Défis en Intelligence Artificielle
Défi 3 : L'IA pour l'analyse et la prévision de séries temporelles (II/III)

Souhaib BEN TAIEB

UMONS
Université de Mons

Faculté des Sciences

December 10, 2020

Kaggle competition

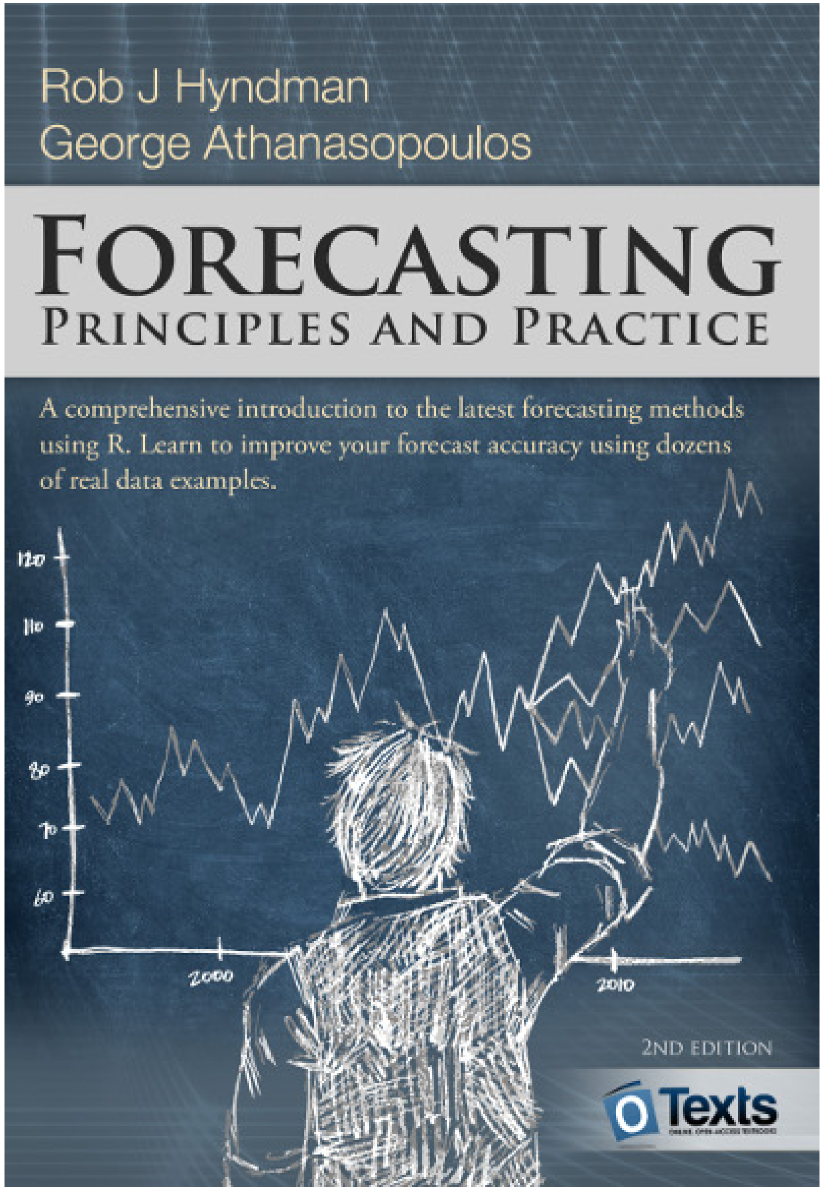
- Web Traffic Time Series Forecasting
  - https://www.kaggle.com/t/17fbaf069307464094828f82a398496f
  - **IMPORTANT**: use the previous link, **not** https://www.kaggle.com/c/hands-on-ai-umons-2020-2021
  - Max. five submissoins per day
  - Notebooks available
- Google Colab or https://www.kaggle.com/kernels

Task	**Due Date**	**Value**
Project		100%
→ Kaggle submission	17 January 11:55pm	35%
→ Report	24 January 11:55pm	65%

Part I

Traditional statistical forecasting methods

Traditional statistical forecasting methods



- `https://otexts.com/fpp3/`
- Exponential smoothing methods
- **Autoregressive integrated moving average (ARIMA)**
- ...

Stationarity

Definition

If $\{y_t\}$ is a stationary time series, then for all s, the distribution of $(y_t, \dots, y_{t+s})$ does not depend on t.

A **stationary series** is:

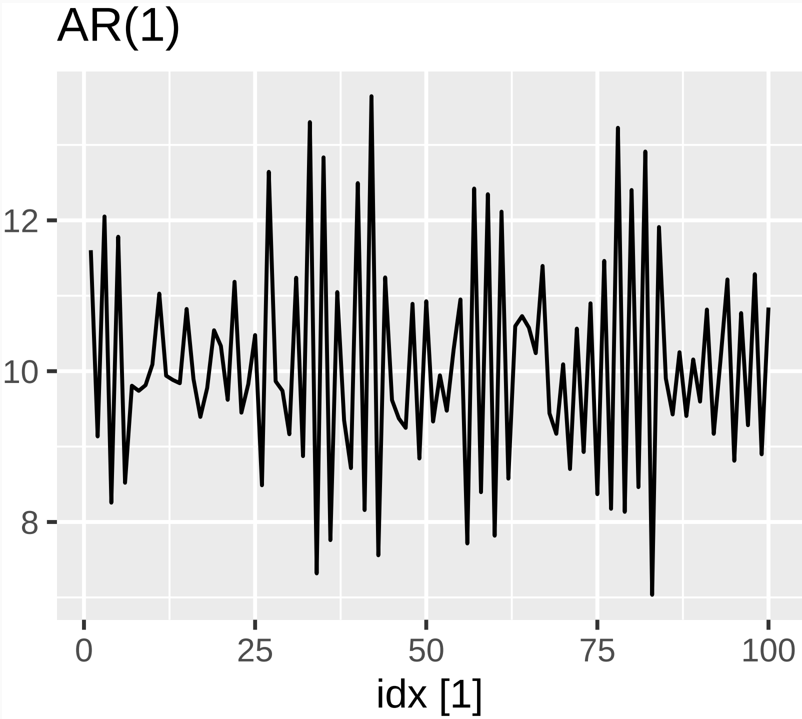
- roughly horizontal
- constant variance
- no patterns predictable in the long-term

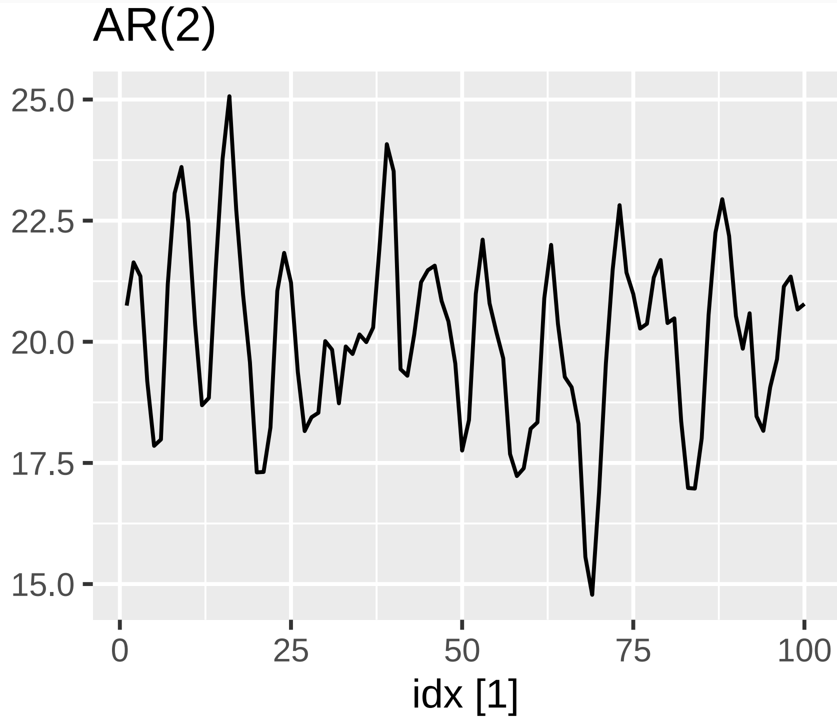
Autoregressive models

Autoregressive (AR) models:

$$y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \varepsilon_t,$$

where ε_t is white noise. This is a multiple regression with **lagged values** of y_t as predictors.

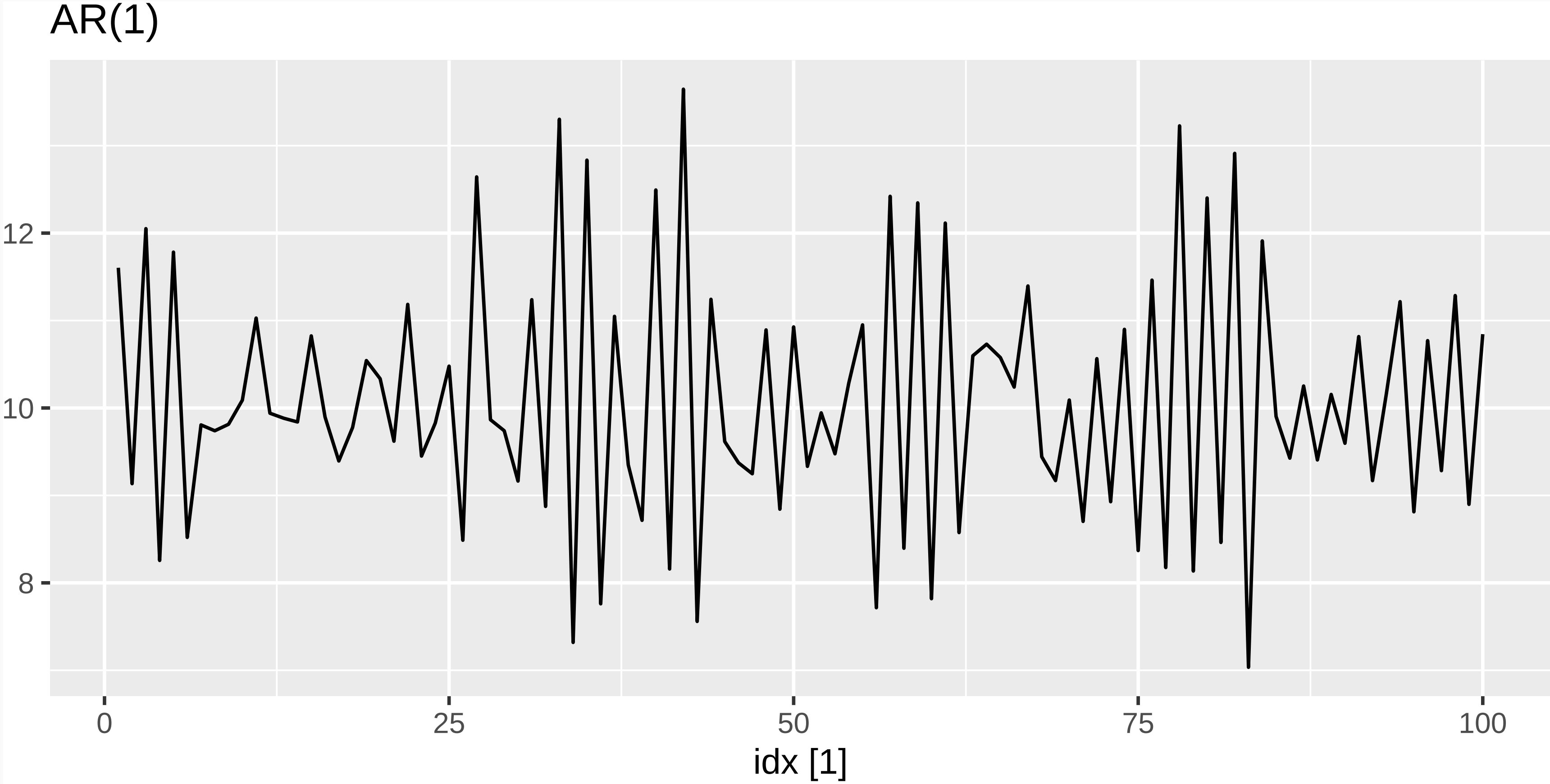




AR(1) model

$$y_t = 18 - 0.8y_{t-1} + \varepsilon_t$$

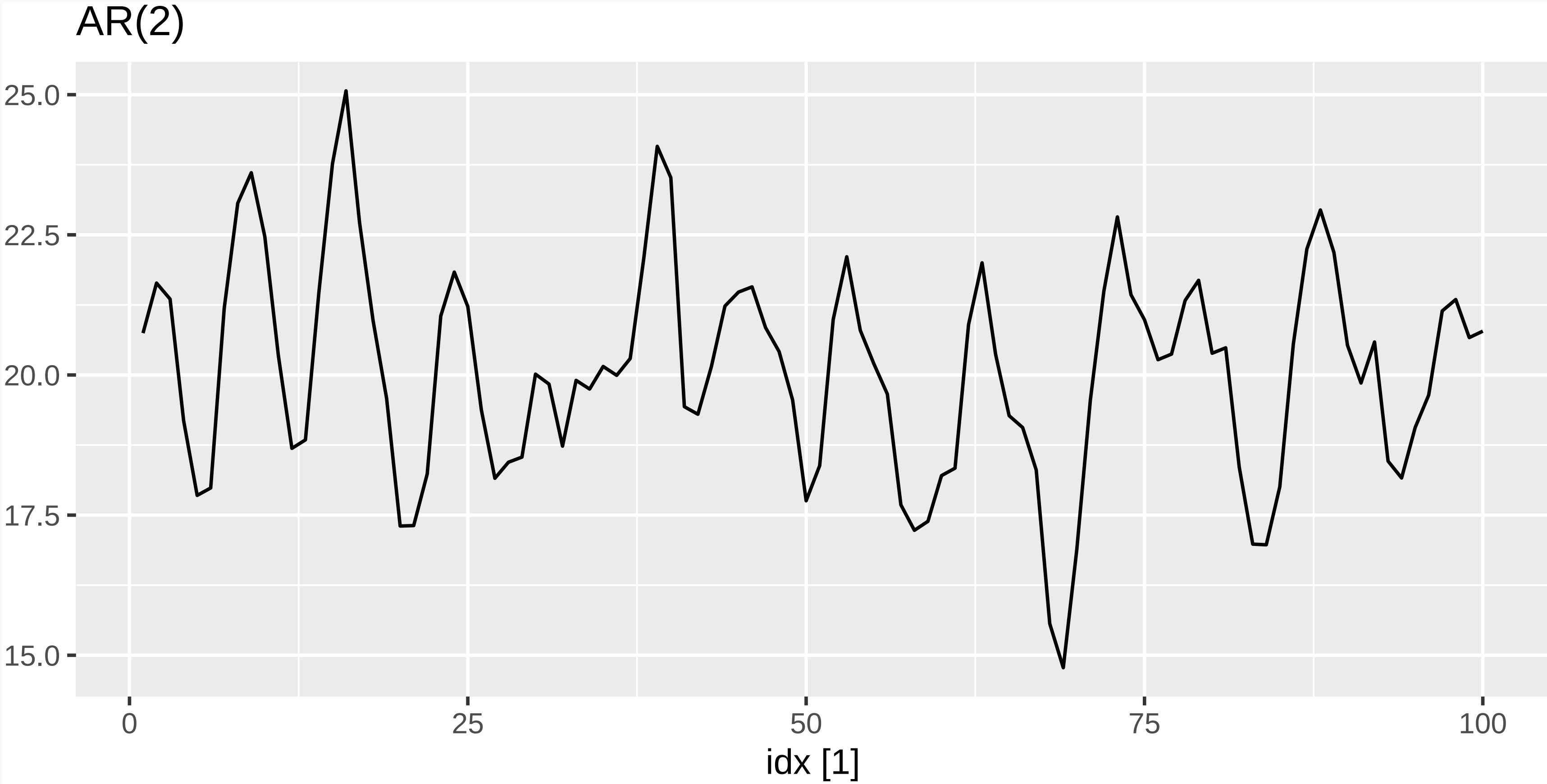
$$\varepsilon_t \sim N(0, 1), \quad T = 100.$$



AR(2) model

$$y_t = 8 + 1.3y_{t-1} - 0.7y_{t-2} + \varepsilon_t$$

$$\varepsilon_t \sim N(0, 1), \qquad T = 100.$$

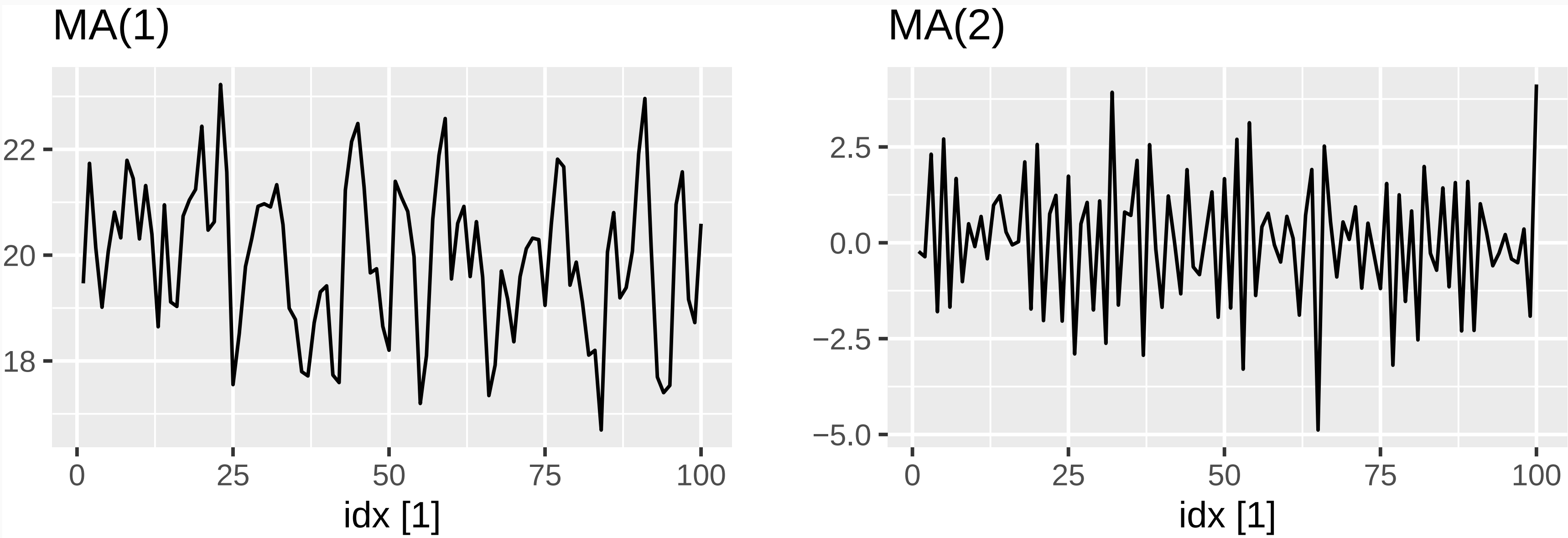


Moving Average (MA) models

Moving Average (MA) models:

$$y_t = c + \varepsilon_t + \theta_1\varepsilon_{t-1} + \theta_2\varepsilon_{t-2} + \cdots + \theta_q\varepsilon_{t-q},$$

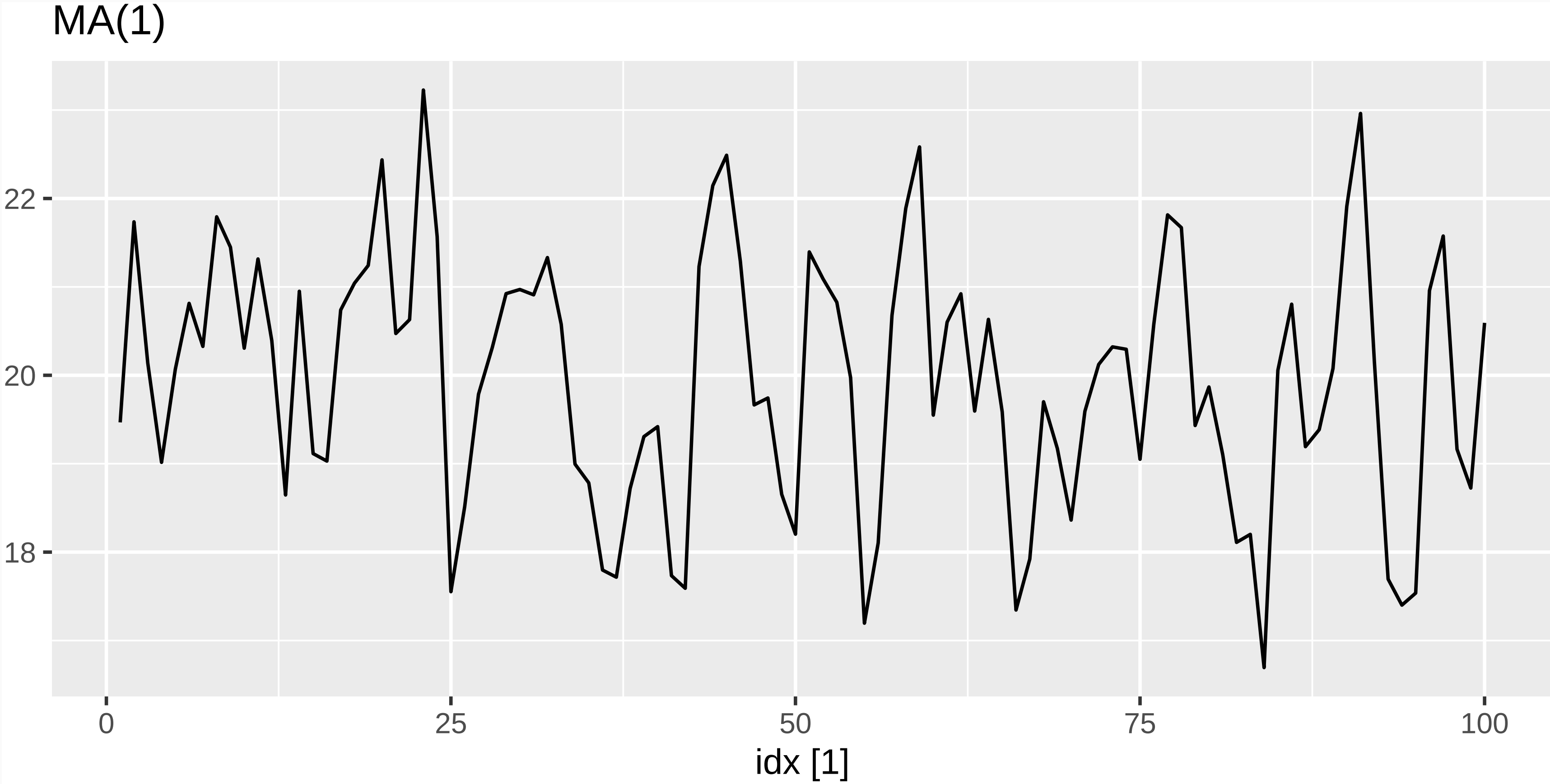
where ε_t is white noise. This is a multiple regression with **past *errors*** as predictors. *Don't confuse this with moving average smoothing!*



MA(1) model

$$y_t = 20 + \varepsilon_t + 0.8\varepsilon_{t-1}$$

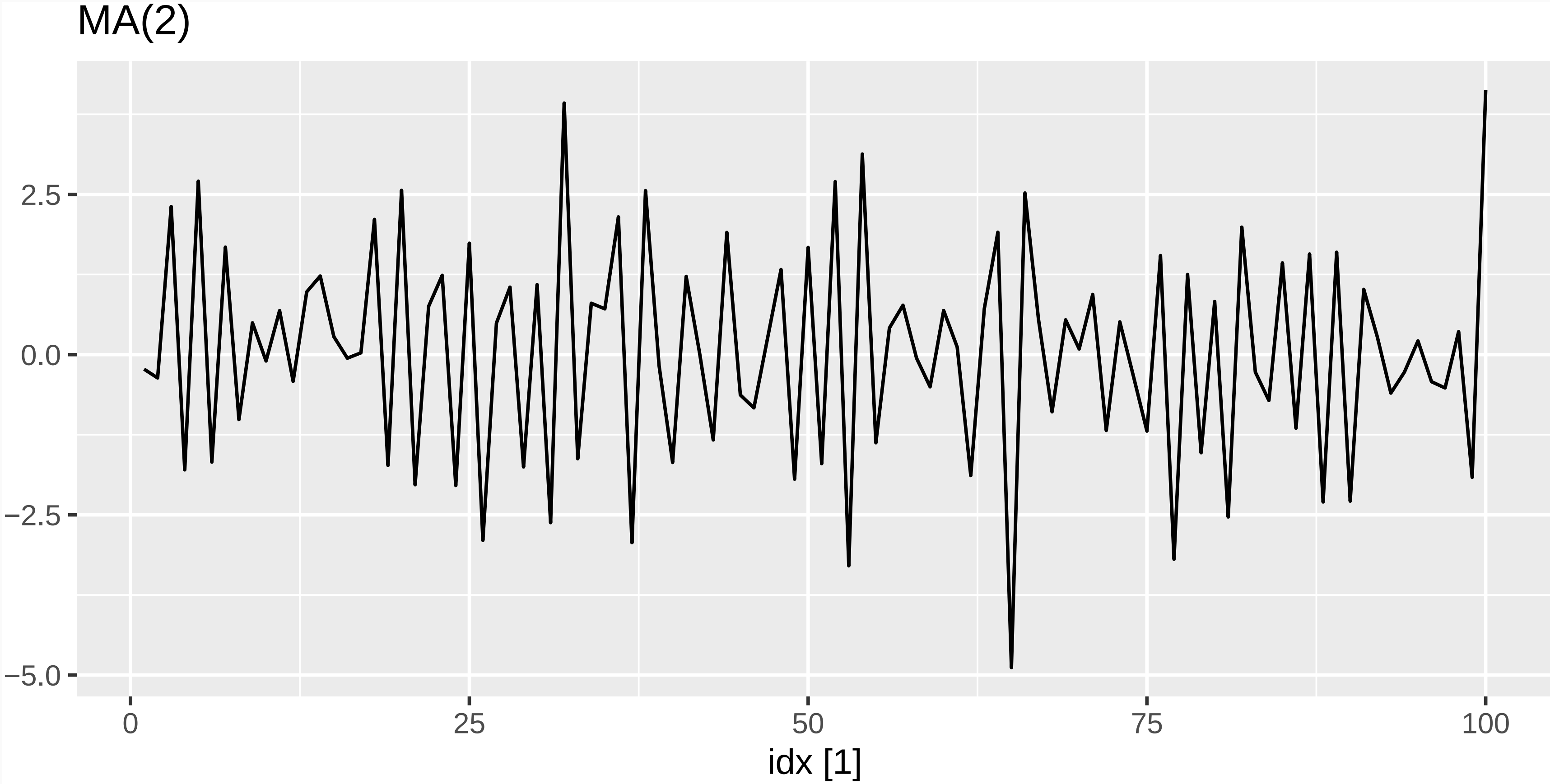
$$\varepsilon_t \sim N(0, 1), \quad T = 100.$$



MA(2) model

$$y_t = \varepsilon_t - \varepsilon_{t-1} + 0.8\varepsilon_{t-2}$$

$$\varepsilon_t \sim N(0, 1), \quad T = 100.$$



ARMA models

Autoregressive Moving Average models:

$$y_t = c + \phi_1 y_{t-1} + \cdots + \phi_p y_{t-p} + \theta_1 \varepsilon_{t-1} + \cdots + \theta_q \varepsilon_{t-q} + \varepsilon_t.$$

- Predictors include both **lagged values of y_t and lagged errors.**
- Conditions on coefficients ensure stationarity.
-

Maximum likelihood estimation

Having identified the model order, we need to estimate the parameters $c, \phi_1, \dots, \phi_p, \theta_1, \dots, \theta_q$.

- MLE is very similar to least squares estimation obtained by minimizing
$$\sum_{t-1}^{T} e_t^2$$
- The `auto_arima` function allows CLS or MLE estimation.
- Non-linear optimization must be used in either case.
- Different software will give different estimates.

Information criteria

Akaike's Information Criterion (AIC):

$$\text{AIC} = -2\log(L) + 2(p + q + k + 1),$$

where L is the likelihood of the data,

$k = 1$ if $c \neq 0$ and $k = 0$ if $c = 0$.

Corrected AIC:

$$\text{AICc} = \text{AIC} + \frac{2(p+q+k+1)(p+q+k+2)}{T-p-q-k-2}.$$

Bayesian Information Criterion:

$$\text{BIC} = \text{AIC} + [\log(T) - 2](p + q + k + 1).$$

Good models are obtained by minimizing either the AIC, AICc or BIC. Our preference is to use the AICc.

Stationarity

Definition

If $\{y_t\}$ is a stationary time series, then for all s, the distribution of $(y_t, \dots, y_{t+s})$ does not depend on t.

Transformations help to **stabilize the variance**.

For ARMA modelling, we also need to **stabilize the mean**.

Differencing

- Differencing helps to **stabilize the mean**.
- The differenced series is the *change* between each observation in the original series: $y'_t = y_t - y_{t-1}$.
- The differenced series will have only $T - 1$ values since it is not possible to calculate a difference y'_1 for the first observation.

Second-order differencing

Occasionally the differenced data will not appear stationary and it may be necessary to difference the data a second time:

$$
\begin{aligned}
y''_t &= y'_t - y'_{t-1} \\
&= (y_t - y_{t-1}) - (y_{t-1} - y_{t-2}) \\
&= y_t - 2y_{t-1} + y_{t-2}.
\end{aligned}
$$

- y''_t will have $T - 2$ values.
- In practice, it is almost never necessary to go beyond second-order differences.

Seasonal differencing

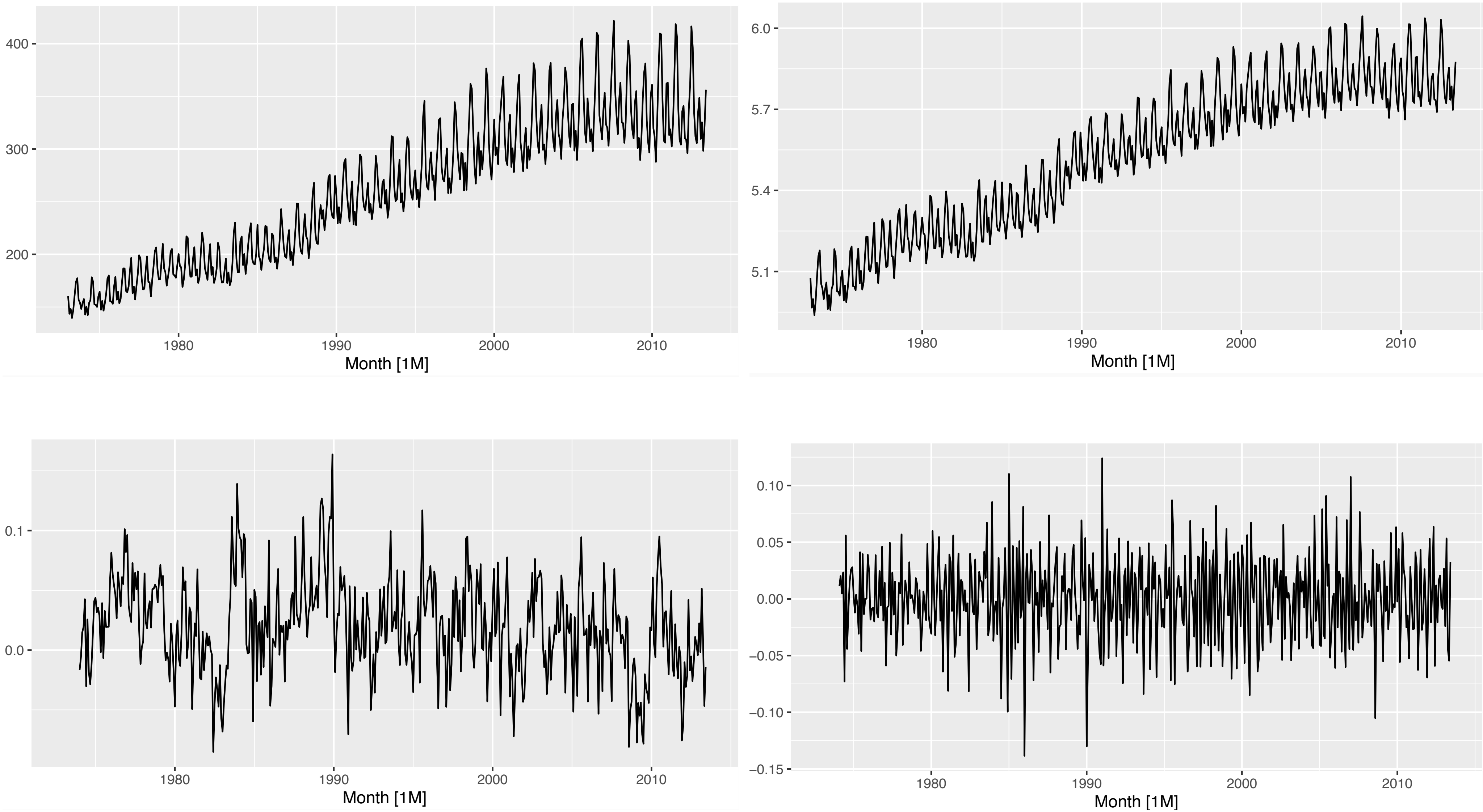
A seasonal difference is the difference between an observation and the corresponding observation from the previous year.

$$y'_t = y_t - y_{t-m}$$

where m = number of seasons.

- For monthly data $m = 12$.
- For quarterly data $m = 4$.

Example



(1) Initial series; (2) Log transformation; (3) Seasonal difference; (4) first difference

ARIMA models

Autoregressive Integrated Moving Average models

ARIMA(p, d, q) model

AR: p = order of the autoregressive part

I: d = degree of first differencing involved

MA: q = order of the moving average part.

- White noise model: ARIMA(0,0,0)
- Random walk: ARIMA(0,1,0) with no constant
- Random walk with drift: ARIMA(0,1,0) with const.
- AR(p): ARIMA(p,0,0)
- MA(q): ARIMA(0,0,q)

Seasonal ARIMA models

ARIMA	$\underbrace{(p, d, q)}$	$\underbrace{(P, D, Q)_m}$
	$\uparrow$	$\uparrow$
	Non-seasonal part of the model	Seasonal part of of the model

where m = number of observations per year.

Software

pmdarima.arima.auto_arima

pmdarima.arima.auto_arima(*y, X=None, start_p=2, d=None, start_q=2, max_p=5, max_d=2, max_q=5, start_P=1, D=None, start_Q=1, max_P=2, max_D=1, max_Q=2, max_order=5, m=1, seasonal=True, stationary=False, information_criterion='aic', alpha=0.05, test='kpss', seasonal_test='ocsb', stepwise=True, n_jobs=1, start_params=None, trend=None, method='lbfgs', maxiter=50, offset_test_args=None, seasonal_test_args=None, suppress_warnings=True, error_action='trace', trace=False, random=False, random_state=None, n_fits=10, return_valid_fits=False, out_of_sample_size=0, scoring='mse', scoring_args=None, with_intercept='auto', sarimax_kwargs=None, **fit_args*) [source] [source]

- https://alkaline-ml.com/pmdarima/modules/classes.html
- https://alkaline-ml.com/pmdarima/modules/generated/pmdarima.arima.AutoARIMA.html#pmdarima.arima.AutoARIMA
- https://alkaline-ml.com/pmdarima/tips_and_tricks.html

Software

```
                               SARIMAX Results
==============================================================================
Dep. Variable:                      y   No. Observations:                 1000
Model:               SARIMAX(2, 0, 2)   Log Likelihood               -1398.466
Date:                Thu, 10 Dec 2020   AIC                           2806.931
Time:                        17:03:42   BIC                           2831.470
Sample:                             0   HQIC                          2816.258
                               - 1000
Covariance Type:                  opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1          0.7012      0.075      9.383      0.000       0.555       0.848
ar.L2         -0.2353      0.060     -3.910      0.000      -0.353      -0.117
ma.L1          0.6982      0.072      9.724      0.000       0.557       0.839
ma.L2          0.3858      0.051      7.513      0.000       0.285       0.486
sigma2         0.9578      0.042     22.699      0.000       0.875       1.041
===================================================================================
Ljung-Box (Q):                       28.93   Jarque-Bera (JB):                  3.83
Prob(Q):                              0.90   Prob(JB):                          0.15
Heteroskedasticity (H):               0.89   Skew:                              0.14
Prob(H) (two-sided):                  0.28   Kurtosis:                          3.09
===================================================================================
```

Part II

Modern statistical/machine learning forecasting methods

AI for time series forecasting

- Reduce the problem of time series forecasting to one or multiple regression problems
  - Use any AI learning algorithm for regression
  - Specific AI architectures have been developped for sequential data
- Challenges
  - (Statistically) depedent data
  - Non-stationarity
  - Specific patterns: seasonality, trend, cycle, etc
  - Multi-step ahead forecasting, i.e. sequential predictions
- Model training
  - Use training data from the past to predict the future.
  - No (naive) shuffling of a time series $\rightarrow$ destroy the temporal dependence structure.

Training with the validation set approach

$$\underbrace{y_1, y_2, y_3, y_4, y_5, y_6}_{\text{Training}}, \underbrace{y_7, y_8, y_9, y_{10}}_{\text{Validation}}$$

$$y_1, y_2, y_3, y_4, y_5, y_6 \longrightarrow y_7$$

$$\longrightarrow y_8$$

$$\longrightarrow y_9$$

$$\longrightarrow y_{10}$$

X			**y**
y_{t-2}	y_{t-1}	y_t	y_{t+1}
y_1	y_2	y_3	y_4
y_2	y_3	y_4	y_5
y_3	y_4	y_5	y_6
y_4	y_5	y_6	y_7
y_5	y_6	y_7	y_8
y_6	y_7	y_8	y_9
y_7	y_8	y_9	y_{10}

Training with the rolling-origin approach

$$\underbrace{y_1, y_2, y_3, y_4, y_5, y_6}_{\text{Training}}, \underbrace{\color{red}{y_7, y_8, y_9, y_{10}}}_{\color{red}{\text{Validation}}}$$

- $y_1, y_2, y_3, y_4, y_5, y_6 \longrightarrow \color{red}{y_7}$
- $y_1, y_2, y_3, y_4, y_5, y_6, y_7 \longrightarrow \color{red}{y_8}$
- $y_1, y_2, y_3, y_4, y_5, y_6, y_7, y_8 \longrightarrow \color{red}{y_9}$
- $y_1, y_2, y_3, y_4, y_5, y_6, y_7, y_8, y_9 \longrightarrow \color{red}{y_{10}}$

X			**y**
y_{t-2}	y_{t-1}	y_t	y_{t+1}
y_1	y_2	y_3	y_4
y_2	y_3	y_4	y_5
y_3	y_4	y_5	y_6
y_4	y_5	y_6	$\color{red}{y_7}$

X			**y**
y_{t-2}	y_{t-1}	y_t	y_{t+1}
y_1	y_2	y_3	y_4
y_2	y_3	y_4	y_5
y_3	y_4	y_5	y_6
y_4	y_5	y_6	y_7
y_5	y_6	y_7	$\color{red}{y_8}$

X			**y**
y_{t-2}	y_{t-1}	y_t	y_{t+1}
y_1	y_2	y_3	y_4
y_2	y_3	y_4	y_5
y_3	y_4	y_5	y_6
y_4	y_5	y_6	y_7
y_5	y_6	y_7	y_8
y_6	y_7	y_8	$\color{red}{y_9}$

X			**y**
y_{t-2}	y_{t-1}	y_t	y_{t+1}
y_1	y_2	y_3	y_4
y_2	y_3	y_4	y_5
y_3	y_4	y_5	y_6
y_4	y_5	y_6	y_7
y_5	y_6	y_7	y_8
y_6	y_7	y_8	y_9
y_7	y_8	y_9	$\color{red}{y_{10}}$

Training with the rolling-origin approach

$$\underbrace{y_1, y_2, y_3, y_4, y_5, y_6}_{\text{Training}}, \underbrace{y_7, y_8, y_9, y_{10}}_{\text{Validation}}$$

- $y_1, y_2, y_3, y_4, y_5, y_6 \longrightarrow y_7$
- $y_2, y_3, y_4, y_5, y_6, y_7 \longrightarrow y_8$
- $y_3, y_4, y_5, y_6, y_7, y_8 \longrightarrow y_9$
- $y_4, y_5, y_6, y_7, y_8, y_9 \longrightarrow y_{10}$

X			**y**
y_{t-2}	y_{t-1}	y_t	y_{t+1}
y_1	y_2	y_3	y_4
y_2	y_3	y_4	y_5
y_3	y_4	y_5	y_6
y_4	y_5	y_6	y_7

X			**y**
y_{t-2}	y_{t-1}	y_t	y_{t+1}
y_2	y_3	y_4	y_5
y_3	y_4	y_5	y_6
y_4	y_5	y_6	y_7
y_5	y_6	y_7	y_8

X			**y**
y_{t-2}	y_{t-1}	y_t	y_{t+1}
y_3	y_4	y_5	y_6
y_4	y_5	y_6	y_7
y_5	y_6	y_7	y_8
y_6	y_7	y_8	y_9

X			**y**
y_{t-2}	y_{t-1}	y_t	y_{t+1}
y_4	y_5	y_6	y_7
y_5	y_6	y_7	y_8
y_6	y_7	y_8	y_9
y_7	y_8	y_9	y_{10}

Multi-step forecasting - recursive strategy

$$y_1, y_2, y_3, y_4, y_5, y_6, y_7, y_8, y_9, y_{10} \rightarrow ?, ?, ?$$

X			**y**
y_{t-2}	y_{t-1}	y_t	y_{t+1}
y_1	y_2	y_3	y_4
y_2	y_3	y_4	y_5
y_3	y_4	y_5	y_6
y_4	y_5	y_6	y_7
y_5	y_6	y_7	y_8
y_6	y_7	y_8	y_9
y_7	y_8	y_9	y_{10}
y_8	y_9	y_{10}	?

$$y_8, y_9, y_{10} \rightarrow \hat{y}_{11} \qquad y_9, y_{10}, \hat{y}_{11} \rightarrow \hat{y}_{12} \qquad y_{10}, \hat{y}_{11}, \hat{y}_{12} \rightarrow \hat{y}_{13}$$

Multi-step forecasting - direct strategy

$$y_1, y_2, y_3, y_4, y_5, y_6, y_7, y_8, y_9, y_{10} \rightarrow ?, ?, ?$$

$\mathbf{X}$			$\mathbf{y}$
y_{t-2}	y_{t-1}	y_t	y_{t+1}
y_1	y_2	y_3	y_4
y_2	y_3	y_4	y_5
y_3	y_4	y_5	y_6
y_4	y_5	y_6	y_7
y_5	y_6	y_7	y_8
y_6	y_7	y_8	y_9
y_7	y_8	y_9	y_{10}
y_8	y_9	y_{10}	?

$\mathbf{X}$			$\mathbf{y}$
y_{t-2}	y_{t-1}	y_t	y_{t+2}
y_1	y_2	y_3	y_5
y_2	y_3	y_4	y_6
y_3	y_4	y_5	y_7
y_4	y_5	y_6	y_8
y_5	y_6	y_7	y_9
y_6	y_7	y_8	y_{10}
y_8	y_9	y_{10}	?

$\mathbf{X}$			$\mathbf{y}$
y_{t-2}	y_{t-1}	y_t	y_{t+3}
y_1	y_2	y_3	y_6
y_2	y_3	y_4	y_7
y_3	y_4	y_5	y_8
y_4	y_5	y_6	y_9
y_5	y_6	y_7	y_{10}
y_8	y_9	y_{10}	?

$$y_8, y_9, y_{10} \rightarrow \hat{y}_{11} \qquad y_8, y_9, y_{10} \rightarrow \hat{y}_{12} \qquad y_8, y_9, y_{10} \rightarrow \hat{y}_{13}$$

Multi-step forecasting - multi-output strategy

$$y_1, y_2, y_3, y_4, y_5, y_6, y_7, y_8, y_9, y_{10} \rightarrow ?, ?, ?$$

$\mathbf{X}$			$\mathbf{y}$		
y_{t-2}	y_{t-1}	y_t	y_{t+1}	y_{t+2}	y_{t+3}
y_1	y_2	y_3	y_4	y_5	y_6
y_2	y_3	y_4	y_5	y_6	y_7
y_3	y_4	y_5	y_6	y_7	y_8
y_4	y_5	y_6	y_7	y_8	y_9
y_5	y_6	y_7	y_8	y_9	y_{10}
y_8	y_9	y_{10}	?		

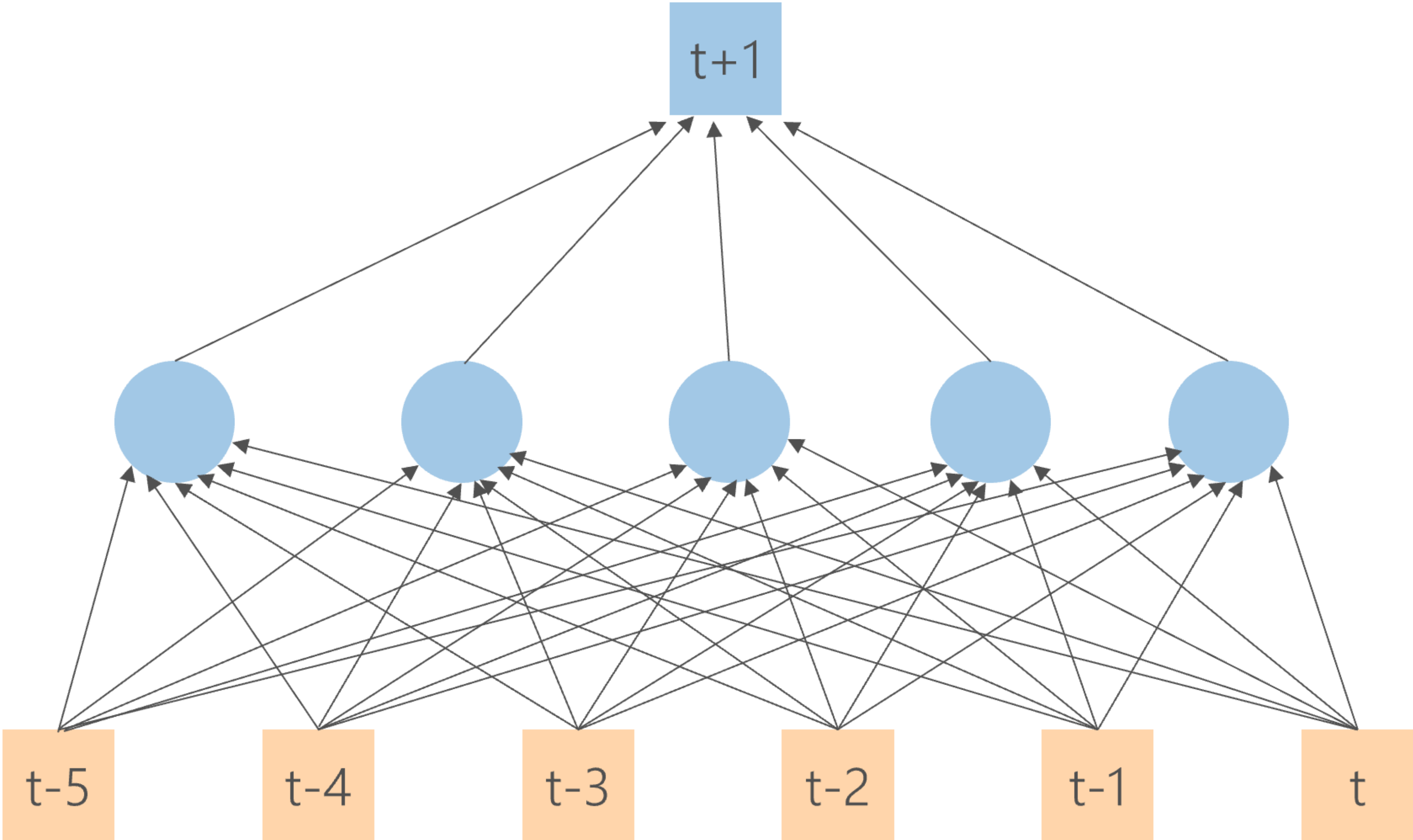
$$y_8, y_9, y_{10} \rightarrow \hat{y}_{11}, \hat{y}_{12}, \hat{y}_{13}$$

$\rightarrow$ The multi-output strategy requires a model that can deal with multiple outputs, e.g. neural networks.

Why (Deep) Neural Networks?

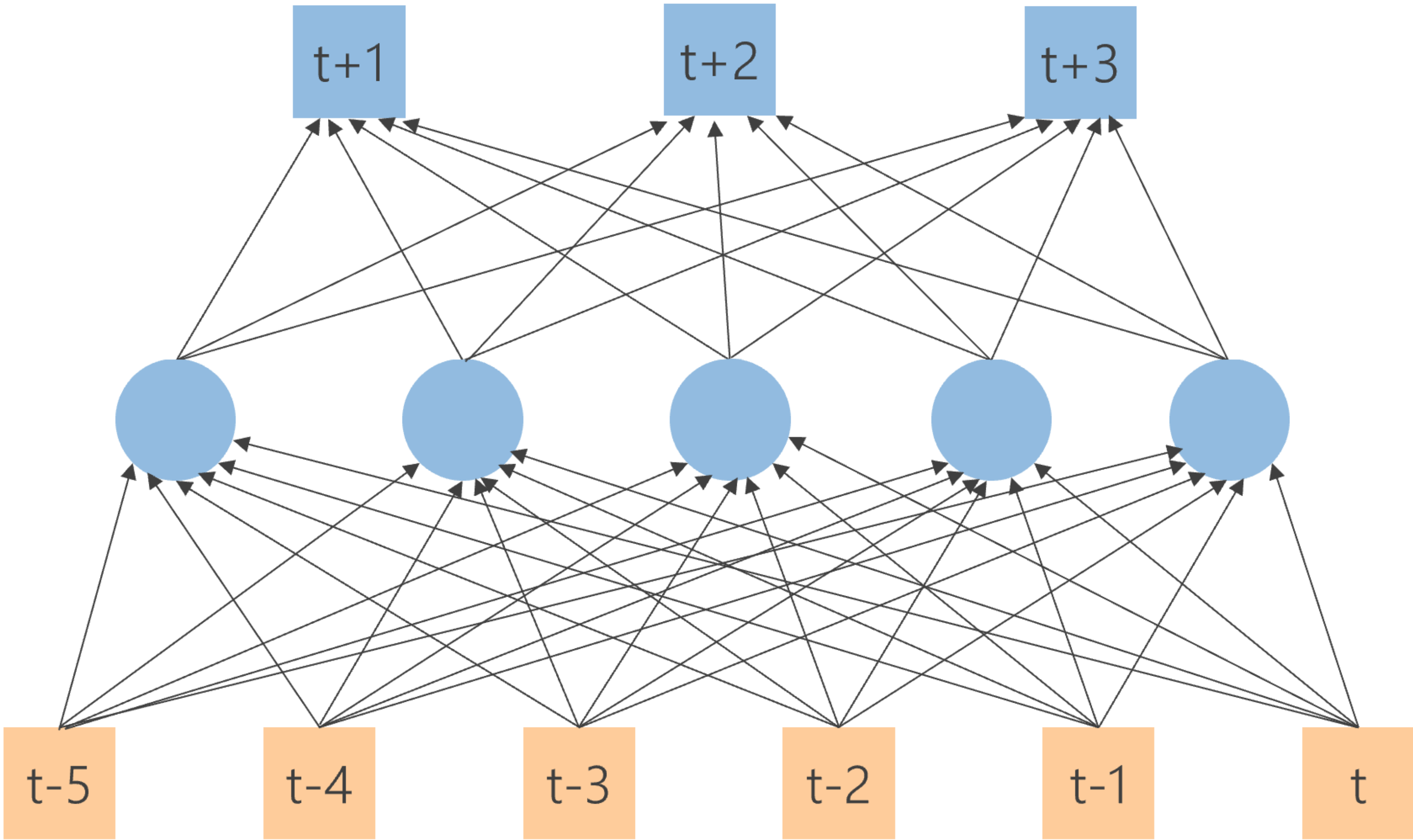
- Deep learning model has been shown to perform well in many scenarios
- Very effective at feature extraction
- Flexible and expressive
- Easily inject exogenous features into the model
- Learn from large time series datasets
- ...
- Challenges
  - Require a lot of data (in general)
  - Computationally demanding
  - Hard to train

Single output network



→ Multi-step forecasts are obtained recursively

Multi-output network



→ The network produces multi-step forecasts.

Neural network hyperparameters

Hyperparameters are adjustable parameters that define the model architecture and govern the learning process. By contrast, other parameters (such as node weights) are derived via model training.

- Architecture
  - Types of layers, number of layers, layer order, number of neurons per layer, layer activations, etc.
- Optimization
  - Opitmizer, weight initialization, learning rate, batch size, number of epochs, stopping criterion, etc.
- Loss function
  - Loss function, form of regularization, etc.
- ...

Hyperparameter search/tuning

- Search across various hyperparameter configurations
- Find the configuration that results in best (out-of-sample) performance
- Grid search vs random search
- Challenges
  - Huge hyperparameter space to explore
  - Computationally and time demanding