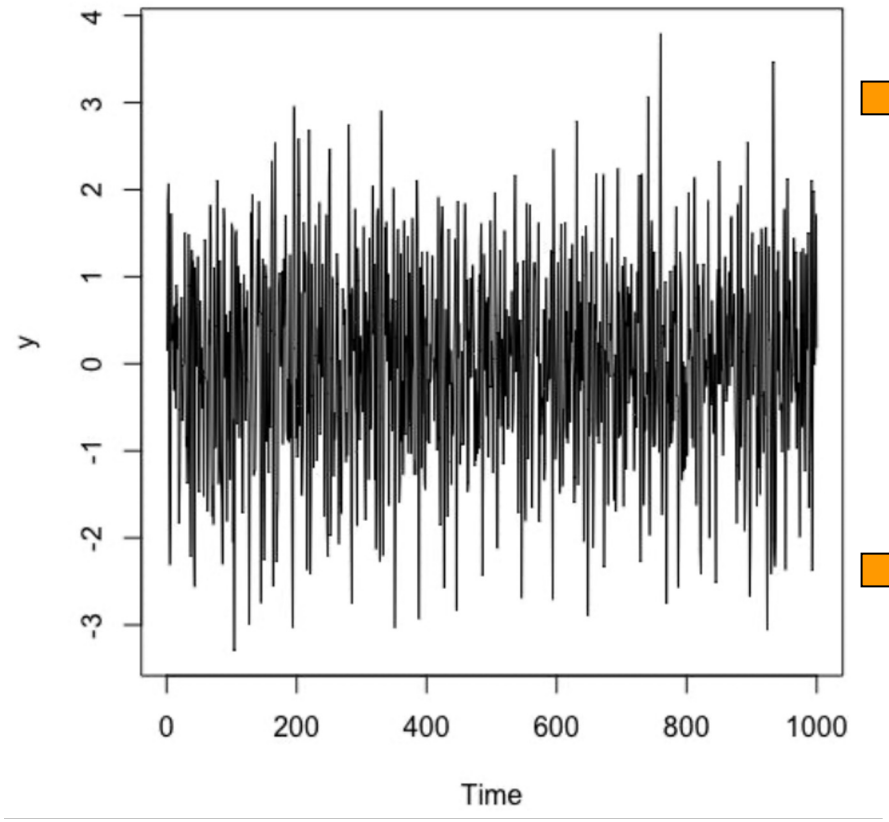


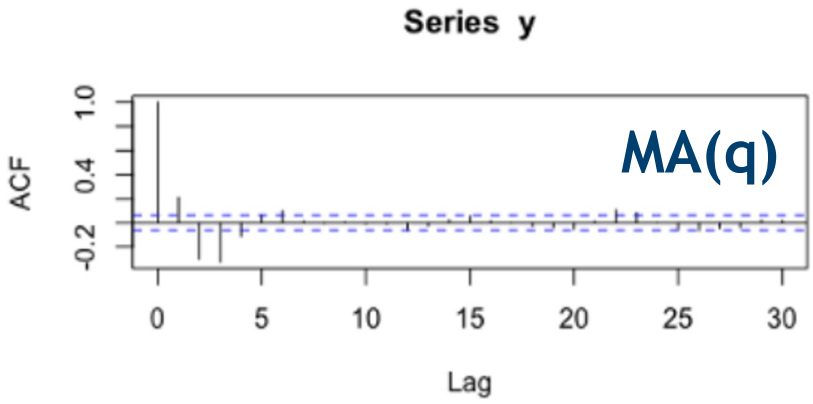
Taken from "Practical Time Series Analysis", Aileen Nielsen, O'Reilly 2019.

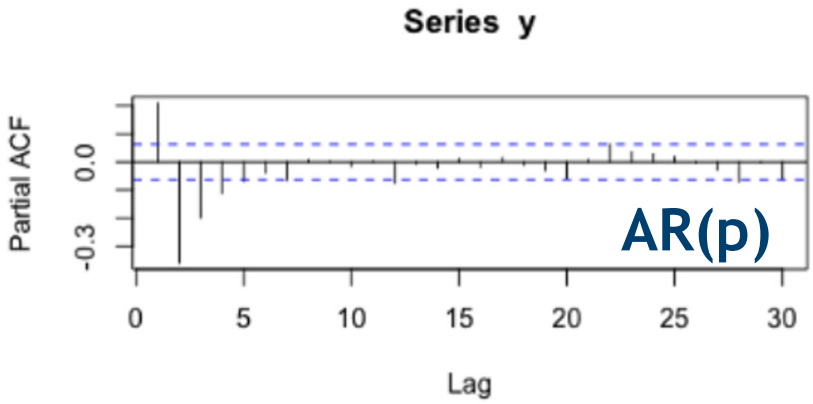
Autoregressive Integrated Moving Average Models (ARIMA)

- **Combining AR and MA models** (by also taking into account trends with the **"integrated" component**)
- "Differencing" refers to the computation of the pairwise difference of adjacent points in time
- These models can provide state-of-the-art performance, particularly in cases of small data sets where more sophisticated machine learning or deep learning models are suitable
- ARIMA (p,i,q):
  - p is the order of AR
  - q is the order of MA
  - i is the differencing order

Selecting p and q of ARIMA(p,i,q) with ACF and PACF







- Evaluating the residual
- Using automated model fitting

Taken from "Practical Time Series Analysis", Aileen Nielsen, O'Reilly 2019.

Vector Autoregression

How to deal with N multivariate time series? Multiple and integrated AR(p) models:

$$X1_t = a_{1,0} + a_{1,1}\, X1_{t-1} + a_{1,2}\, X_{t-2} + \ldots + a_{N,1}\, XN_{t-1} + a_{N,2}\, XN_{t-1}$$

...

$$XN_t = a_{N,0} + a_{1,N}\, X1_{t-1} + a_{1,N}\, X_{t-1} + \ldots + a_{N,N}\, XN_{t-1} + a_{N,N}\, XN_{t-1}$$

- Able to learn dependencies among the N time series
- Typically on the AR component is used

Other types of statistical models

- Seasonal ARIMA:
  - ARIMA(p,i,q) + the modelling of a seasonal component
- Hierarchical time series models:
  - Multiple time series are aggregated at different granularities

Advantages and Disadvantages of Statistical Methods for Time Series

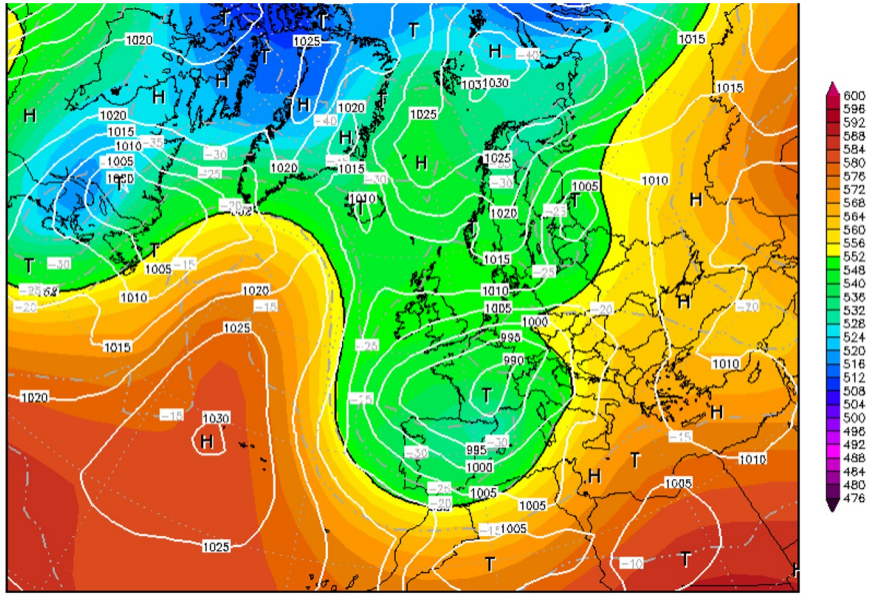
Advantages and Disadvantages of Statistical Methods for Time Series

- Simple and intuitive (good for explainability)
- Theoretically grounded (with rigorous properties)
- Good also for small data set
- Generally perform pretty well (sometimes in line with machine and deep learning models)
- No overfitting
- (Rather) Easy to train and configure

Advantages and Disadvantages of Statistical Methods for Time Series

- They might not take advantage of large data sets (where machine learning and deep learning models might work better)
- They are meant to provide a point-wise forecasting (not easy to model the uncertainty)
- Linear models are suited for linear systems!!!

Machine Learning for Time series
Which is the goal here? Prediction? Classification?



$Y = g(X) + e$ *Unavailable*

$Y = f_{\boldsymbol{\theta}}(X)$

Dataset $\{(X_1, Y_1), \ldots, (X_N, Y_N)\}$

Select $f_{\boldsymbol{\theta}}(X)$ and train $\boldsymbol{\theta}$

The task(s) of Machine Learning

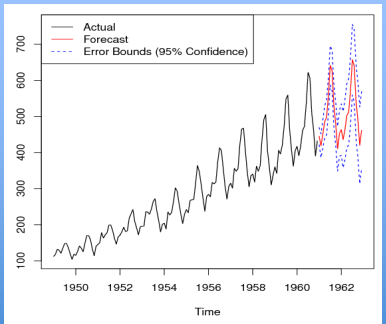
- Up to now we saw models for time series prediction
- Create models for accurate forecasting/prediction
  - ✓ Model identification
  - ✓ Parameter estimation

- Model identification implies the selection of the model family

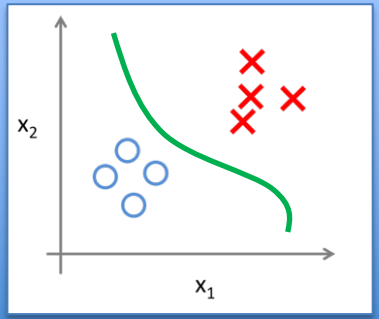
Which are the tasks that we can consider in machine learning?

Supervised and unsupervised learning

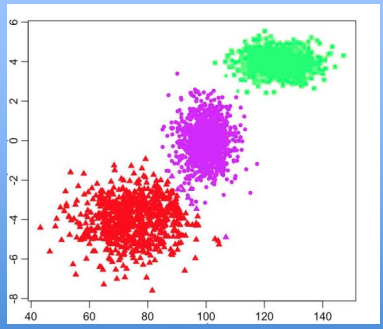
Prediction



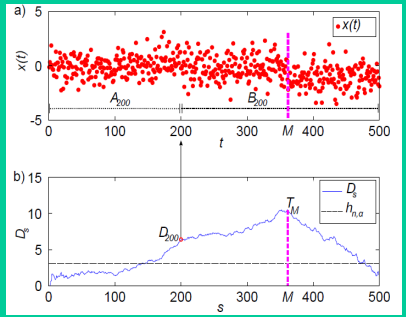
Classification



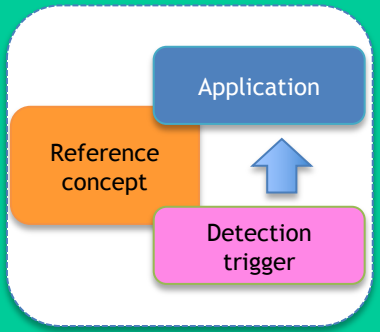
Clustering



Change Detection

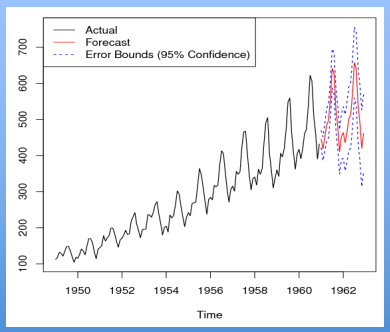


Adaptation

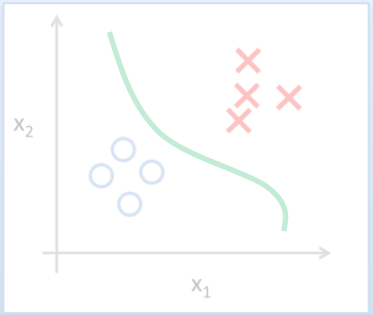


Supervised learning: time series classification

Prediction

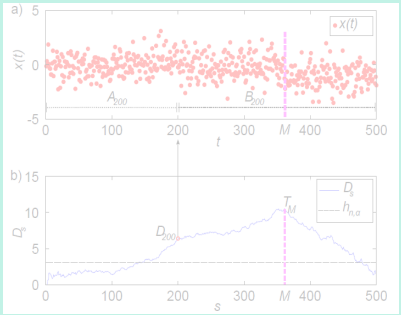


Classification

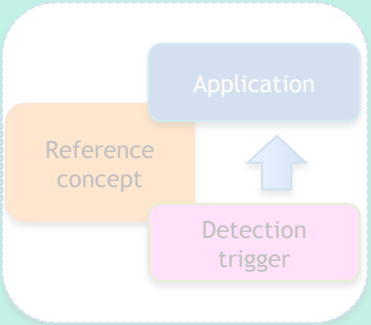


Clustering

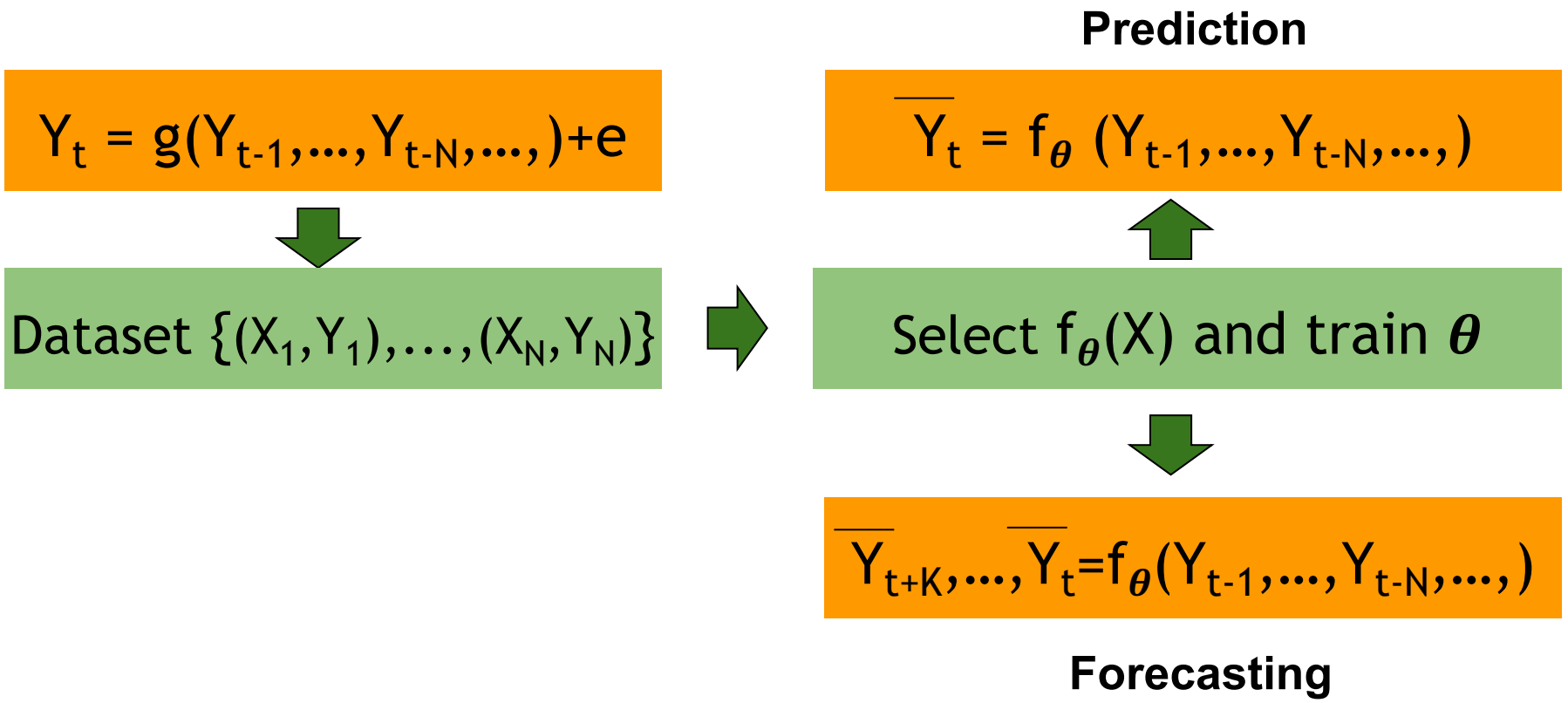
Change Detection



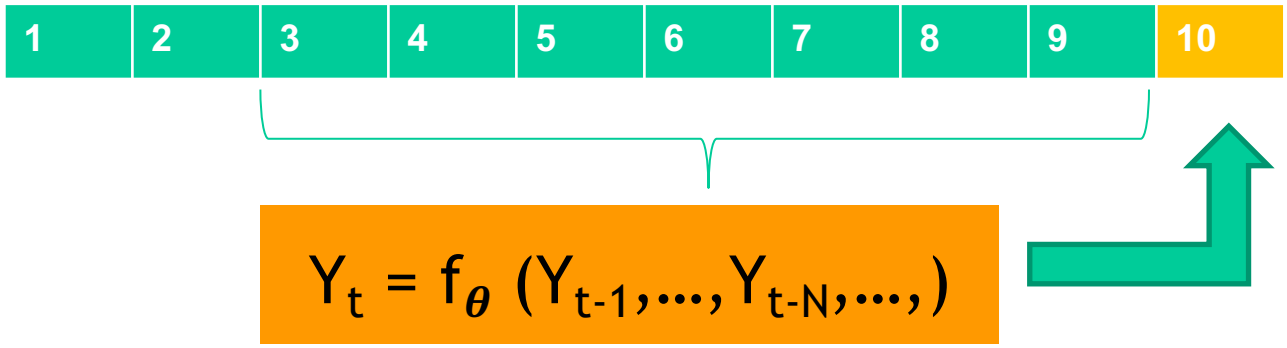
Adaptation

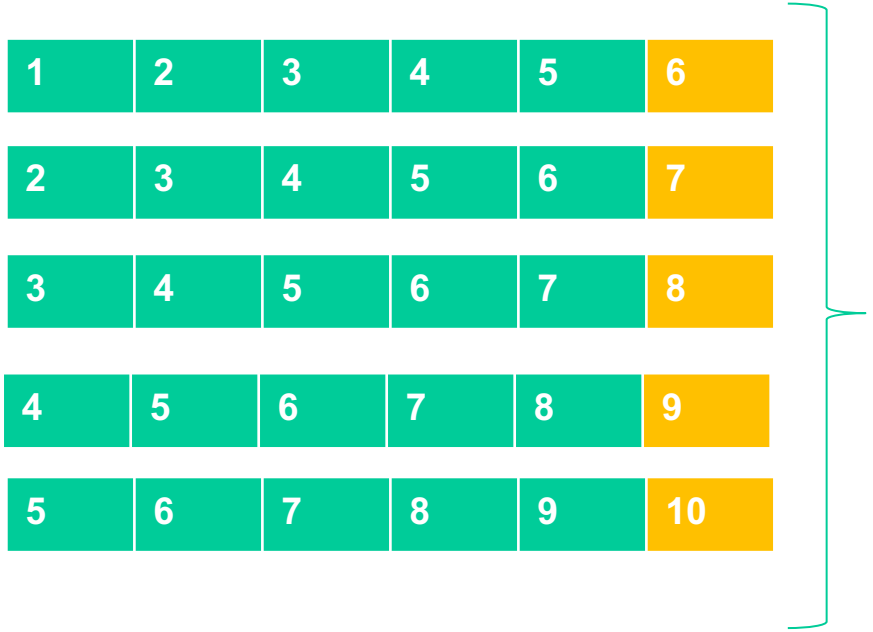


Prediction and forecasting



Prediction as a time-aware regression problem

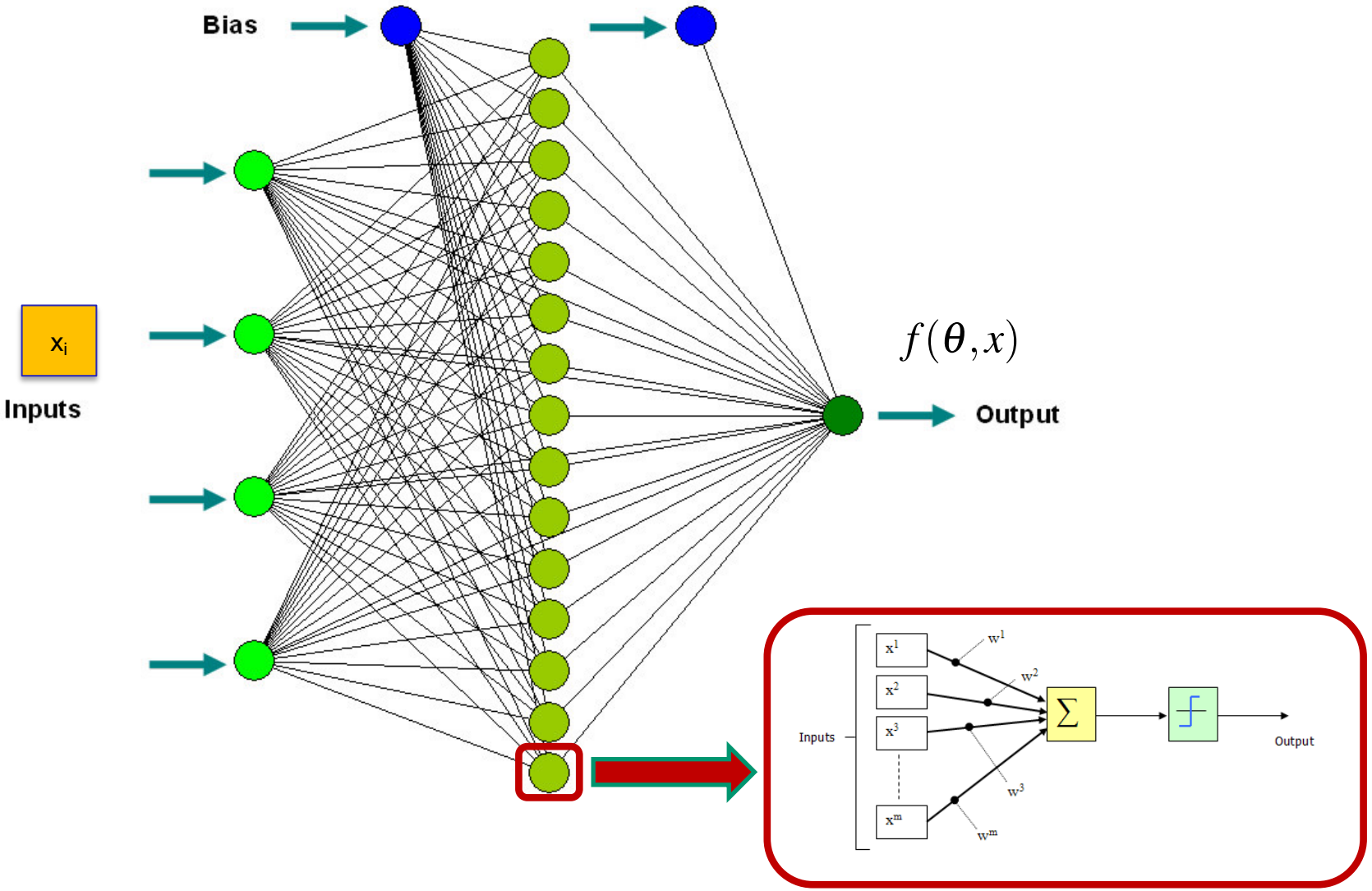




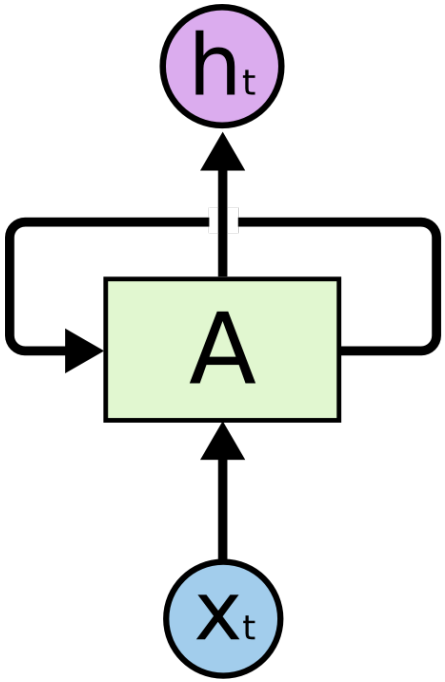
$$Y_t = f_\theta (Y_{t-1},\dots,Y_{t-K})$$

Non-linear regression problem

Multi-layer Feed Forward Neural Networks



What about time?

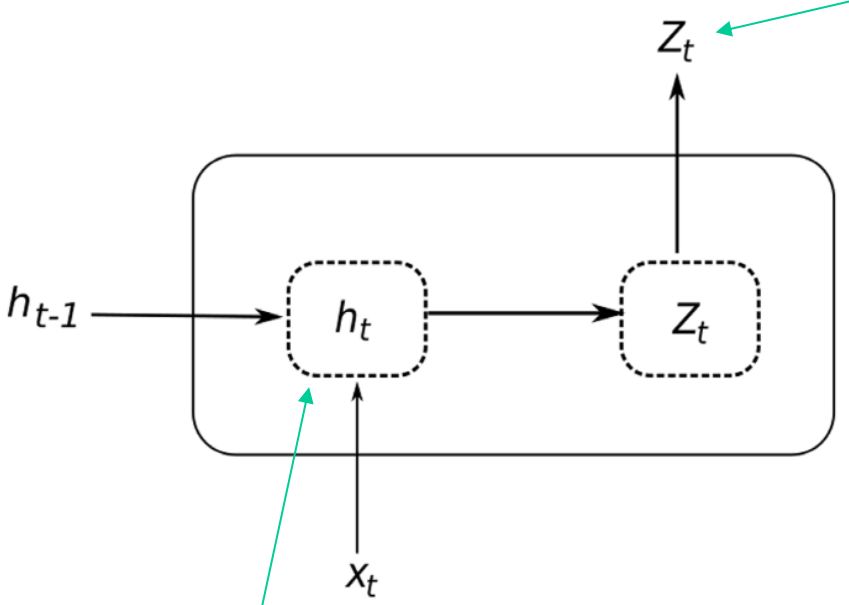


We need to introduce recurrence in our neural networks…

Recurrent Neural Networks

- (Elman) Recurrent Units
- Long short-term memory (LSTM)
- Gated Recurrent Units

(Elman) Recurrent Units



Output $z_t \in \mathbb{R}^d$

$$h_t = \sigma(W_i \cdot h_{t-1} + V_i \cdot x_t + b_i)$$
$$z_t = tanh(W_o \cdot h_t + b_o)$$

Weight matrices

Bias Vectors

$h_t \in \mathbb{R}^d$

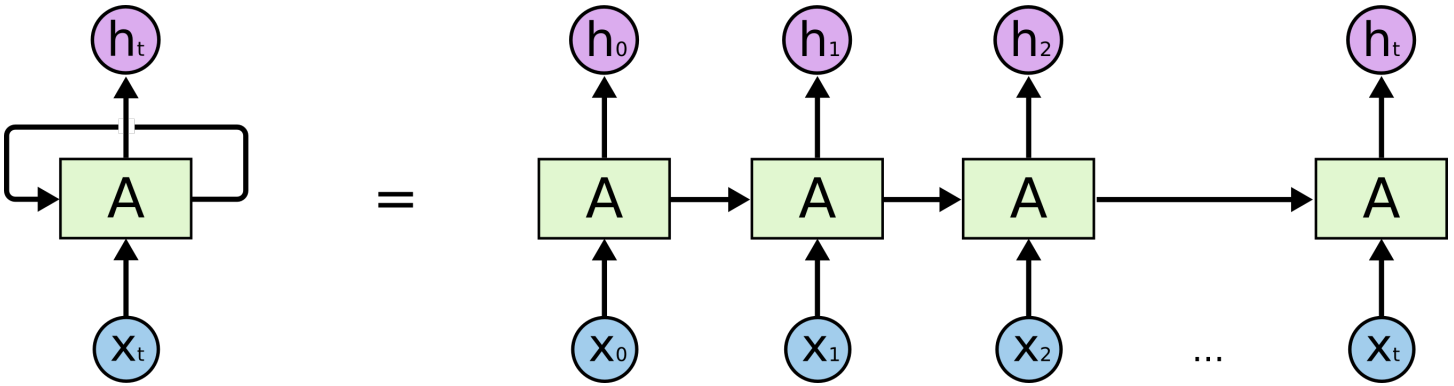
Hidden State
This is the only form of memory in the Elman RNN

Input
$x_t \in \mathbb{R}^m$

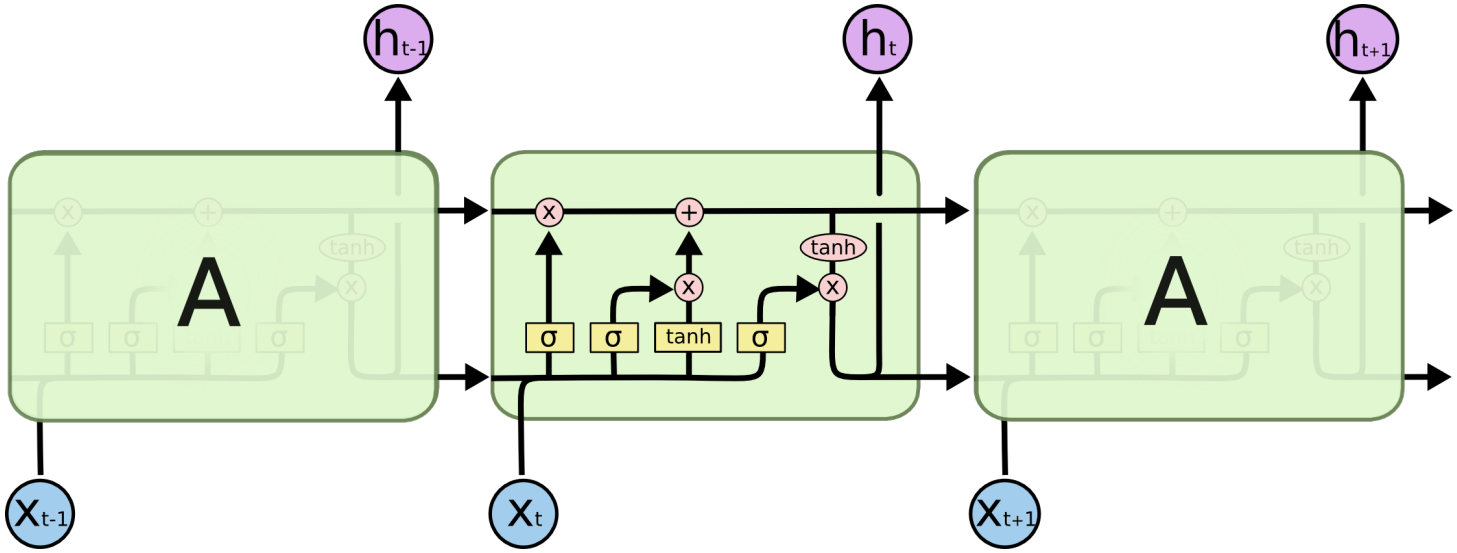
The current hidden state depends on the hidden state of the previous time step as well as the current input.

The vanishing and exploding gradient problems

- Simple RNN cells are **not capable of carrying long-term dependencies** to the future
- In case of long sequences:
  - the **backpropagated gradients tend to vanish** and consequently weights are not updated adequately
  - the **backpropagated gradients can explode** over long sequences resulting in unstable weight matrices



Long short-term memory (LSTM)

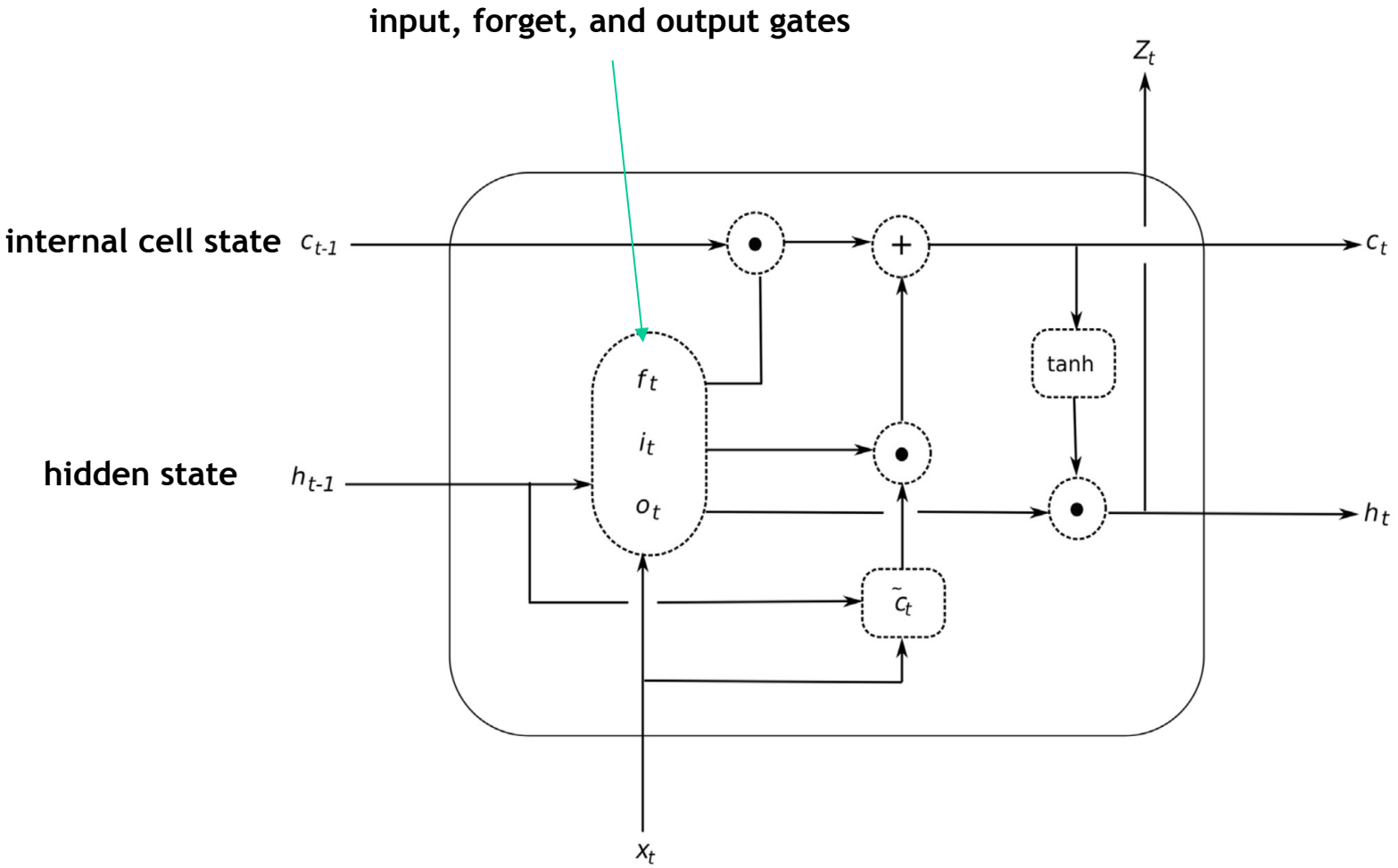


Differently from RNNs, the LSTM cell has two components to its state:

- the **hidden state**: corresponding to the short-term memory component
- the **internal cell state**: corresponding to the to the long-term memory.

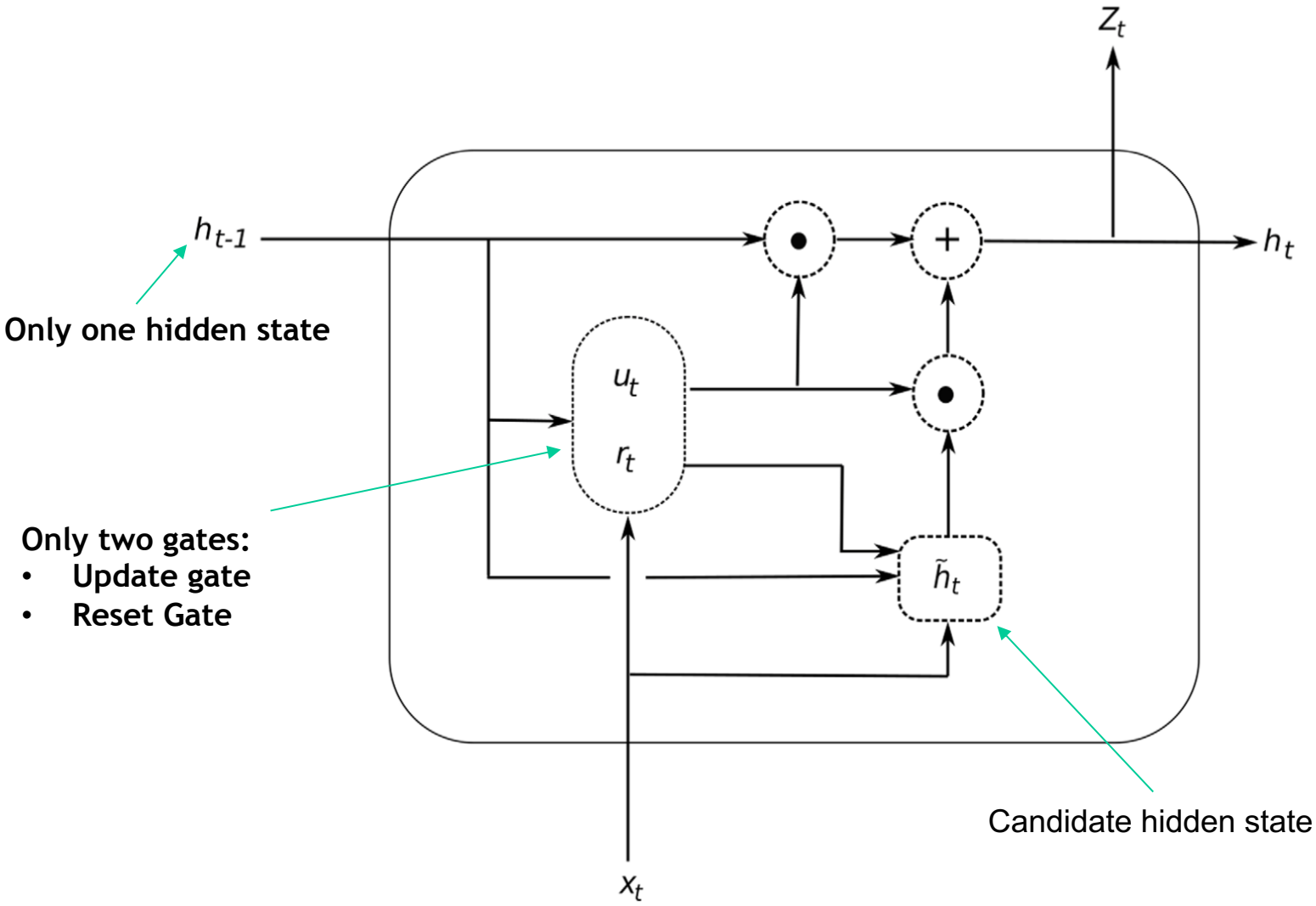
LSTM avoids the vanishing and exploding gradient issue

Time-series prediction with LSTMs



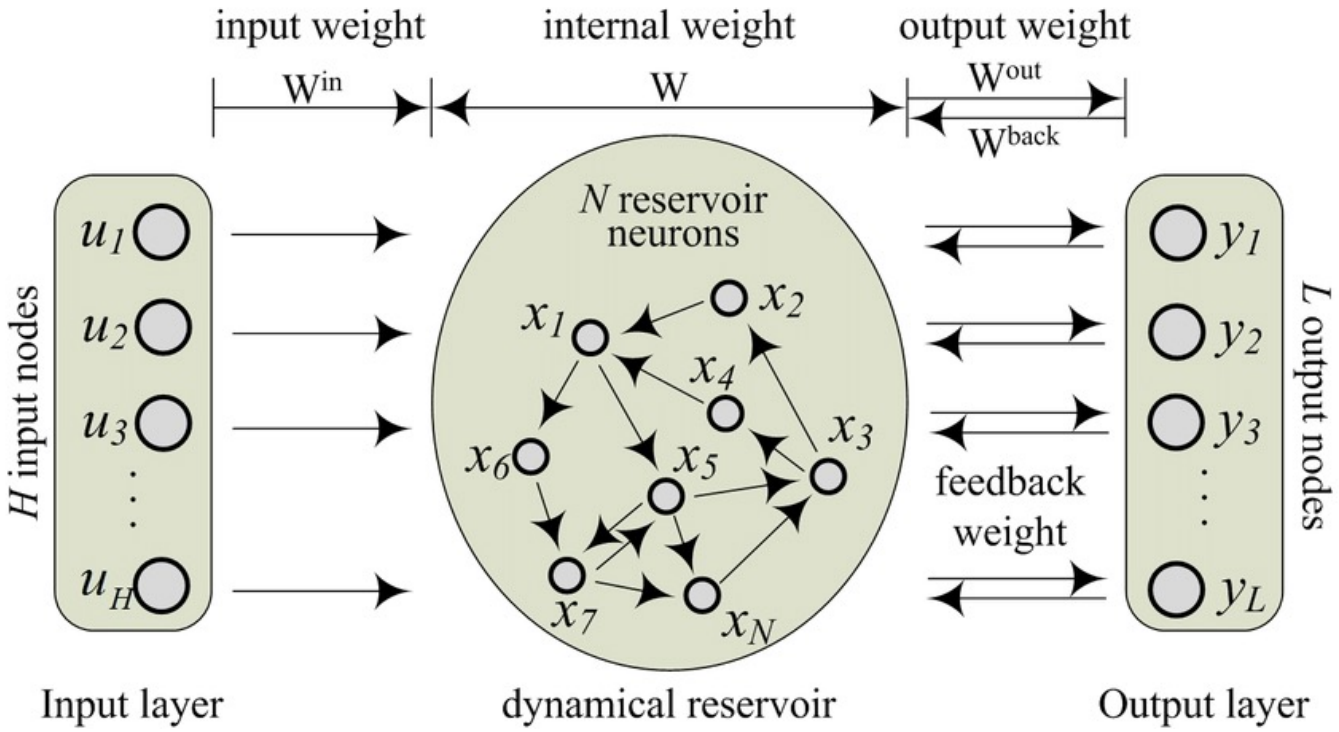
Input and the forget gates together determine how much of the past information to retain in the current cell state and how much of the current context to propagate forward to the future time steps. A value of 0 in ft implies that nothing is carried out from previous step for the cell state.

Gate Recurrent Unit (a simplified version of LSTM)



The GRU is popular thanks to its simplicity (having fewer parameters than LSTM) and its training efficiency

Recurrent neural networks: Echo-state networks



Not trained **Not trained** **Trained**

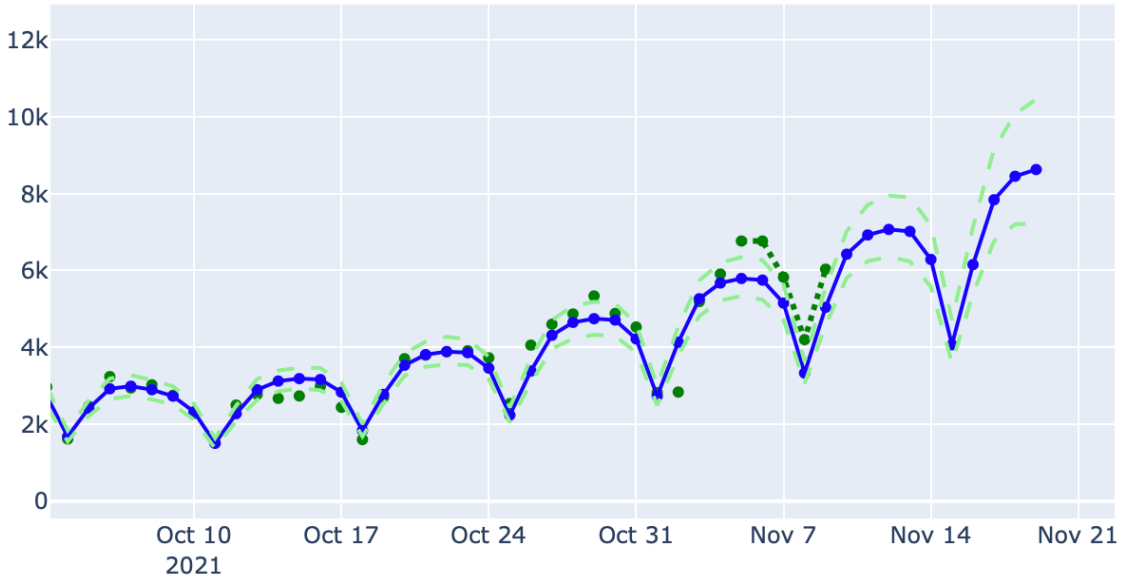
Regression-based Prediction models: Facebook Prophet

$$Y_t = g(t) + s(t) + h(t) + e_t$$

Growth function (modelling linear or log trends)

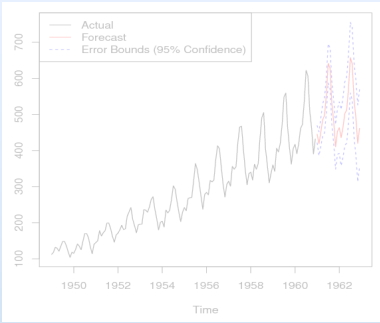
Seasonality function (modelled with a fourier series)

Holiday/Event function

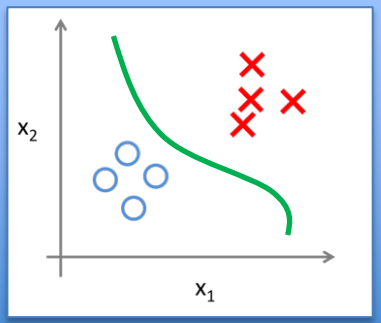


Supervised learning: time series classification

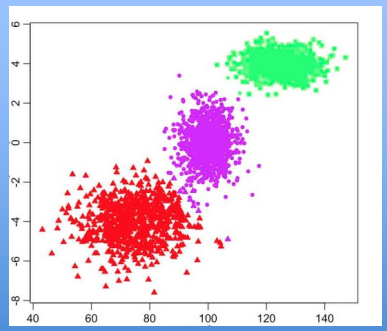
Prediction



Classification

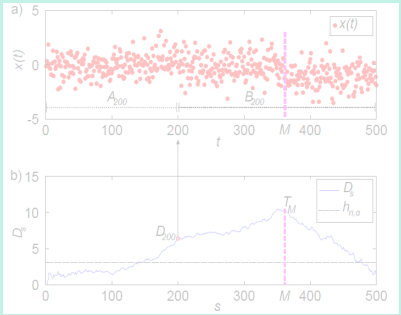


Clustering

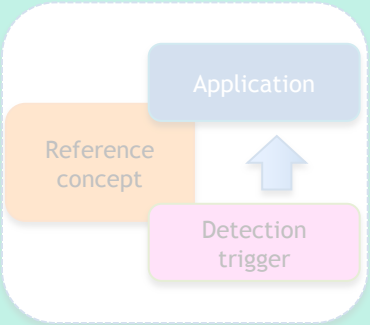


In appendix

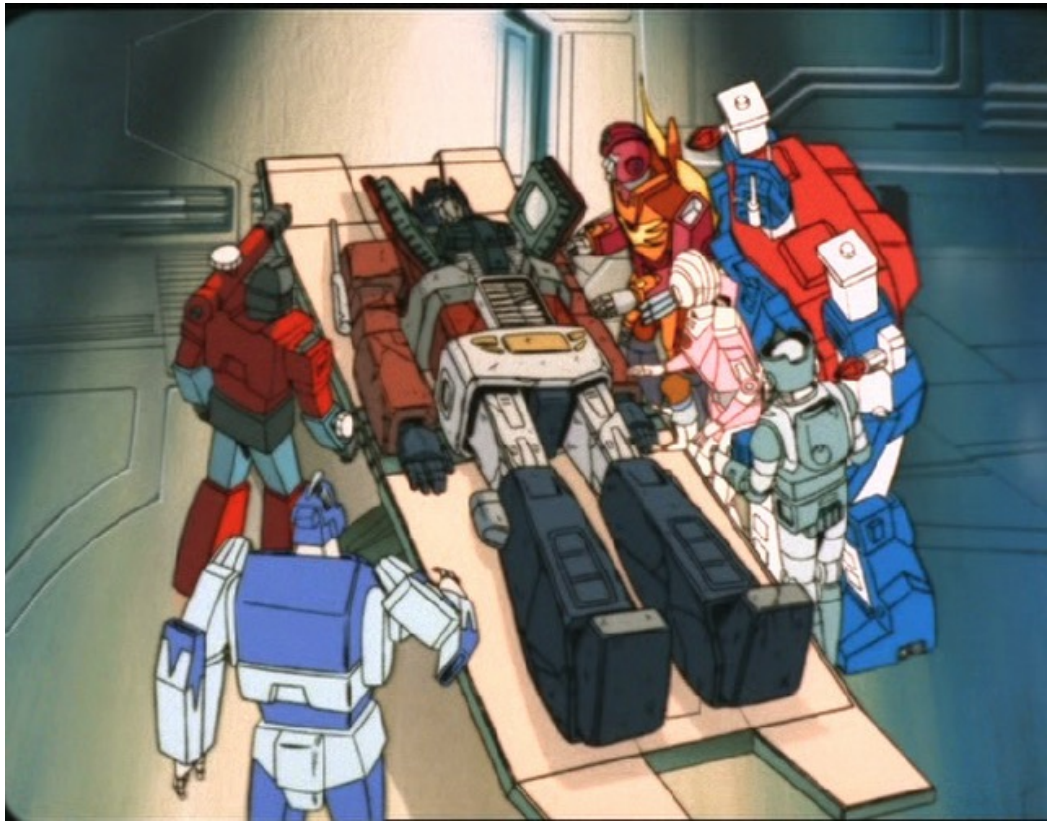
Change Detection

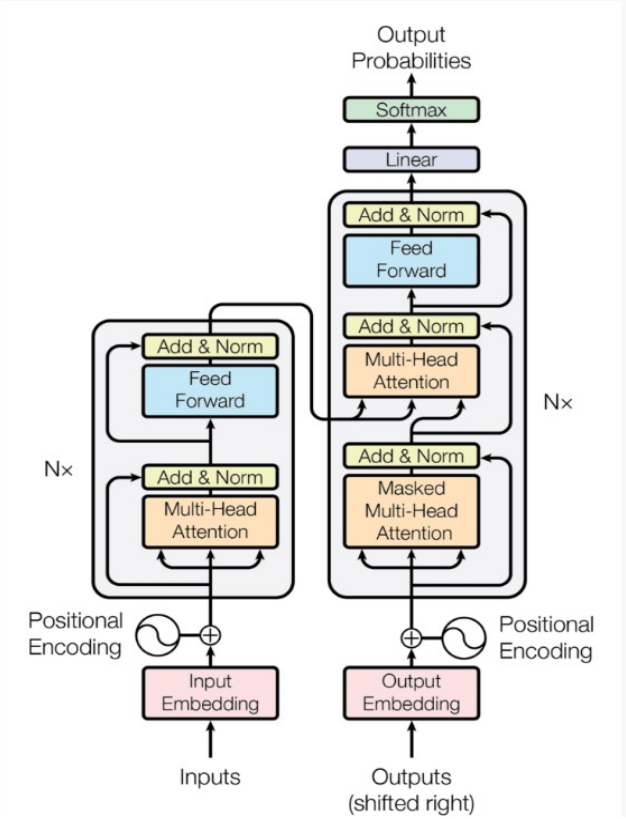


Adaptation



… the fall of Transformers





From recurrent to Transformers

Recurrent Neural Networks (RNNs):

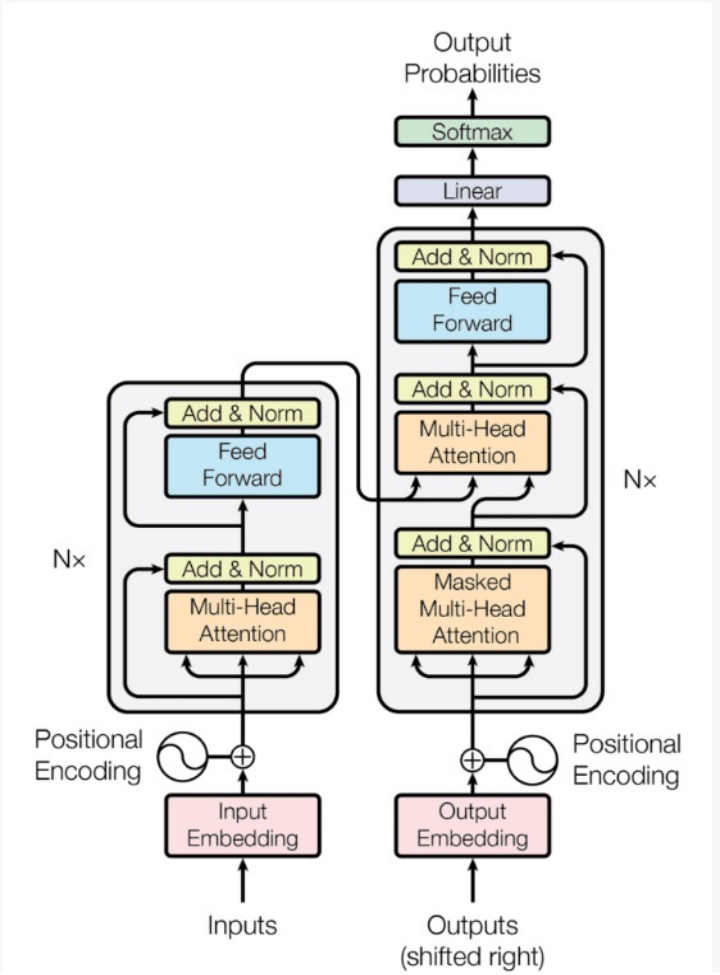
- ✓ PROs: strong results on a variety of natural language processing tasks and on temporal forecasting applications.
- ✓ CONs: limitations in manage long sequences and unstable gradient behaviours.

Long Short-Term Memory (LSTM):

- ✓ PROs: partially mitigate the gradient problem and outperform RNN in almost all applications
- ✓ CONs: still limited in manage long temporal dependencies

A quick recap to Transformers …

- The key point: the attention-based model
- The state of the art in NLP tasks
- How can/must we modify it to deal with time series?



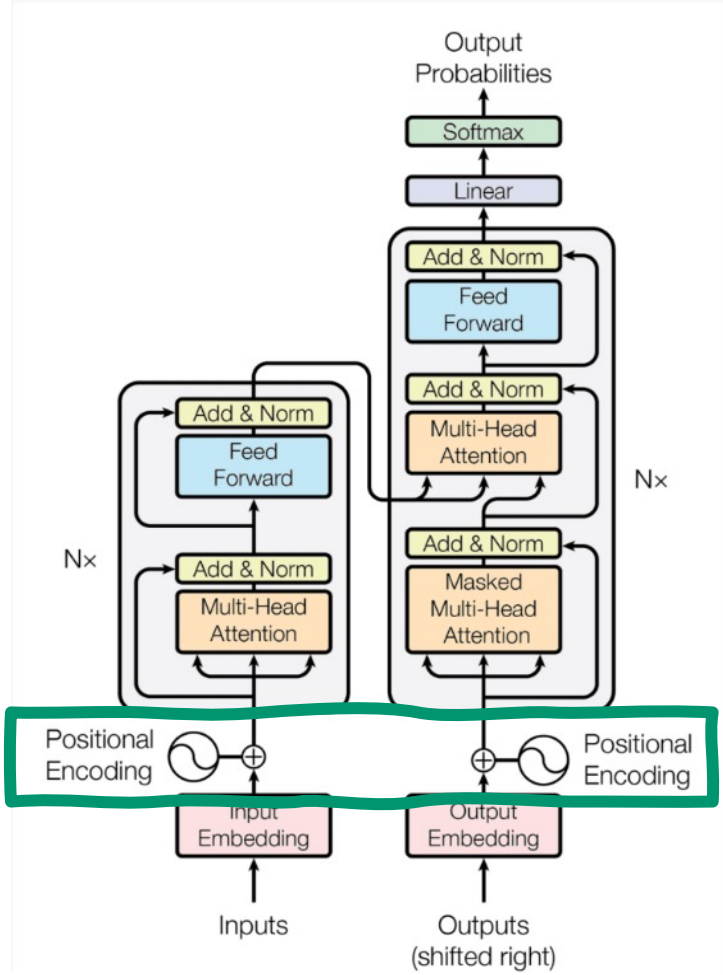
https://arxiv.org/pdf/1706.03762.pdf
https://arxiv.org/pdf/1810.04805.pdf

The encoding layer

Positional Encoding: encoding the position of a sample in the input vector.

$$p_{i,j} = \begin{cases} \sin\left(\frac{i}{10000^{\frac{j}{d_{emb\_dim}}}}\right) & \text{if } j \text{ is even} \\ \cos\left(\frac{i}{10000^{\frac{j-1}{d_{emb\_dim}}}}\right) & \text{if } j \text{ is odd} \end{cases}$$

The model can associate a unique value to each sample in a series and embeds their relative position inside a vector

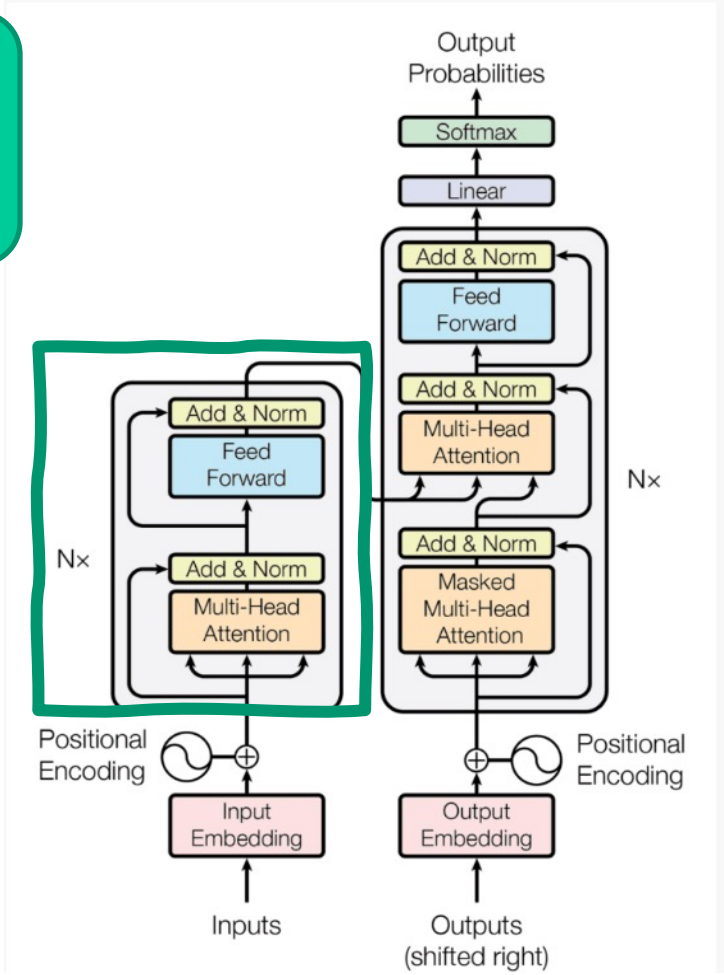


The role of the encoder

Encoder: conceptually captures the context in which the last observations have been generated.

Composed by:

- Multi-Head attention mechanism;
- Normalization layers: to reduce the risk of overfitting and suboptimal signals;
- Feed-forward layer: functional layer to fix the dimension of its input to the next component.

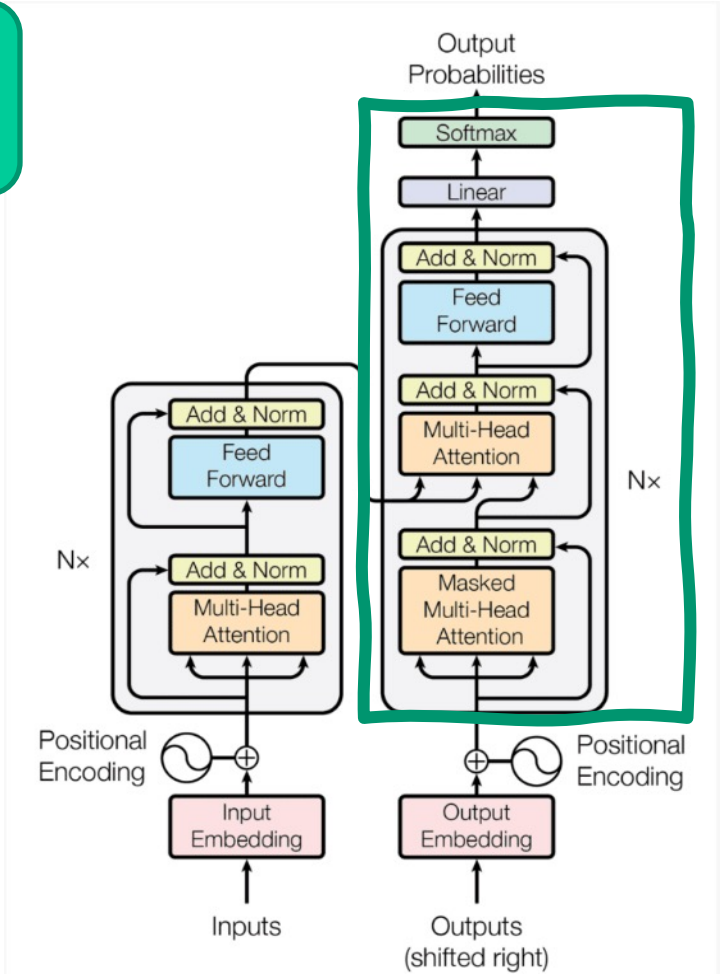


The role of the decoder

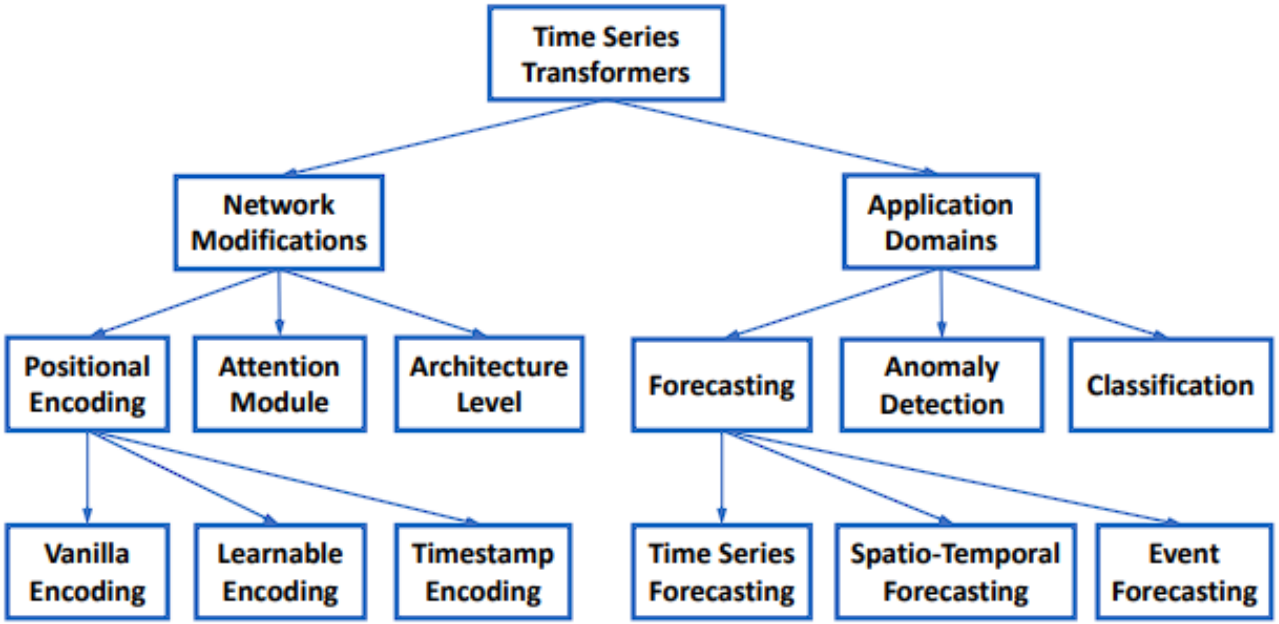
Decoder: interpreting the latest observations within the context defined by the Encoder

- In addition to the two sub-layers in each encoder layer, the decoder inserts a third sub-layer, which performs multi-head attention over the output of the encoder stack.

- The **self-attention mechanism is further modified** to prevent positions from attending to subsequent positions.

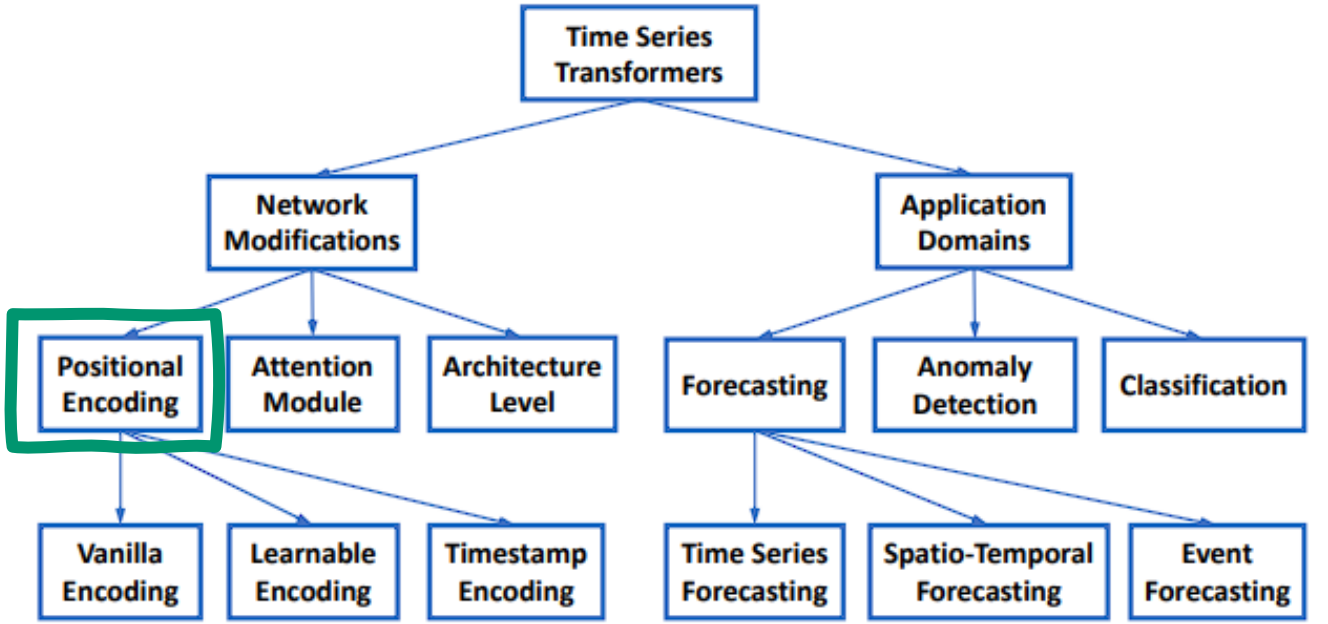


How to modify a transformers for time series?

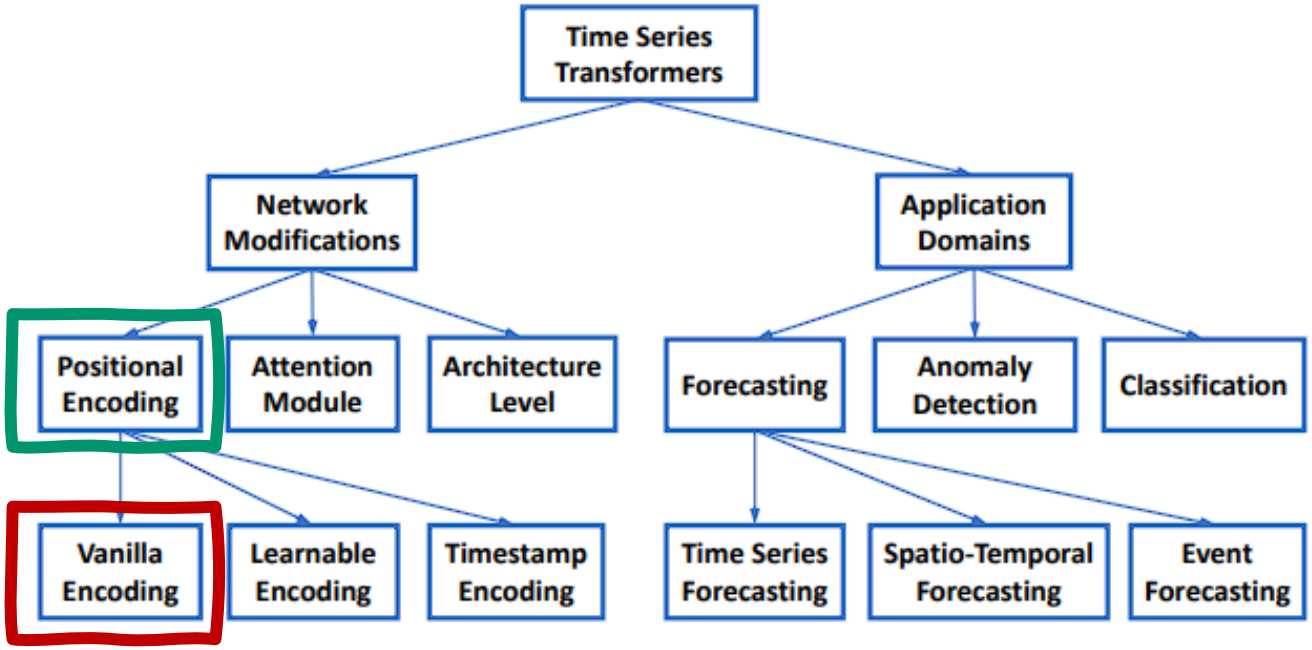


A big (and relevant) challenge: time series deal with real numbers, while NLP has a vast but still limited feature space, related to the vocabulary, to explore.

How to modify a transformers for time series?



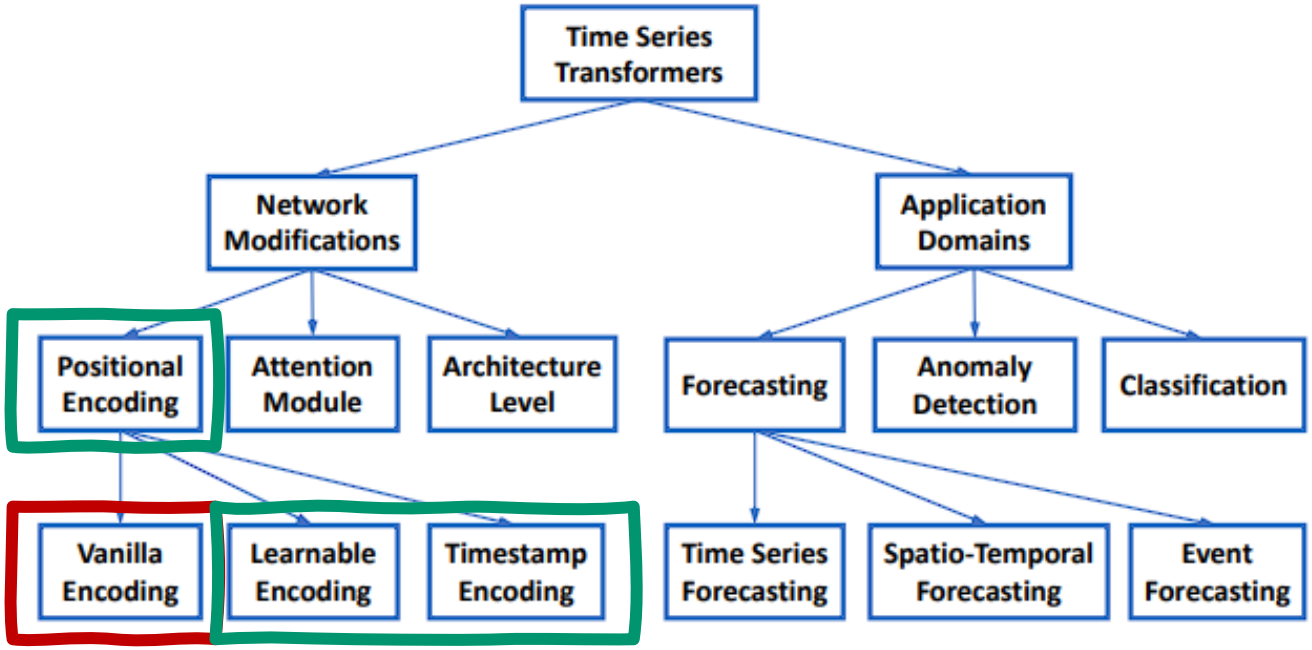
How to modify a transformers for time series?



Vanilla positional encoder:

- **a fixed configuration**
- **not able to adapt its changes in a time dependent domain.**

How to modify a transformers for time series?



Vanilla positional encoder:

- **a fixed configuration**
- **not able to adapt its changes in a time dependent domain.**

Encoding strategies for time series (1/2)

1. **Relative position encoding**:
   representation of the relative positions, or distances be sequence elements fed in the model with the input as a directed, fully-connected graph.
   (https://arxiv.org/pdf/1803.02155.pdf)

2. **Learnable positional encoding:**
   a convolutional layer whose weights are learned during the model training. It makes the model more flexible and adaptable to new temporal patterns. Some works also propose to use LSTM to learn the best possible embedding schema. [See later on in Temporal Fusion Transformer] (https://arxiv.org/pdf/1912.09363.pdf)

Encoding strategies for time series (2/2)

3. Temporal Encoding:

the main idea is to adopt not only the position of a sample of a time series but also to interpret correctly the temporal data available in the dataset.

Two approaches are proposed in the literature:

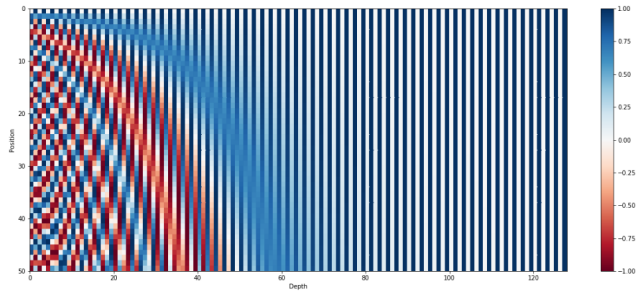
I. Fixed time embedding: define a global timestamp encoding for each combination of day/month/year. It's not a smooth presentation due to the discontinuity of the dummy variables;

II. Learnable time embedding: similar to the first approach but the variables are learned during the training of the model.

Encoding strategies: a recap

Vanilla Transformer:

- Fixed Positional encoding

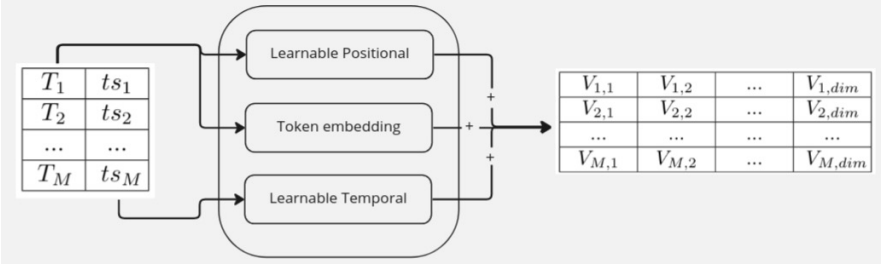


- Word encoding: from text to float vector

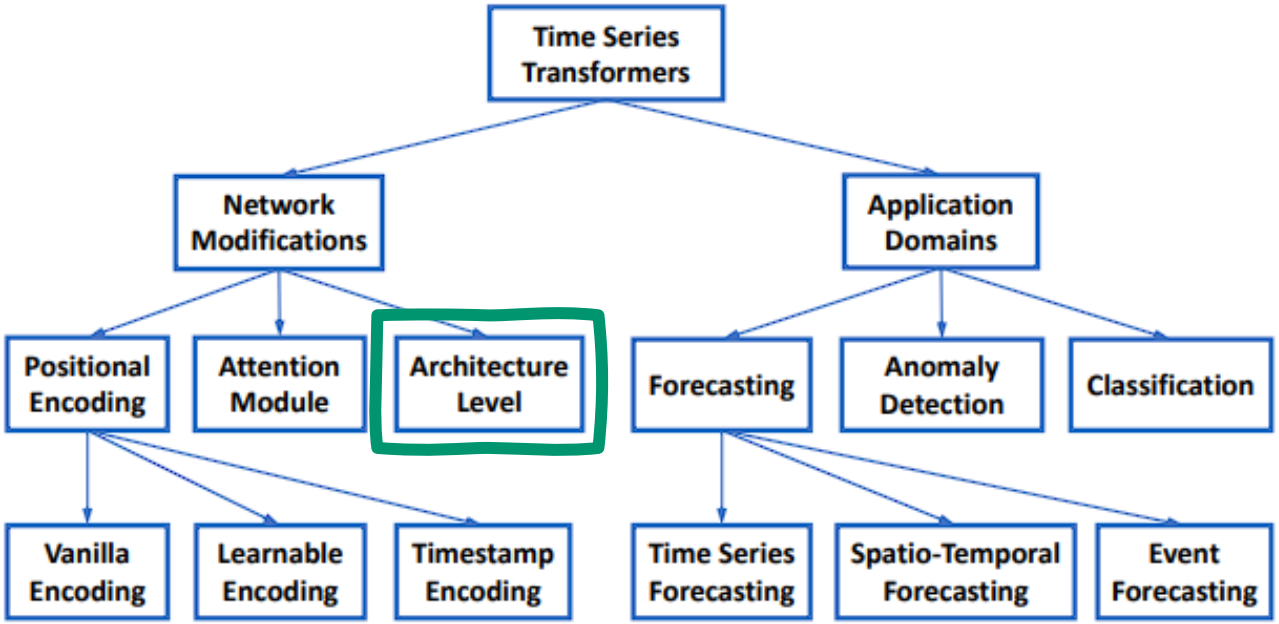
Time-Series Transformer:

- Learnable Positional encoding: from Conv layer to LSTM.
- Fixed Temporal encoding: fixed representation of the time (e.g., day, week, year, holiday, weekand).
- Learnable Temporal encoder: from Conv layer to LSTM.

Learnable encoding is more robust to learn lagged positional dependencies

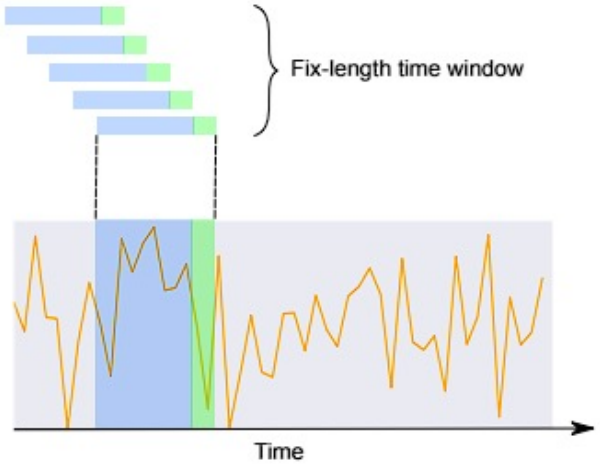


How to modify a transformers for time series?



How to handle window of data in Transformers?

The model's input is a window over the time series and it is composed by couples (***T***, ***ts***): ***T*** is the feature's value at a specific time, while ***ts*** is its corresponding timestamp needed in the Temporal Encoding.

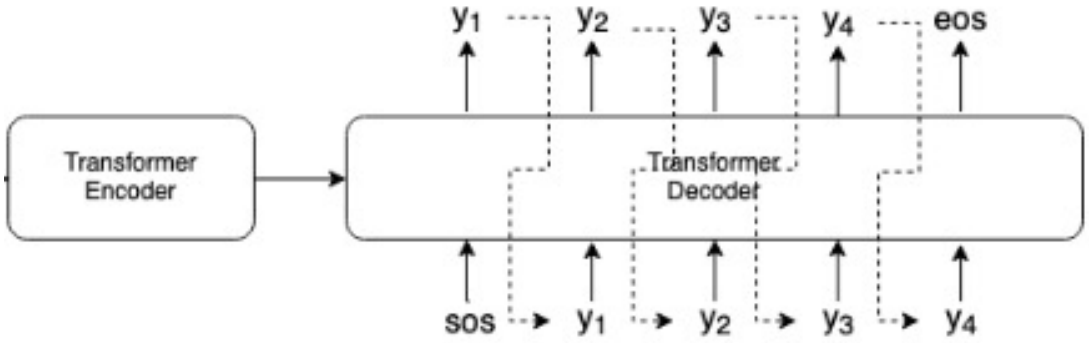


Window length

T_1	T_2	...	T_{w-1}	T_w
ts_1	ts_2	...	ts_{w-1}	ts_w

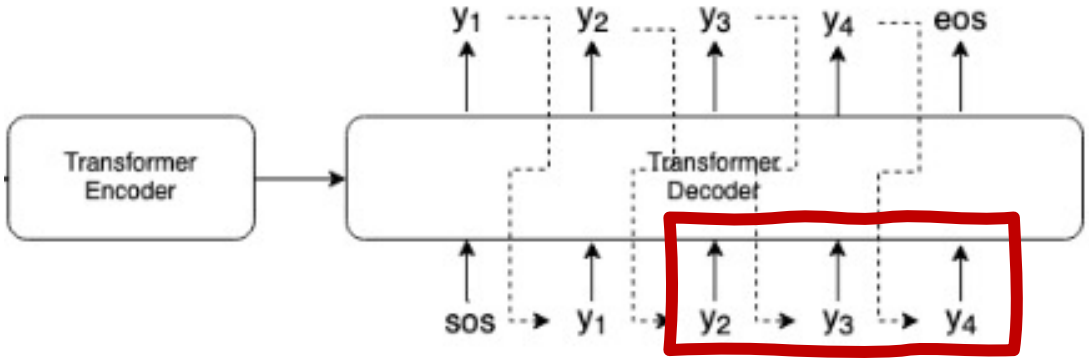
How to handle window of data in Transformers?

- In the literature, the most used approach is the one proposed in the original version of Transformer.
- The encoder takes samples collected from t-N to t-M where N>M to take in consideration the context.
- The inputs to the decoder is the output of the encoder and the previous outputs of decoder block itself.



The peculiarity of training models for time series!

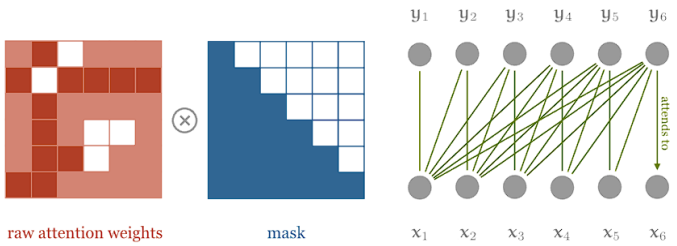
- To avoid the model to look at future values during the training phase, the first approach needs to implement a masking strategy.
- An alternative approach could be feed the decoder with samples that have been already collected. However, the performances of this approach still need to be properly validated.



So, how to deal with time series window of data in the training of Transformers?

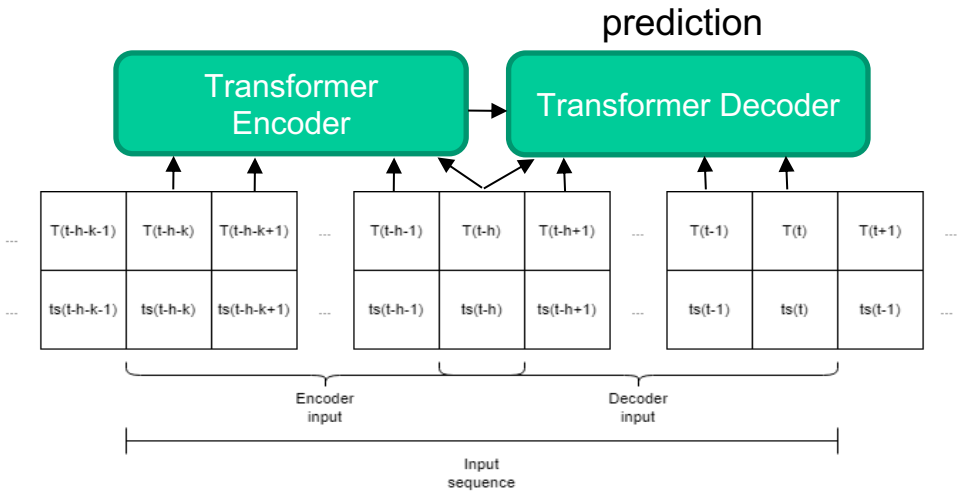
Vanilla Transformer:

- Masked attention mechanism in the decoder: it becames an autoregressive model

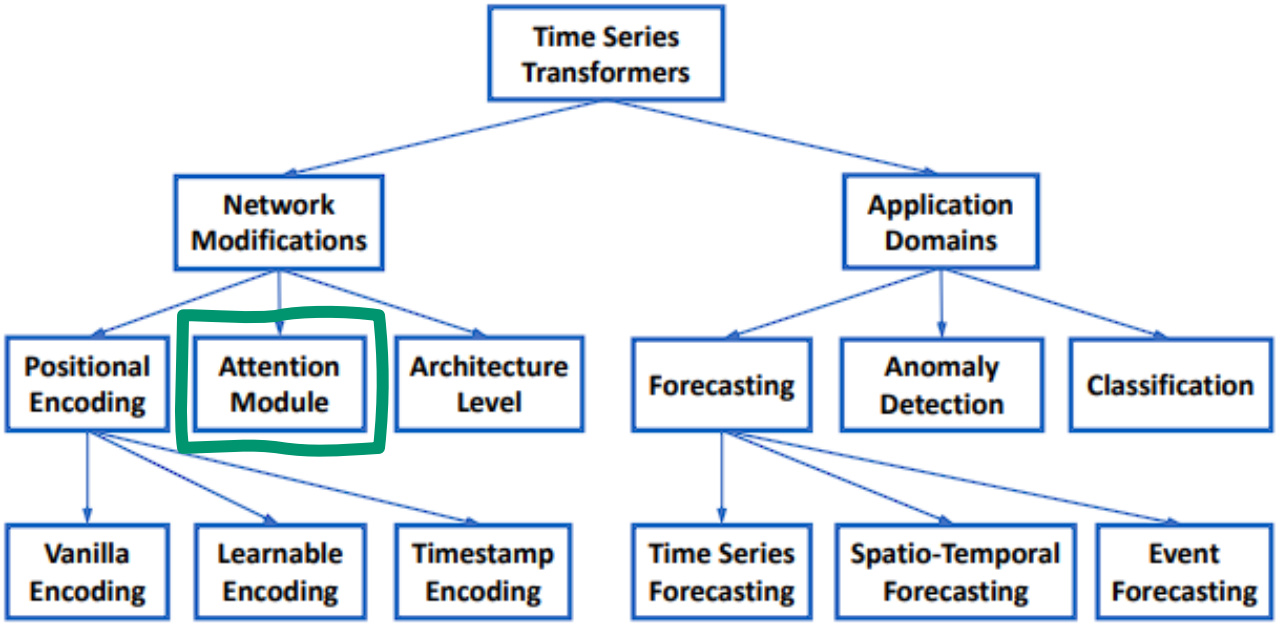


Time-Series Transformer:

- No need of an autoregressive behaviour. We can exploit the whole architecture and reorganize the input s.t. the decoder is fed with samples collected until t to predict t+1.

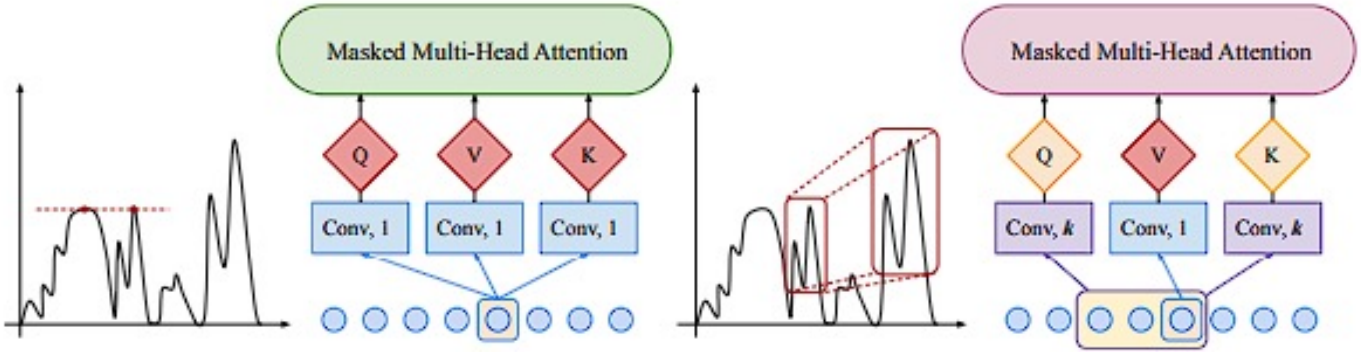


How to modify a transformers for time series?



The role of attention in time series …

- **Query-key matching agnostic of local context may confuse the self-attention module** in terms of whether the observed value is an anomaly, change point or part of patterns, and bring underlying optimization issues.
- Rather than using convolution of kernel size 1 with stride 1 (matrix multiplication), we can employ ***causal convolution*** of kernel size k with stride 1 to transform inputs (with proper paddings) into queries and keys to **be more aware of local context**.



Vanilla vs Casual Convolution attention

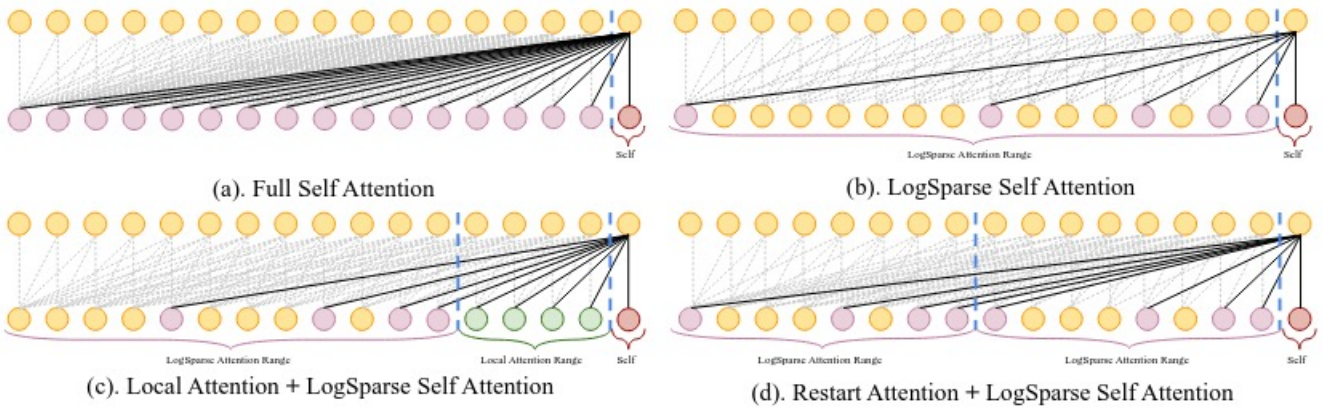
https://arxiv.org/pdf/1907.00235.pdf

Keeping long-term dependencies in the attention module for time series

The self-attention module in vanilla Transformer has **a time and memory complexity** directly proportional to the time series length: **a computational bottleneck** when dealing with long sequences.

A possible option is to use ***LogSparse Attention***:

- LogSparse Self Attention
- Local + LogSparse Self Attention
- Restart + LogSparse Self Attention



(a). Full Self Attention

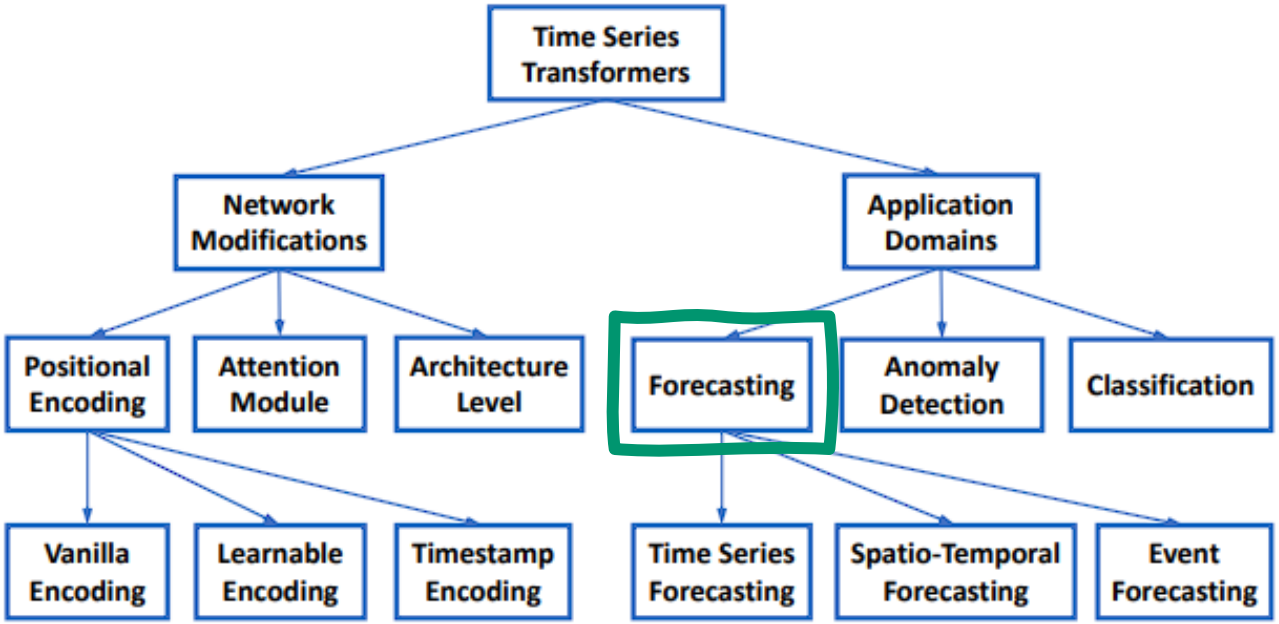
(b). LogSparse Self Attention

(c). Local Attention + LogSparse Self Attention

(d). Restart Attention + LogSparse Self Attention

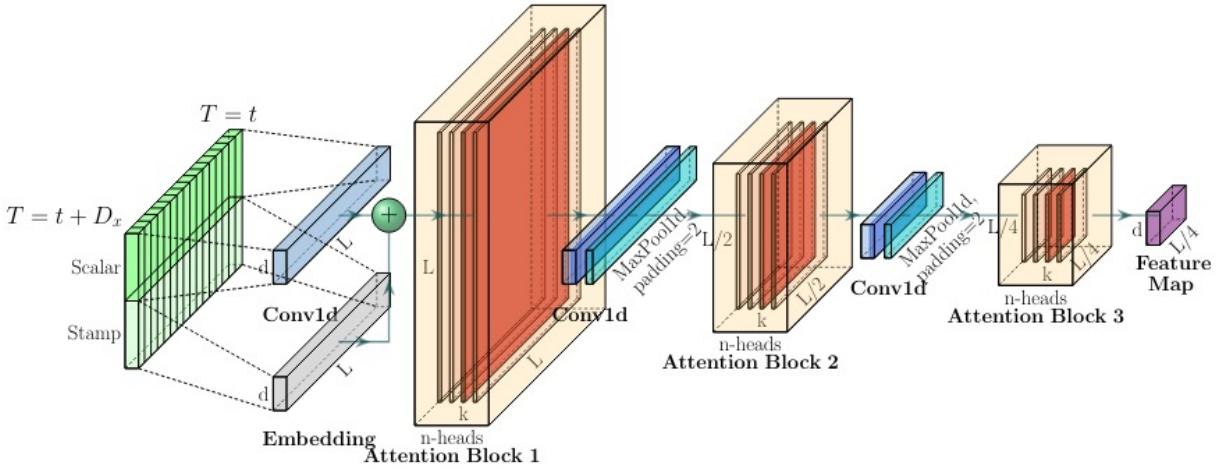
https://arxiv.org/pdf/1907.00235.pdf

How to modify a transformers for time series?



Transforming Transformers architectures for time series forecasting

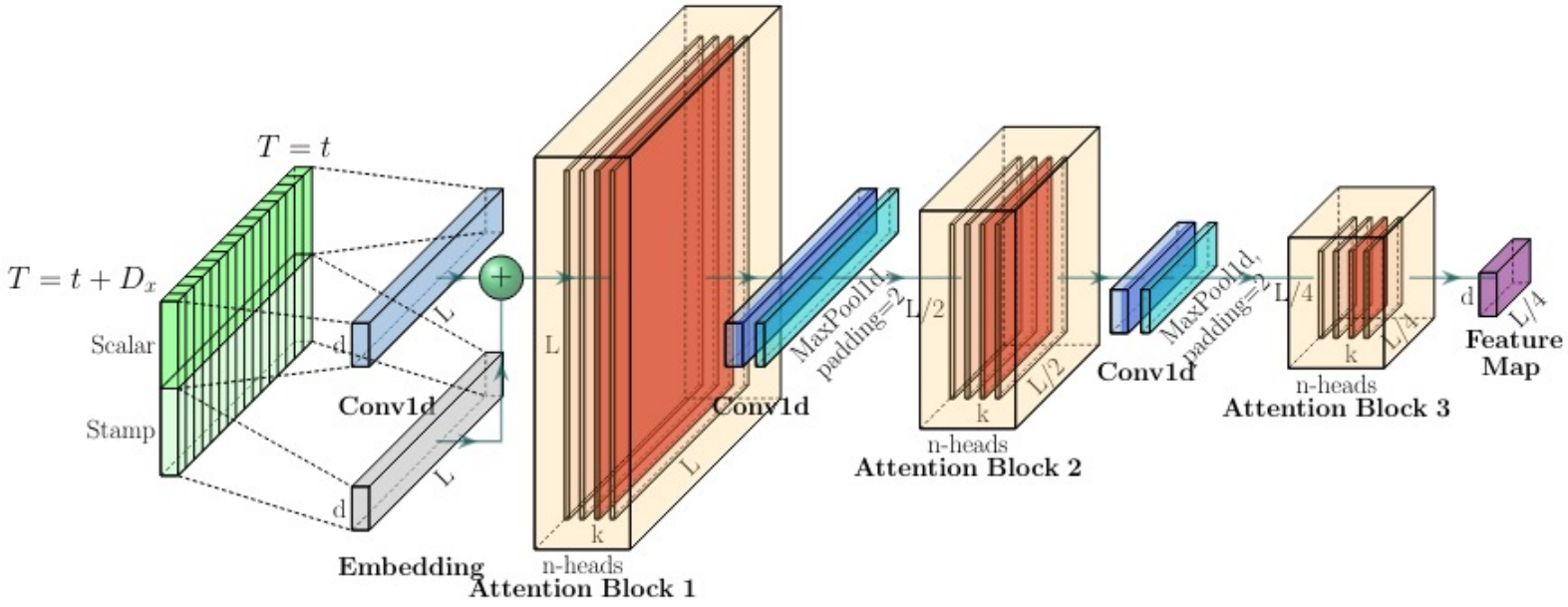
- Informers
- Pyraformer
- Temporal Fusion Transformer
- Adversarial Sparse Transformer
- Autoformer



https://arxiv.org/pdf/2012.07436.pdf

Informers

- Introducing hierarchical architecture into Transformer to take into account multi-resolution aspect of time series.
- Inserting max-pooling layers with stride 2 between attention blocks to down-sample the time series

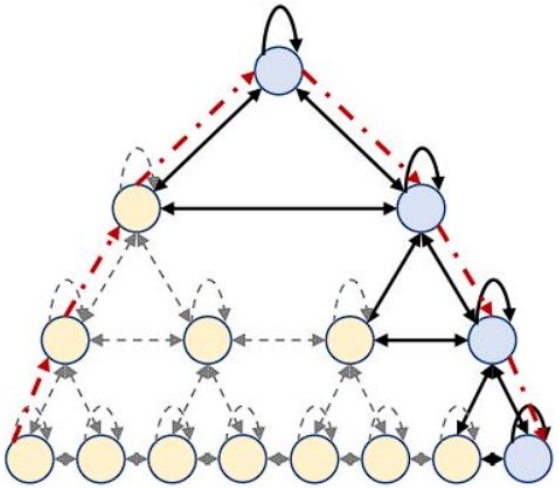


https://arxiv.org/pdf/2012.07436.pdf

Pyraformer

- A C-ary tree-based attention mechanism
- Leaf nodes at the finest scale correspond to the original time series
- Nodes in the coarser scales represent series at lower resolutions.

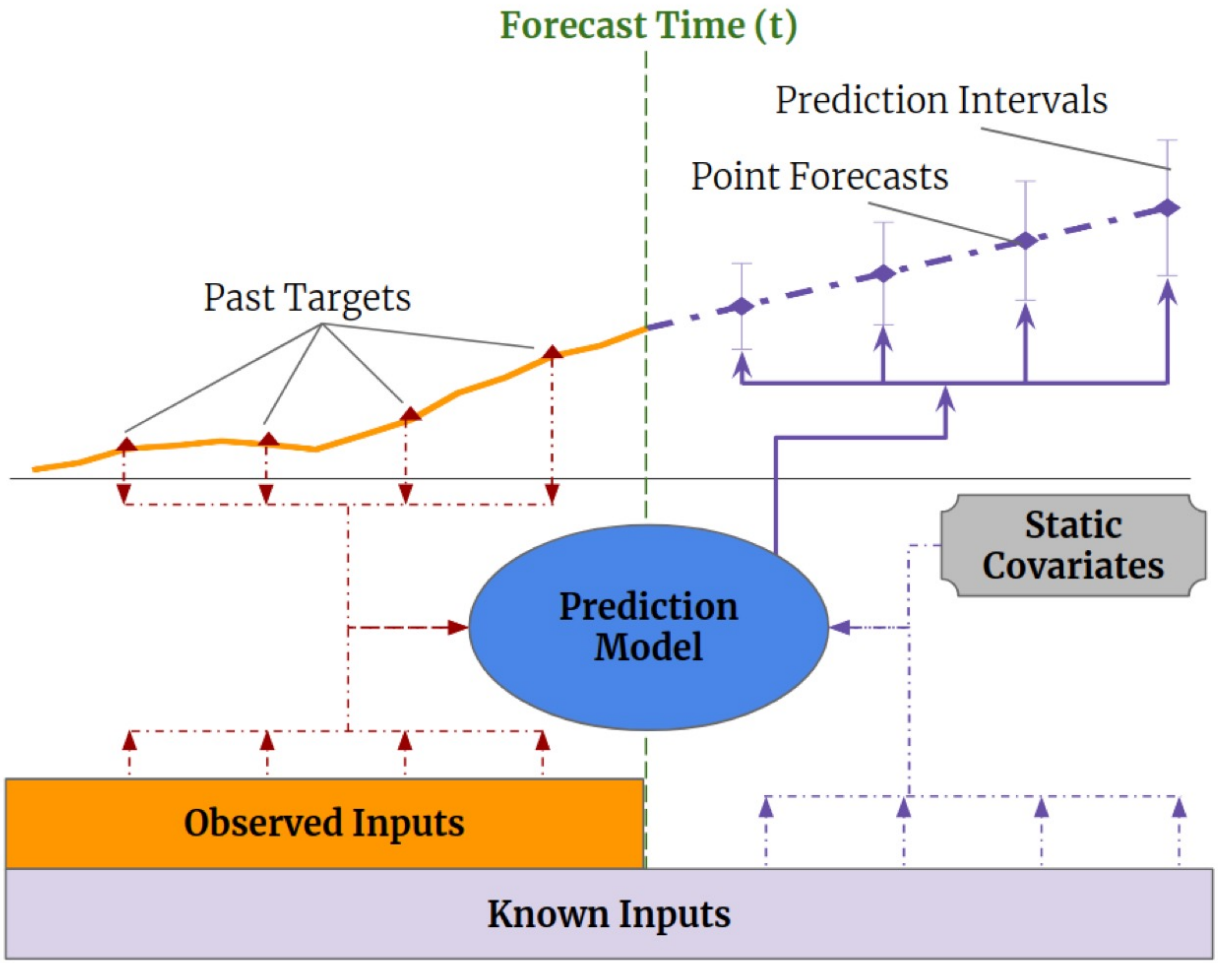
Pyraformer relies on both intra-scale and inter-scale attentions in order to better capture temporal dependencies across different resolutions



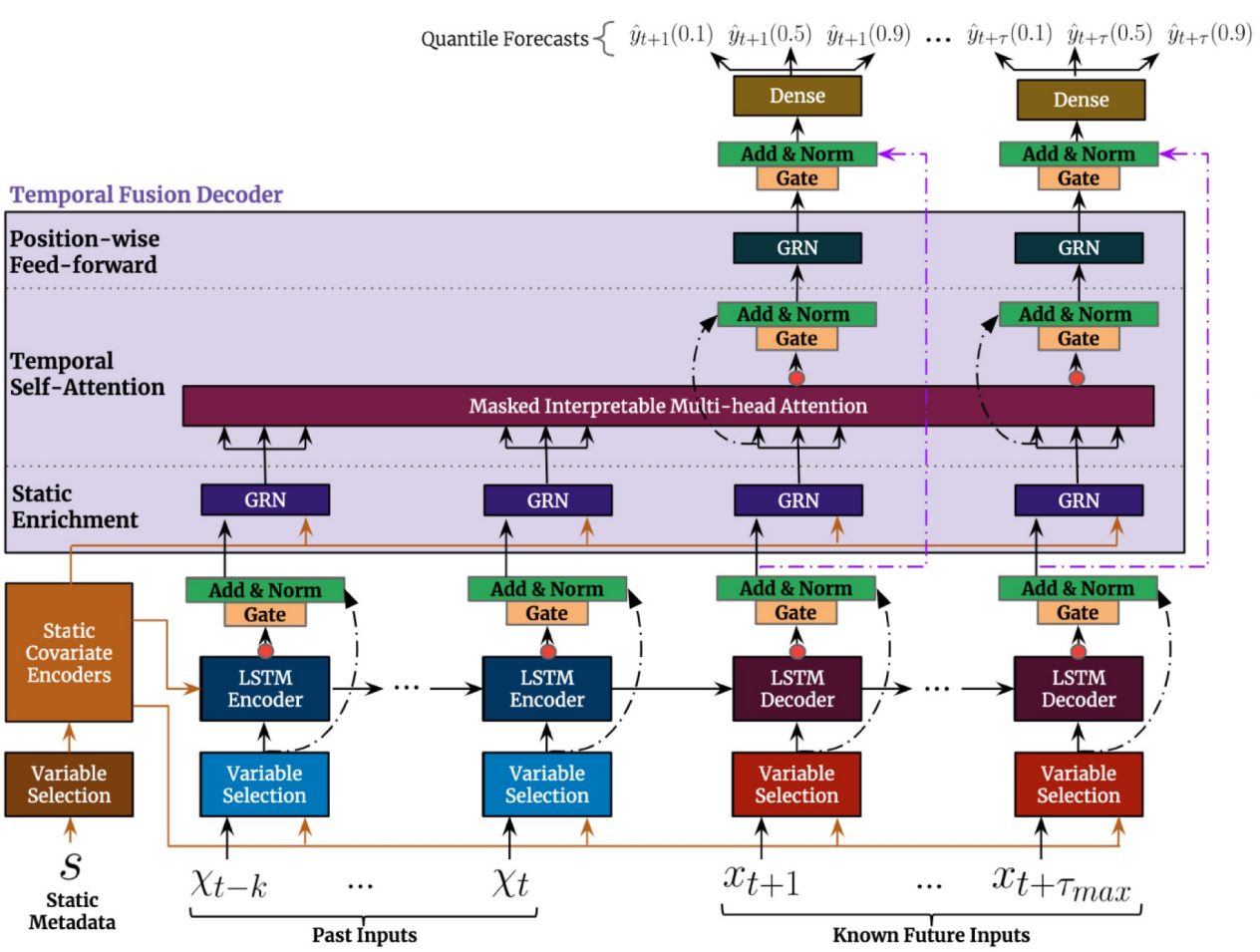
The Pyramidal Attention Mechanism.

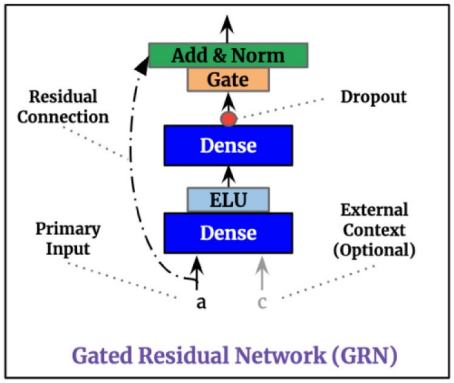
https://openreview.net/pdf?id=0EXmFzUn5I

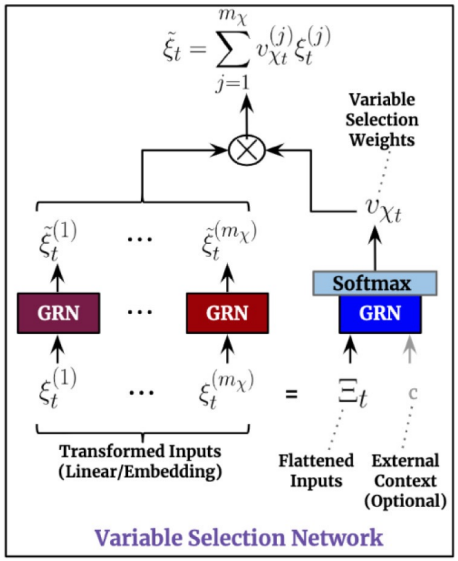
Temporal Fusion Transformer: the model



Temporal Fusion Transformer: the architecture (TFT)



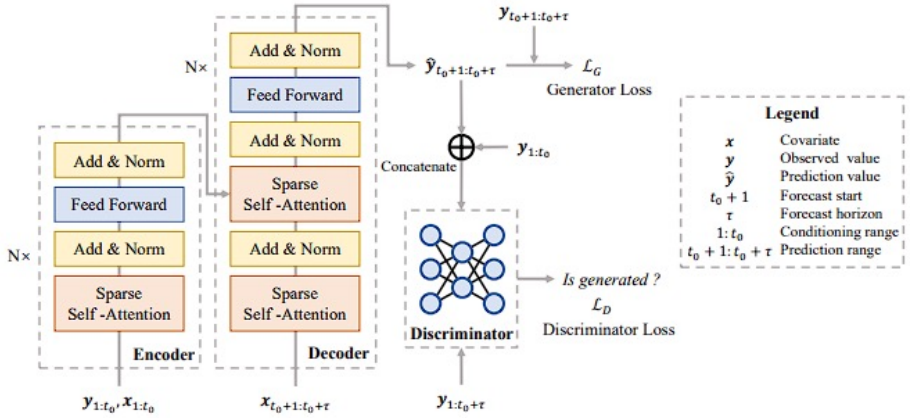




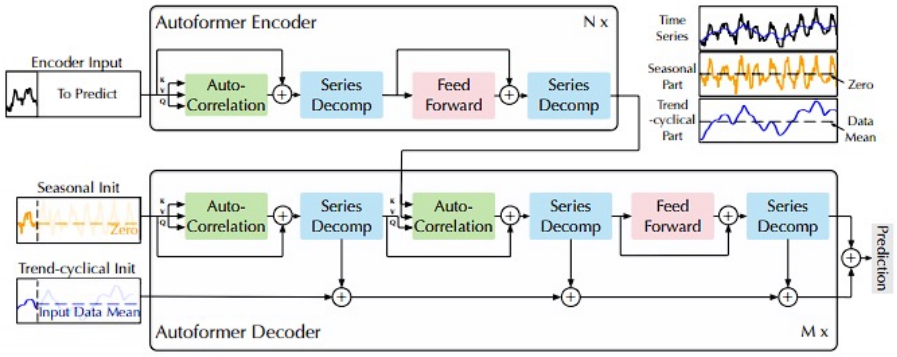
https://arxiv.org/pdf/1912.09363.pdf

Adversarial Sparse Transformer and Autoformer

- **Adversarial Sparse Transformer** uses a generative adversarial encoder-decoder framework. Adversarial training can improve the time series forecasting by directly shaping the output distribution of network to avoid the error accumulation through one-step ahead inference.

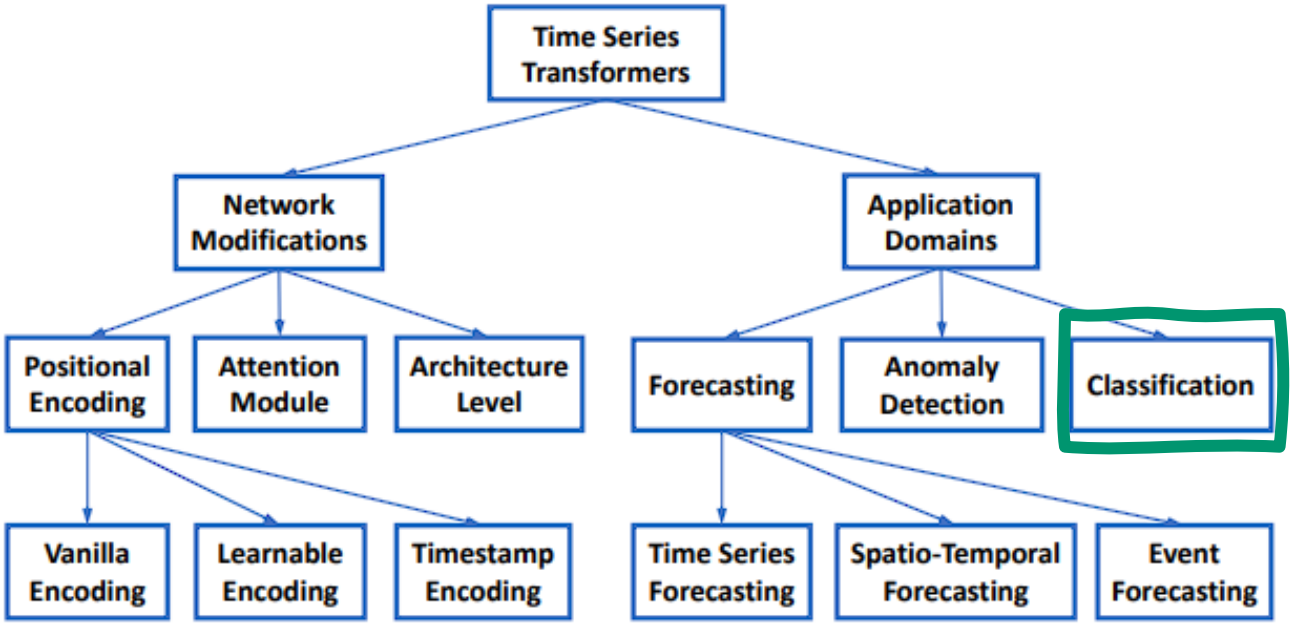


- **Autoformer** devises a simple seasonal-trend decomposition architecture with an auto-correlation mechanism working as an attention module: it measures the time-delay similarity between inputs signal and aggregate the top-k similar sub-series to forecast with a reduced complexity



https://proceedings.neurips.cc/paper/2020
https://proceedings.neurips.cc/paper/2021/autoformer.pdf

How to modify a transformers for time series?

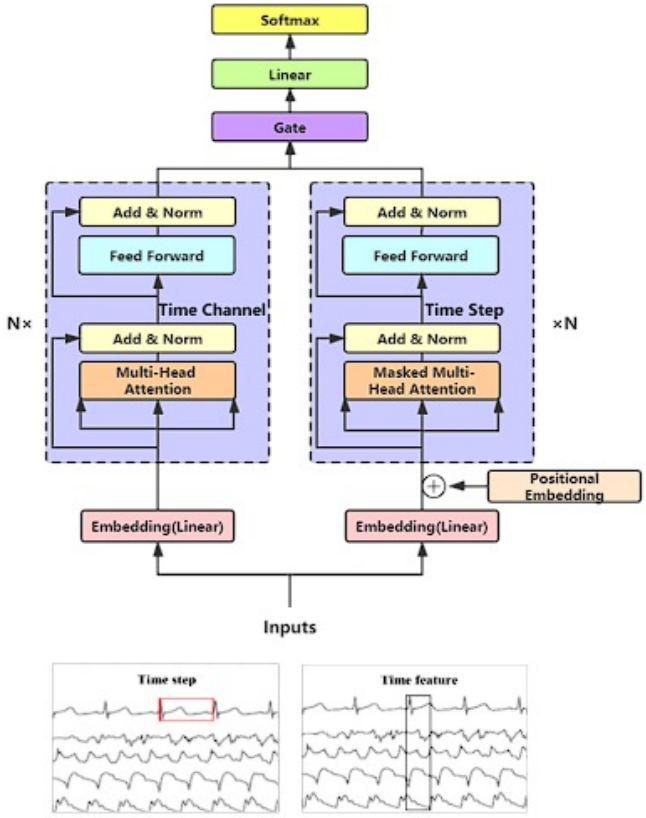


Transformers for time series classification

Classification Transformers usually employ a simple encoder structure, in which self-attention layers performs representation learning and feed forward layers produce probability of each class.

- Gated Transformer Network uses a two-tower Transformer with each tower respectively working on time-step-wise attention and channel-wise attention. To merge the feature of the two towers, a learnable weighted concatenation layer is used.

- Transformer for raw optical satellite image time series classification has been introduced. The solution relies on a self-supervised pre-trained schema because of limited labeled data.



Gated Transformer Network

https://arxiv.org/pdf/2103.14438.pdf
https://ieeexplore.ieee.org/satellite_classification

Is transfer learning useful in time series forecasting/classification (with Transformers)?

Advantages:

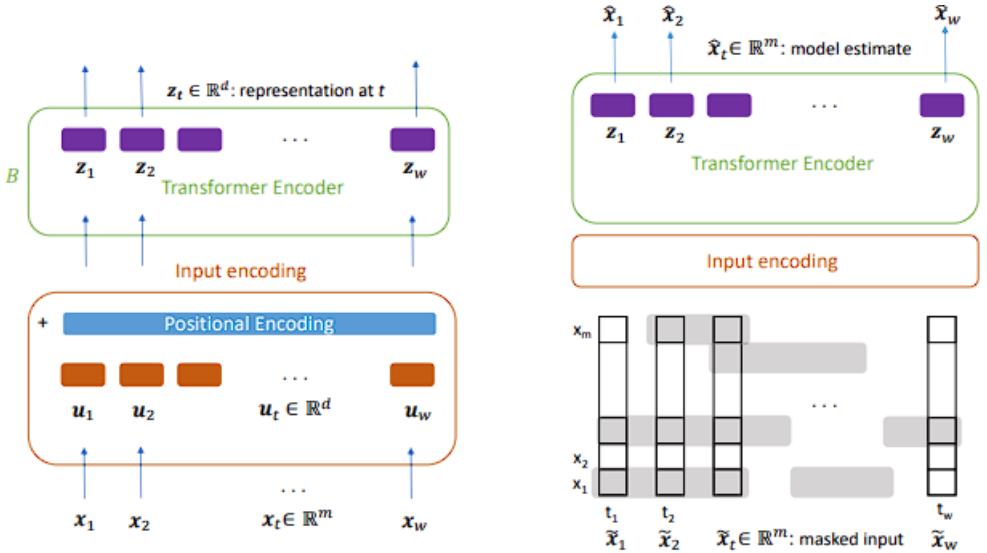
- better performances on the target dataset if its size is very small
- faster training

Disadvantages:

- the similarity measure between the source and target dataset must evidence a strong relationship between the time series otherwise the use of TL leads to bad results
- in time series it is not always trivial to choose the best similarity measure that correctly interprets the relationships between two time series

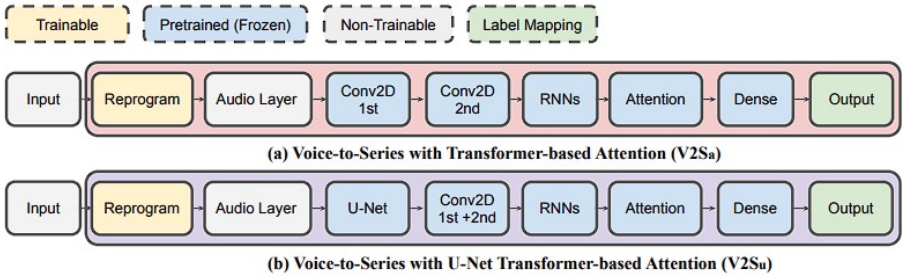
Transfer learning for Transformers (for classification)

- An unsupervised pre-trained framework and the model is pre-trained with proportionally masked data. The pre-trained models are then fine-tuned in downstream tasks such as classification.



Left: standard architecture. Right: unsupervised pretraining task

- A large-scale pre-trained speech processing model for downstream time series classification problems.



https://dl.acm.org/doi/pdf/zerveas
https://arxiv.org/pdf/2106.09296.pdf

Some comments

The Transformer, originally applied to NLP task, is a model that c**an potentially find its application also in the time series** domain

However, it is worthy to **highlight critical issues**:

- The architecture of Transformers does not fit well with time series
- The domain in NLP, in which Transformers reach state of the art performances, is large but still finite. Their training procedure adopt a huge number of samples (billions) that are not usually available in time series problems;
- Some works evidence that the most sophisticated Transformer-based models can be outperformed by very simple network.

The (motivated) fall of Transformers

Methods	Metric	Electricity 96	192	336	720	Exchange-Rate 96	192	336	720	Traffic 96	192	336	720	Weather 96	192	336	720	ILI 24	36	48	60
DLinear-S*	MSE	0.194	0.193	0.206	0.242	**0.078**	0.159	0.274	**0.558**	0.650	**0.598**	**0.605**	0.645	0.196	0.237	0.283	0.345	**2.398**	**2.646**	2.614	2.804
	MAE	0.276	0.280	0.296	0.329	0.197	0.292	0.391	**0.574**	0.396	**0.370**	0.373	0.394	0.255	0.296	0.335	0.381	**1.040**	1.088	1.086	1.146
DLinear-I*	MSE	**0.184**	**0.184**	**0.197**	**0.234**	0.084	**0.157**	**0.236**	0.626	0.647	0.602	0.607	0.646	**0.164**	**0.209**	**0.263**	**0.338**	3.015	2.737	**2.577**	2.821
	MAE	**0.270**	**0.273**	**0.289**	**0.323**	0.216	0.298	**0.379**	0.634	0.403	0.375	0.377	0.398	**0.237**	**0.282**	**0.327**	**0.380**	1.192	**1.036**	1.043	**1.091**
FEDformer	MSE	0.193	0.201	0.214	0.246	0.148	0.271	0.460	1.195	**0.587**	0.604	0.621	**0.626**	0.217	0.276	0.339	0.403	3.228	2.679	2.622	2.857
	MAE	0.308	0.315	0.329	0.355	0.278	0.380	0.500	0.841	**0.366**	0.373	0.383	**0.382**	0.296	0.336	0.380	0.428	1.260	1.080	1.078	1.157
Autoformer	MSE	0.201	0.222	0.231	0.254	0.197	0.300	0.509	1.447	0.613	0.616	0.622	0.660	0.266	0.307	0.359	0.419	3.483	3.103	2.669	**2.770**
	MAE	0.317	0.334	0.338	0.361	0.323	0.369	0.524	0.941	0.388	0.382	**0.337**	0.408	0.336	0.367	0.395	0.428	1.287	1.148	1.085	1.125
Informer	MSE	0.274	0.296	0.300	0.373	0.847	1.204	1.672	2.478	0.719	0.696	0.777	0.864	0.300	0.598	0.578	1.059	5.764	4.755	4.763	5.264
	MAE	0.368	0.386	0.394	0.439	0.752	0.895	1.036	1.310	0.391	0.379	0.420	0.472	0.384	0.544	0.523	0.741	1.677	1.467	1.469	1.564
Pyraformer*	MSE	0.386	0.378	0.376	0.376	1.748	1.874	1.943	2.085	0.867	0.869	0.881	0.896	0.622	0.739	1.004	1.420	7.394	7.551	7.662	7.931
	MAE	0.449	0.443	0.443	0.445	1.105	1.151	1.172	1.206	0.468	0.467	0.469	0.473	0.556	0.624	0.753	0.934	2.012	2.031	2.057	2.100
LogTrans	MSE	0.258	0.266	0.280	0.283	0.968	1.040	1.659	1.941	0.684	0.685	0.734	0.717	0.458	0.658	0.797	0.869	4.480	4.799	4.800	5.278
	MAE	0.357	0.368	0.380	0.376	0.812	0.851	1.081	1.127	0.384	0.390	0.408	0.396	0.490	0.589	0.652	0.675	1.444	1.467	1.468	1.560
Reformer	MSE	0.312	0.348	0.350	0.340	1.065	1.188	1.357	1.510	0.732	0.733	0.742	0.755	0.689	0.752	0.639	1.130	4.400	4.783	4.832	4.882
	MAE	0.402	0.433	0.433	0.420	0.829	0.906	0.976	1.016	0.423	0.420	0.420	0.423	0.596	0.638	0.596	0.792	1.382	1.448	1.465	1.483
Repeat-C*	MSE	1.588	1.595	1.617	1.647	0.081	0.167	0.305	0.823	2.723	2.756	2.791	2.811	0.259	0.309	0.377	0.465	6.587	7.130	6.575	5.893
	MAE	0.946	0.950	0.961	0.975	**0.196**	**0.289**	0.396	0.681	1.079	1.087	1.095	1.097	0.254	0.292	0.338	0.394	1.701	1.884	1.798	1.677

Linear

Transformers

- Methods* are implemented by us; Other results are from FEDformer [29].

Method	MACs	Parameter	Time	Memory
DLinear	**0.04G**	**139.7K**	**0.4ms**	**687MiB**
Transformer×	4.03G	13.61M	26.8ms	6091MiB
Informer	3.93G	14.39M	49.3ms	3869MiB
Autoformer	4.41G	14.91M	164.1ms	7607MiB
Pyraformer	0.80G	241.4M*	3.4ms	7017MiB
FEDformer	4.41G	20.68M	40.5ms	4143MiB

Zeng, A., Chen, M., Zhang, L., & Xu, Q. (2022). Are Transformers Effective for Time Series Forecasting?. *arXiv preprint arXiv:2205.13504*.

Put hands on with …

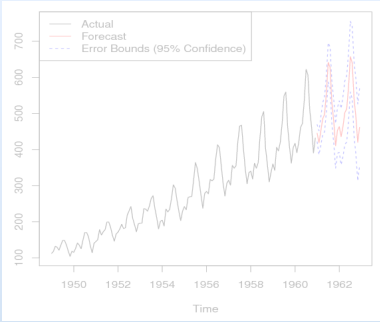
Alessandro Falcetta

Diego Riva

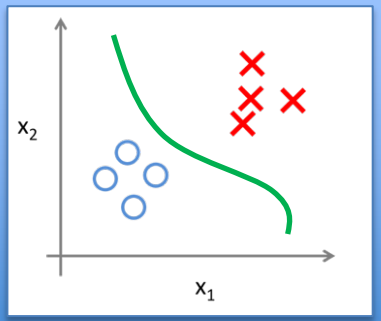
Appendix

Supervised learning: time series classification

Prediction

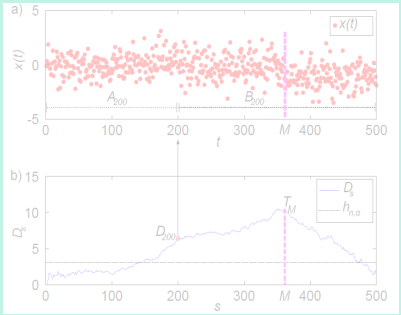


Classification

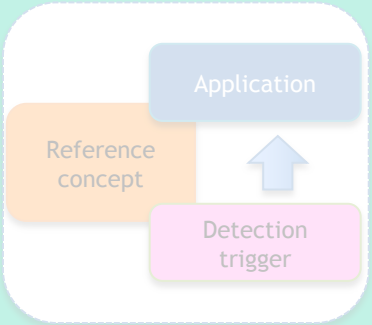


Clustering

Change Detection



Adaptation



Time series classification

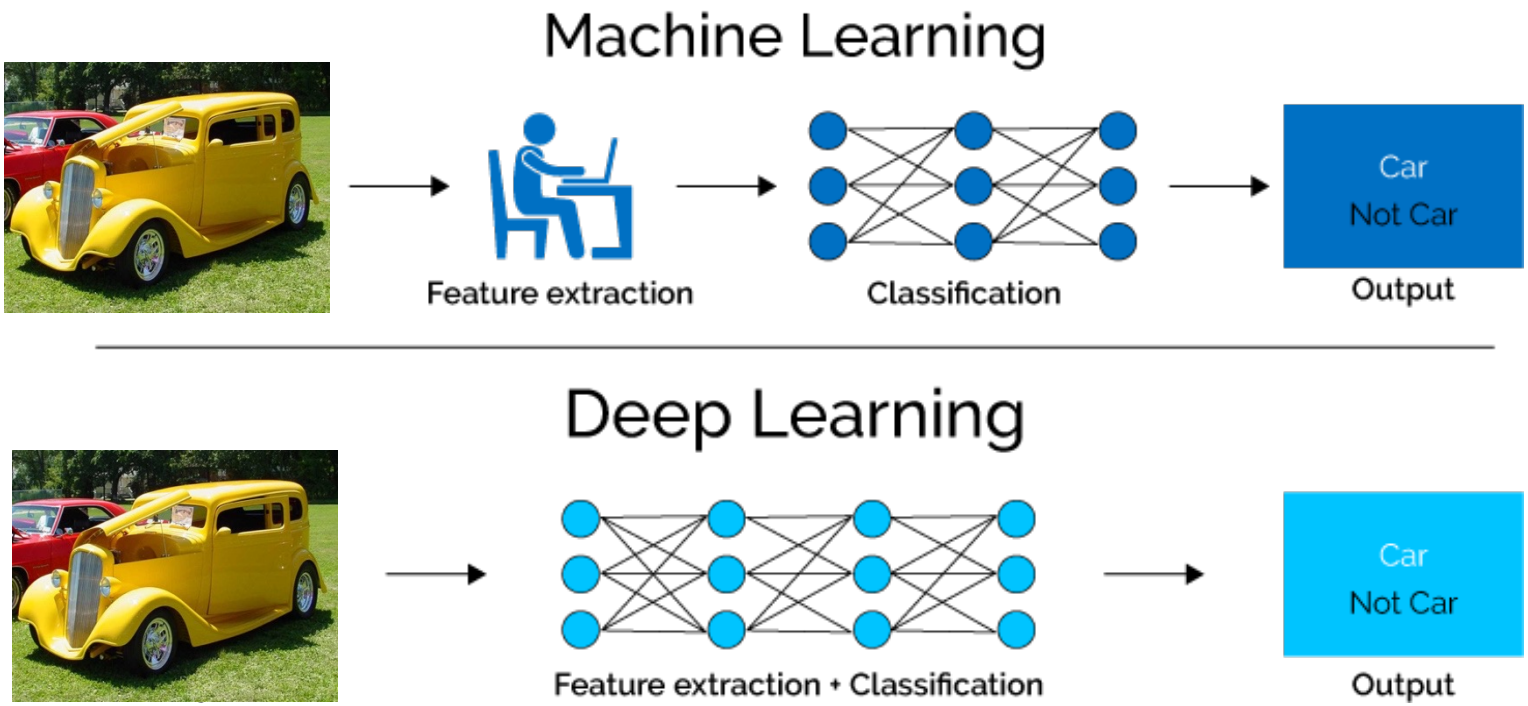
$$Y = f_{\theta}(X)$$

- The output classification, i.e., the class of the time series as output from the model.
- The cardinality of the class set is finite.

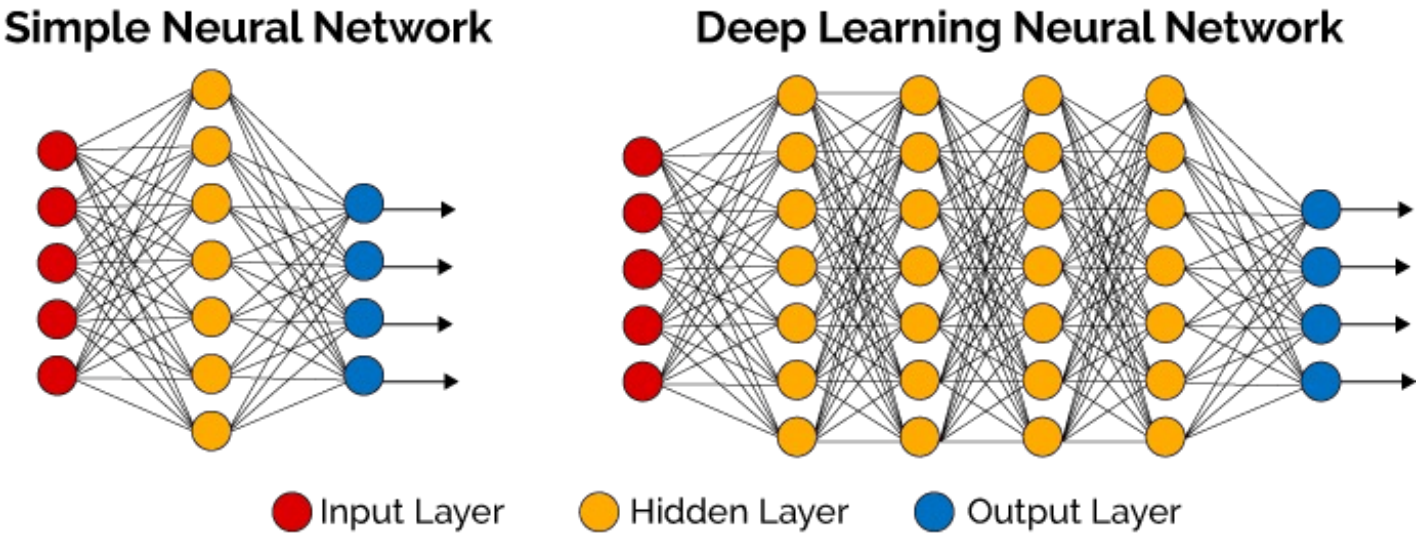
- The model receiving in input features extracted from the time series and providing in output the classification

- X can be features extracted from time series (machine learning) or directly the time series itself (deep learning)

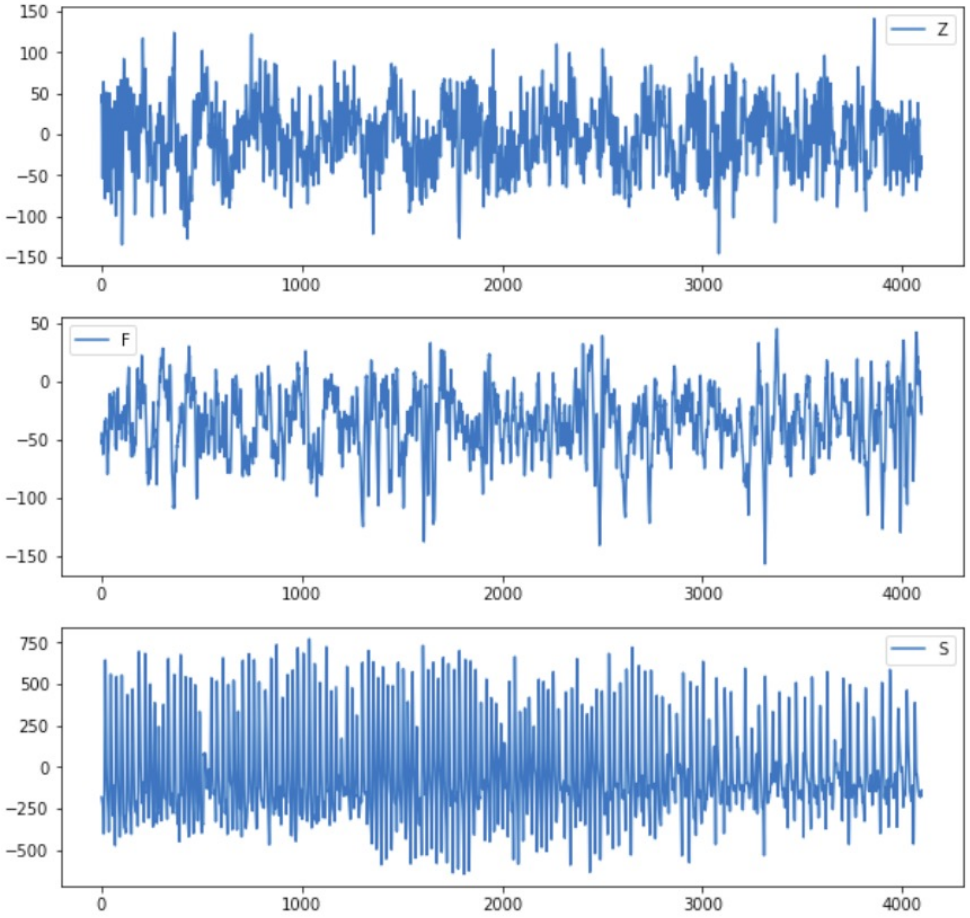
Machine vs deep learning



From shallow to deep learning



Extracting features from Time Series: a visual example



- Class Z and G seem to have less **skewed** data than class S.
- Each class has quite a different range of values, as we can see by inspecting the y-axis: **amplitude** feature could be useful.
- Not just the overall amplitude but the **overall distributi**on of the points that seems characteristically different

```
## python
>>> from cesium import featurize.featurize_time_series as ft
>>> features_to_use = ["amplitude",
>>>                    "percent_beyond_1_std",
>>>                    "percent_close_to_median",
>>>                    "skew",
>>>                    "max_slope"]
```

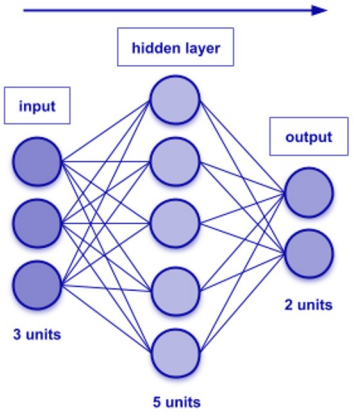
Using off-the-shelf libraries to compute features

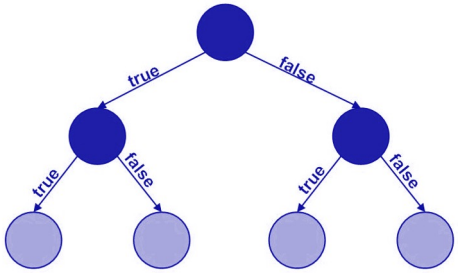
- **Python tsfresh** (794 time series features):
  - Energy, trend,
  - Autocorrelation, quantiles,
  - Absolute min, max, etc..
  - AR coefficients, etc

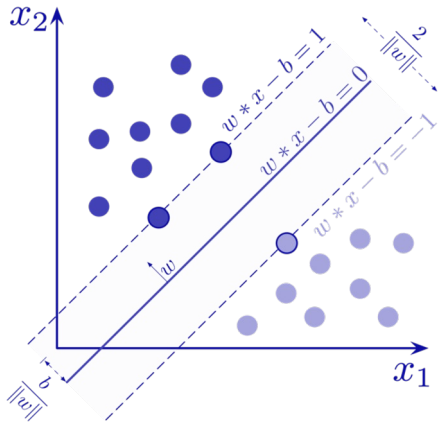
- **Python cesium** («large library of features»):
  - Amplitude,
  - Min, Max, std,
  - Peak distance,
  - Slope, etc…

How to process features?

- We need a classifier:
  - Feedforward Neural Networks
  - Decision trees
  - Support vector machines





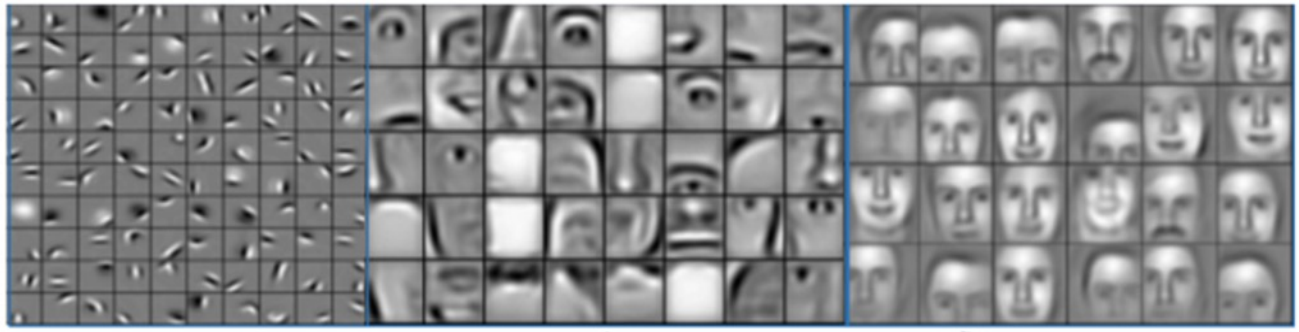


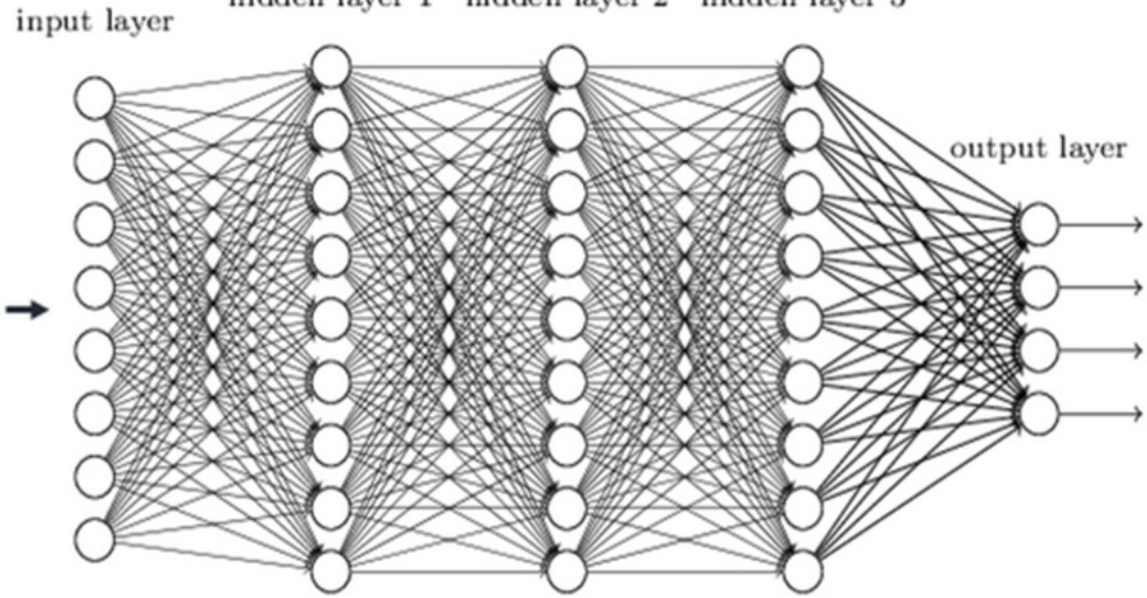
What about deep learning?

- Convolutional Neural Networks
- Recurrent Neural Networks (LSTM, ESN, etc..)

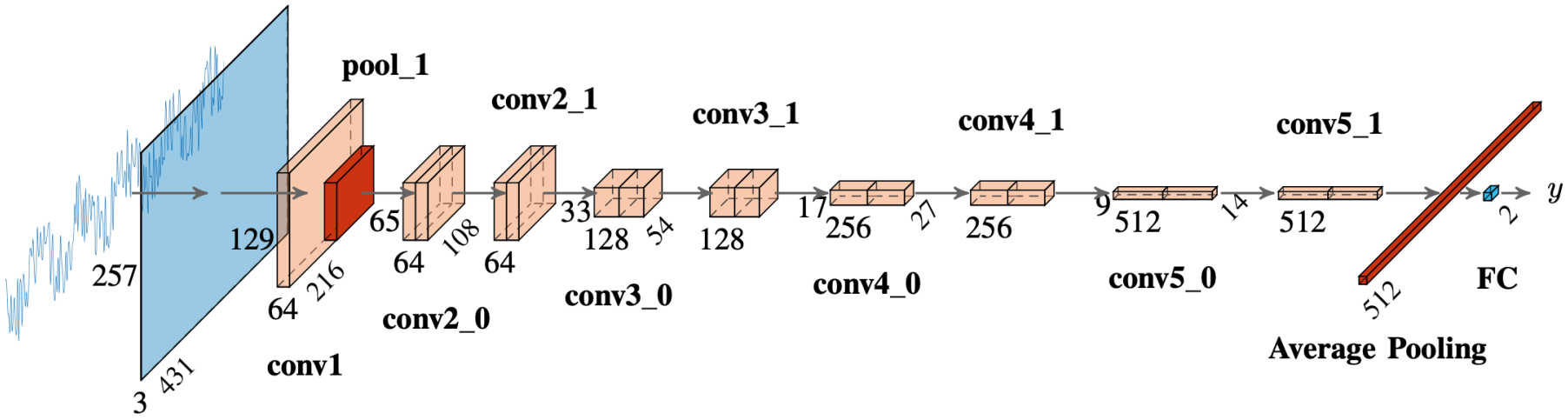
Convolutional Neural Networks (for images)

Deep neural networks learn hierarchical feature representations

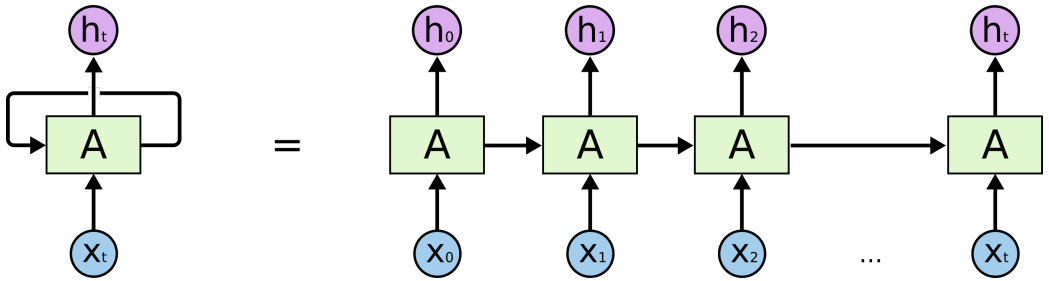


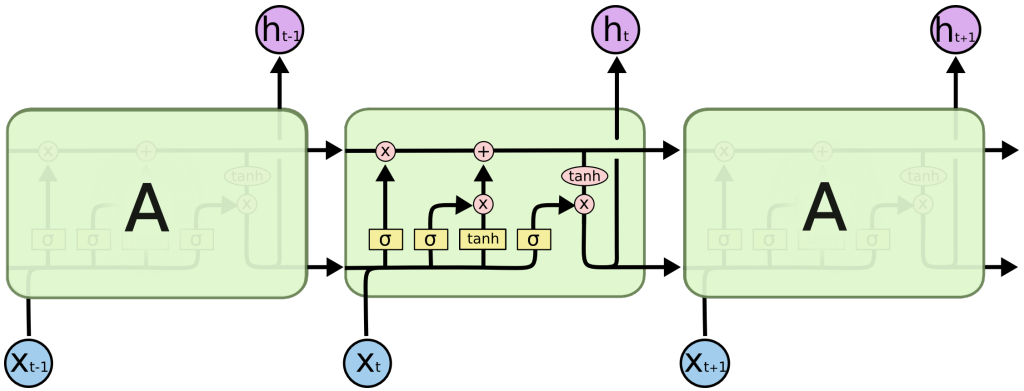


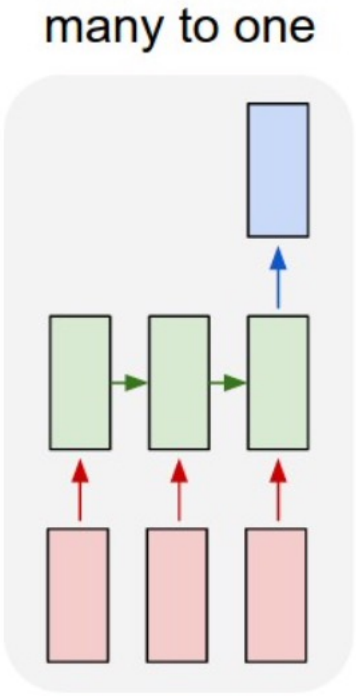
Convolutional Neural Networks (for time series)



Recurrent neural networks: RNN and LSTM



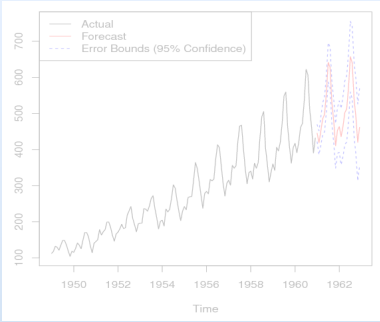




learnings

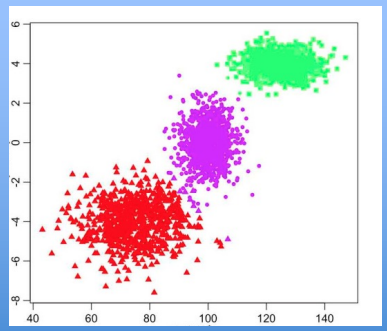
Supervised learning: time series clustering

Prediction

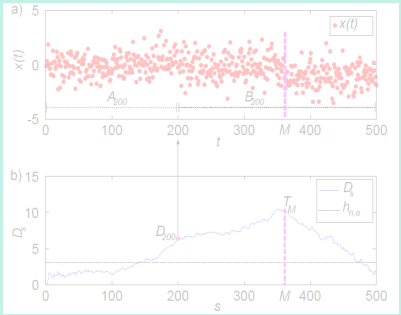


Classification

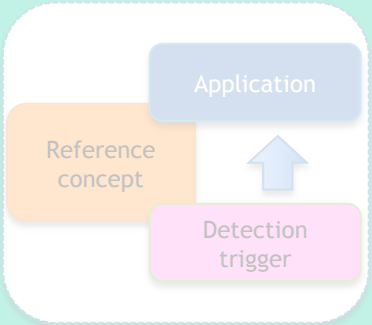
Clustering



Change Detection



Adaptation

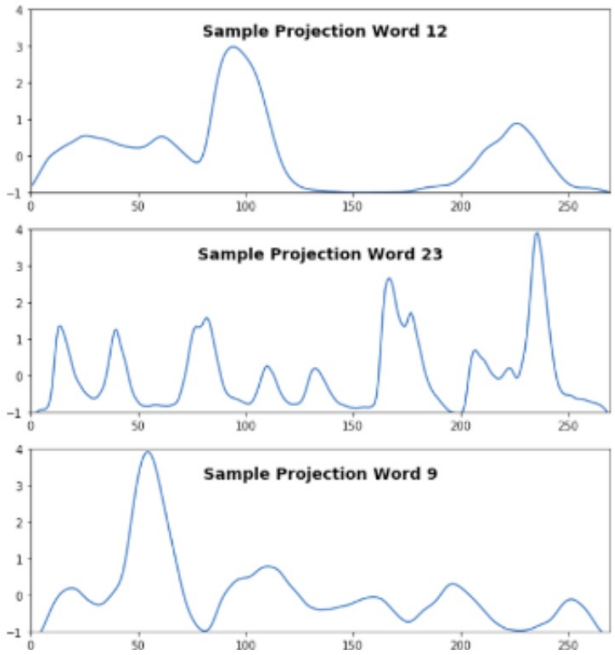


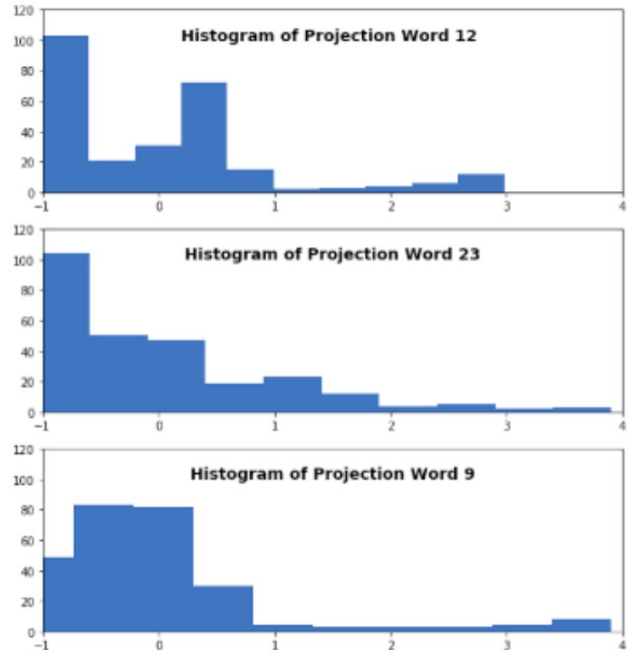
Clustering

- The general idea of clustering is that **data points that are similar to one another** constitute **meaningful groups** for purposes of analysis.
- This idea holds just as true for time series data as for other kinds of data.
- **Two classes of distance-metric options:**
  - ✓ *Distance based on features*
  - ✓ *Distance based on the raw time series data*

Clustering: *Distance based on features*

- Generate features for the time series
- Use these features as the coordinates for our time series
- This does not fully solve the problem of choosing a distance metric but …. it reduces the problem to the same distance metric problem posed by any cross-sectional data set.



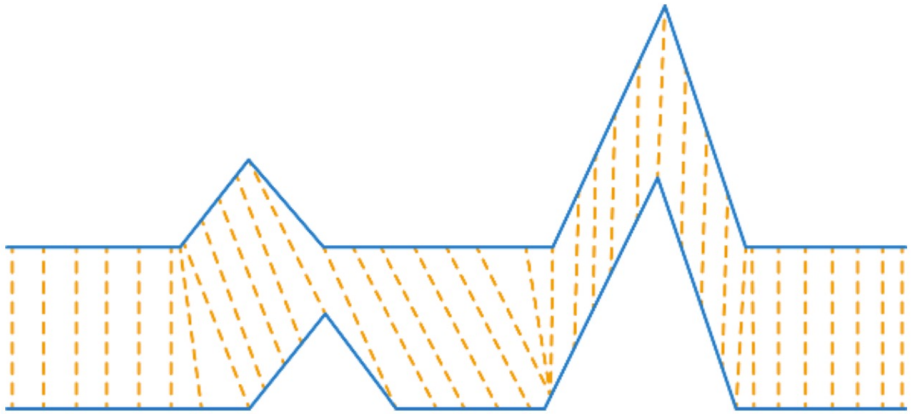


Features to be used

Clustering: *Distance based on the raw time series data*

- **Determine how "close"** different time series are
- They can handle:
  - different temporal scales,
  - different numbers of measurements,
  - other disparities between time series samples.
- **Dynamic Time Warping (DTW):**
  - Clustering a time series whose most salient feature is its overall shap
  - Relies on temporal "warping" to align time series along their temporal axis so as to compare their shapes
- **Other distance-based figures of merit:**
  - Pearson correlation, Longest common subsequence
- **Avoid Euclidean Distance Measures for Time Series!!!**

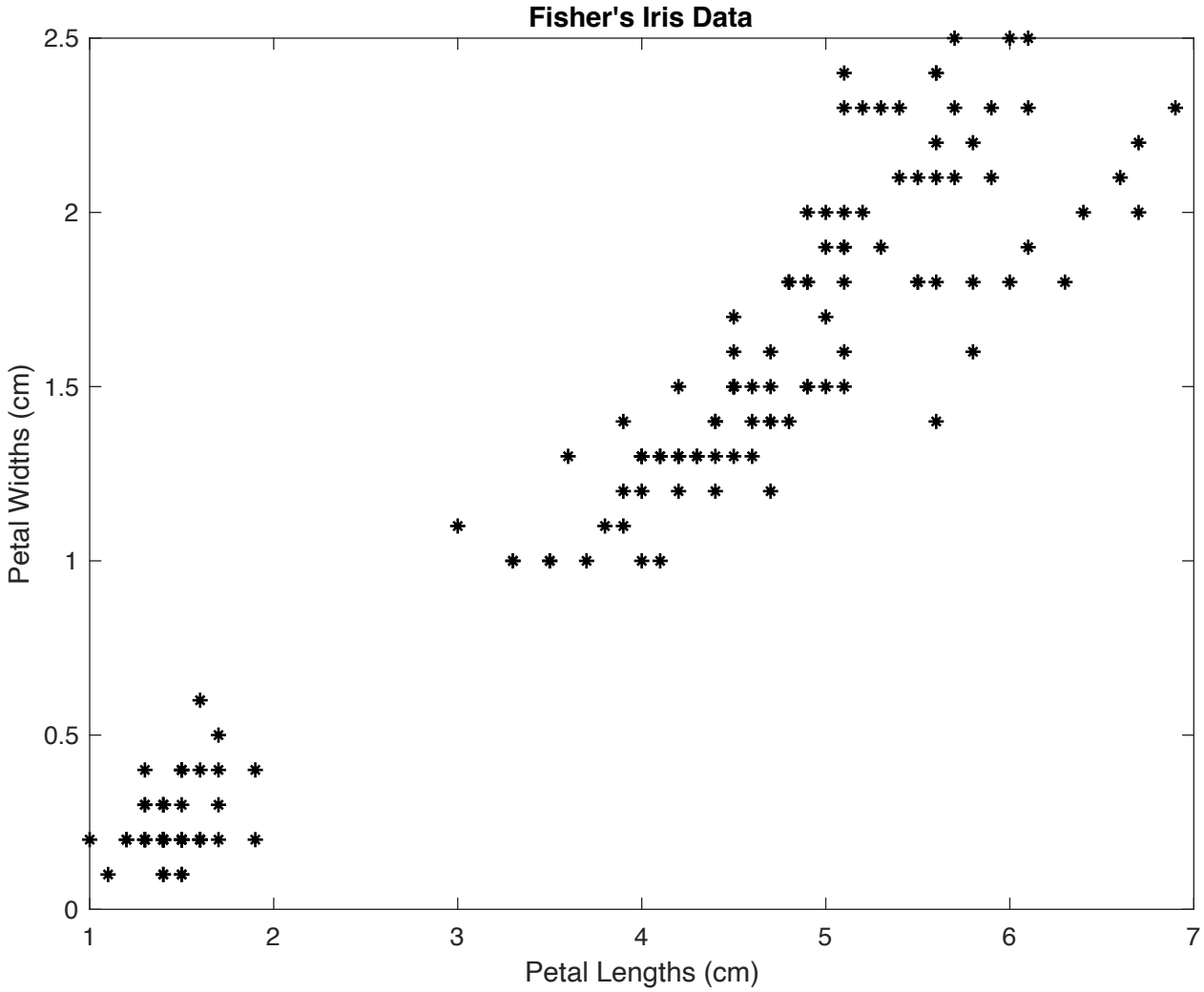
Dynamic Time Warping (DTW)



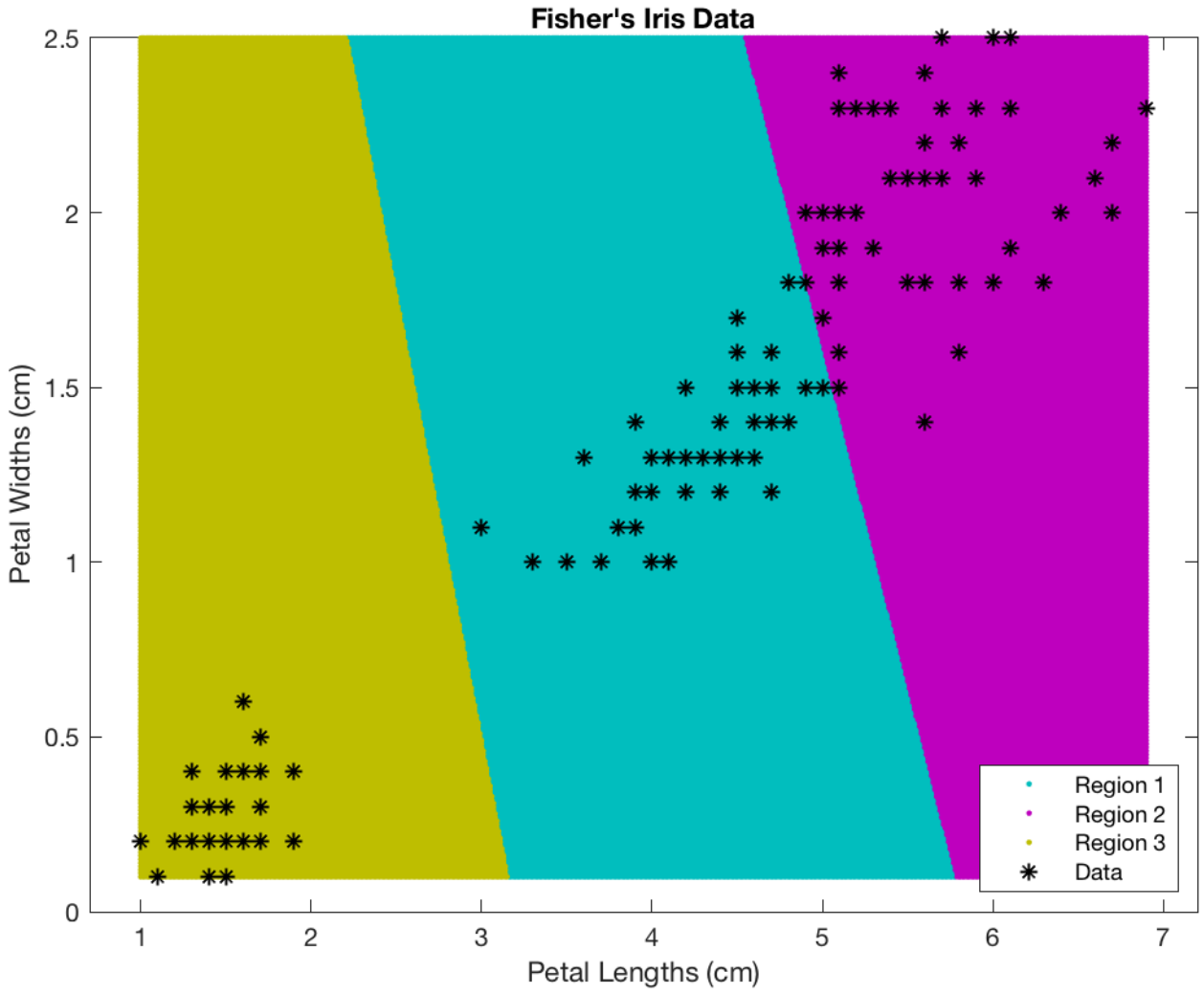
Comparing one time series that is measured in nanoseconds to another that is measured in millennia: comparing the visual “shape” rather than how much time is passing

- Many ways for temporal alignment
- DTW selectes **the one that minimizes the distance between the curves.**
- This distance, or cost function, is often measured as t**he sum of absolute differences between matched points**

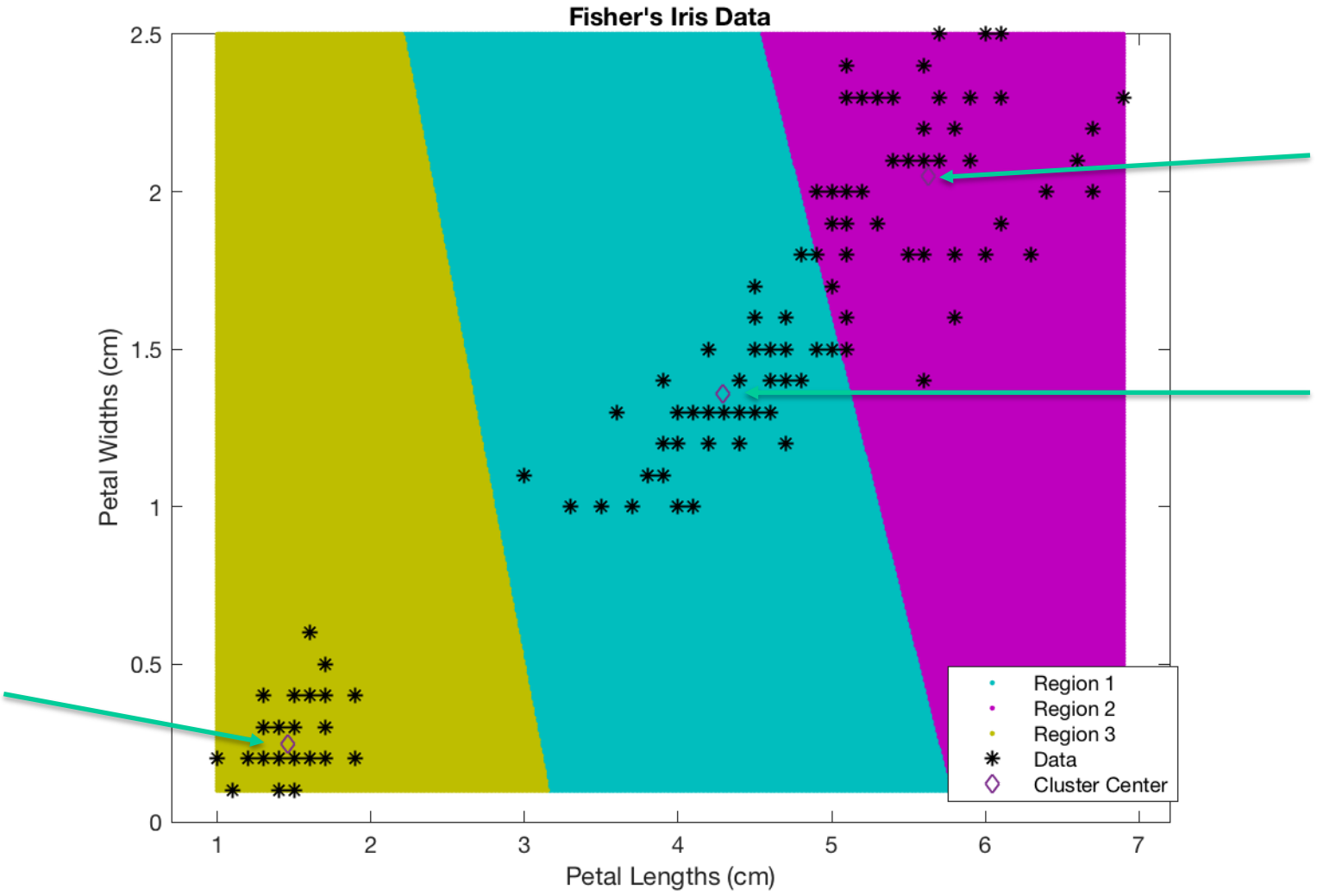
K-means clustering algorithm



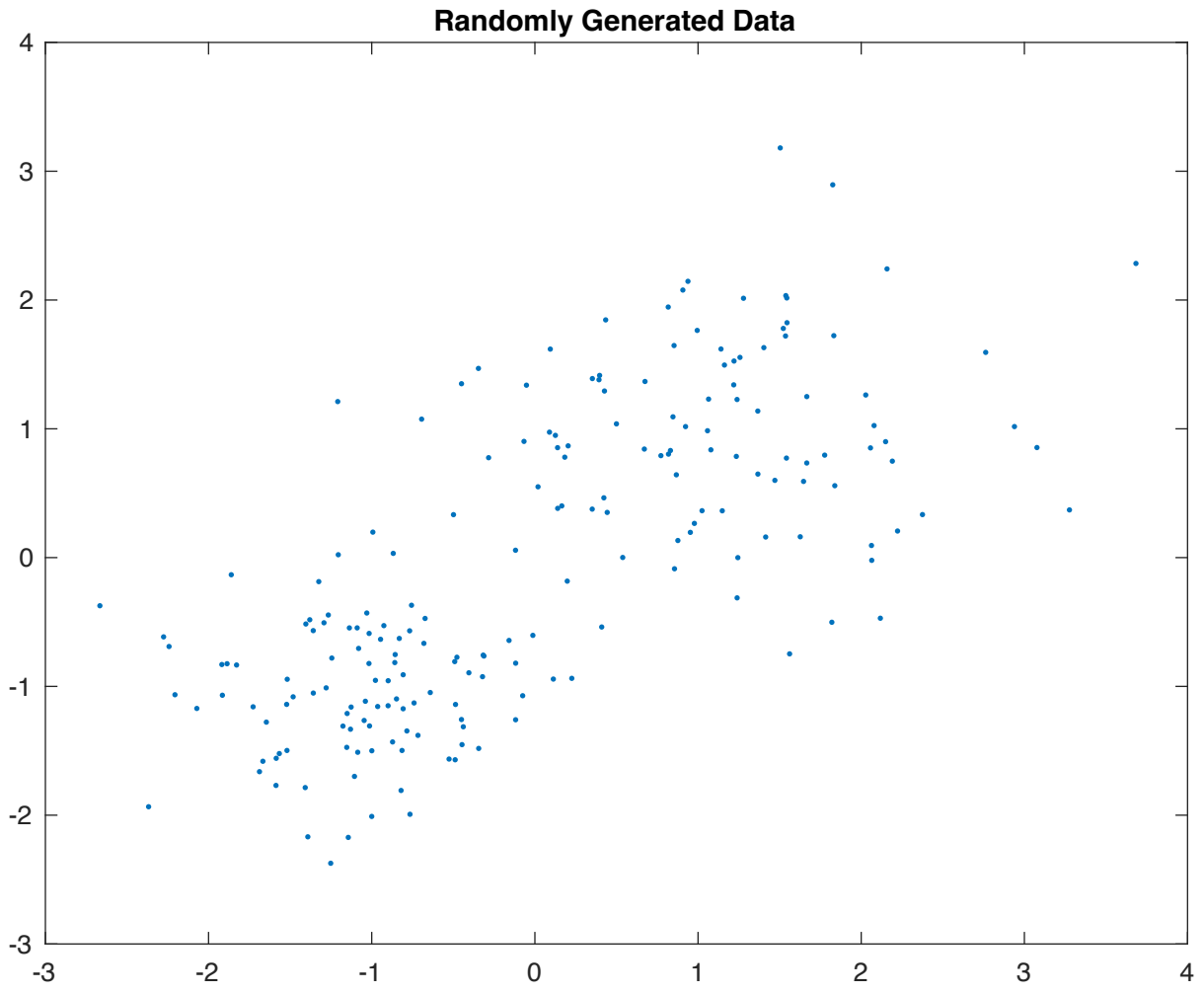
K-means clustering algorithm

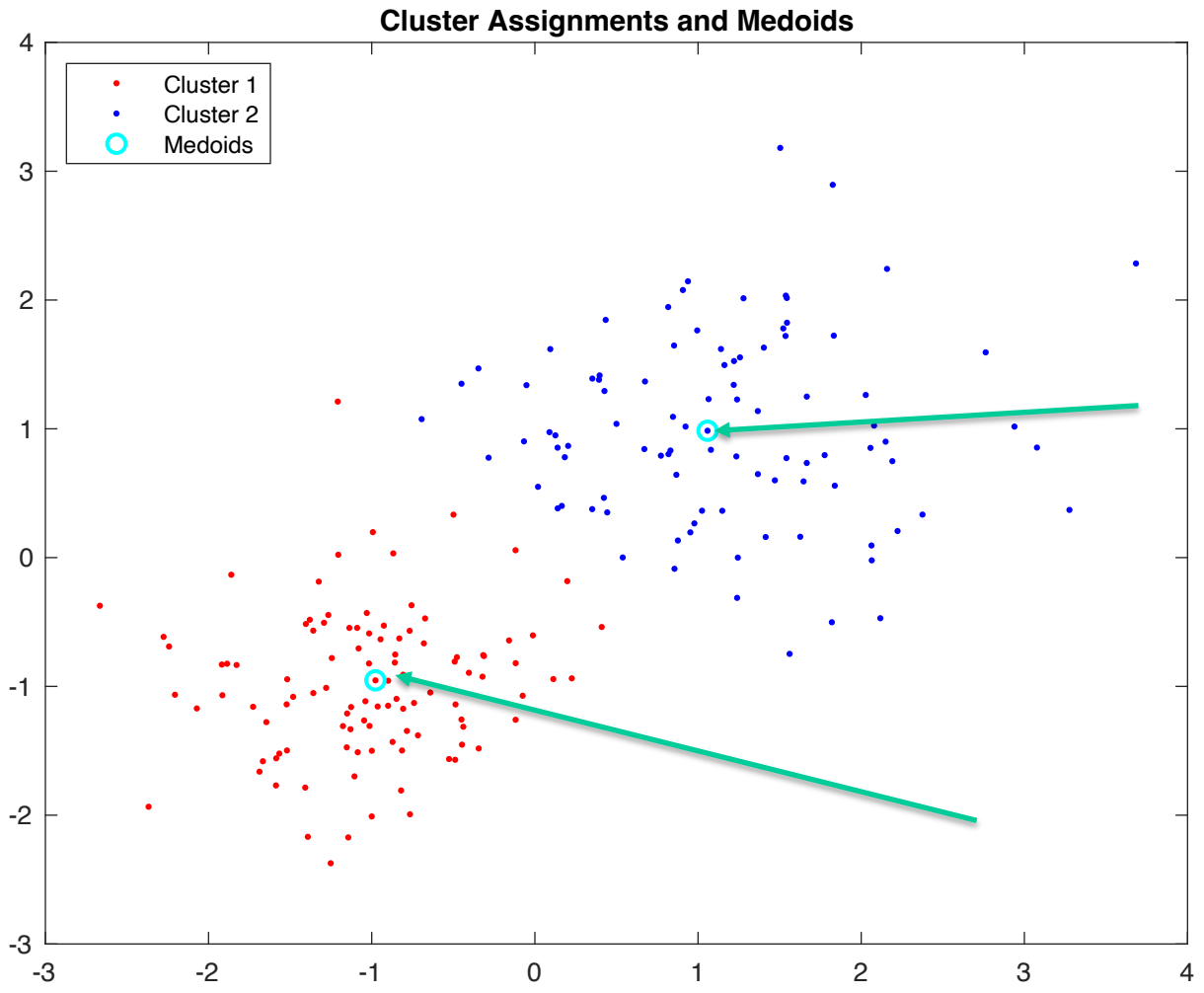


K-means clustering algorithm



K-medoids clustering algorithm




Cluster Assignments and Medoids
Cluster 1
Cluster 2
Medoids
4
3
2
1
0
-1
-2
-3
-3
-2
-1
0
1
2
3
4

Some comments

- **K-means**: computing the center of the cluster
  - Linear complexity O(n)
  - Random Initialization
- **K-medoids**: median vector of the cluster
  - More complex
  - Less sensitive to outlier
- The **choice of K** is not trivial!!
- **Other clustering algorithms are available (see lectures on clustering)**