



# High dimensional time series analysis



## 4. Automatic forecasting algorithms

# Outline

- 1 Exponential smoothing
- 2 Lab Session 7
- 3 ARIMA models
- 4 Lab Session 8
- 5 Seasonal ARIMA models
- 6 Lab Session 9
- 7 Forecast accuracy measures

# Outline

- 1 Exponential smoothing
- 2 Lab Session 7
- 3 ARIMA models
- 4 Lab Session 8
- 5 Seasonal ARIMA models
- 6 Lab Session 9
- 7 Forecast accuracy measures

# Historical perspective

- Developed in the 1950s and 1960s as methods (algorithms) to produce point forecasts.
- Combine a “level”, “trend” (slope) and “seasonal” component to describe a time series.
- The rate of change of the components are controlled by “smoothing parameters”:  $\alpha$ ,  $\beta$  and  $\gamma$  respectively.
- Need to choose best values for the smoothing parameters (and initial states).
- Equivalent ETS state space models developed in the 1990s and 2000s.

# A model for levels, trends, and seasonalities

We want a model that captures the level ( $\ell_t$ ), trend ( $b_t$ ) and seasonality ( $s_t$ ).

**How do we combine these elements?**

# A model for levels, trends, and seasonalities

We want a model that captures the level ( $\ell_t$ ), trend ( $b_t$ ) and seasonality ( $s_t$ ).

How do we combine these elements?

**Additively?**

$$y_t = \ell_{t-1} + b_{t-1} + s_{t-m} + \varepsilon_t$$

# A model for levels, trends, and seasonalities

We want a model that captures the level ( $\ell_t$ ), trend ( $b_t$ ) and seasonality ( $s_t$ ).

**How do we combine these elements?**

**Additively?**

$$y_t = \ell_{t-1} + b_{t-1} + s_{t-m} + \varepsilon_t$$

**Multiplicatively?**

$$y_t = \ell_{t-1} b_{t-1} s_{t-m} (1 + \varepsilon_t)$$

# A model for levels, trends, and seasonalities

We want a model that captures the level ( $\ell_t$ ), trend ( $b_t$ ) and seasonality ( $s_t$ ).

**How do we combine these elements?**

**Additively?**

$$y_t = \ell_{t-1} + b_{t-1} + s_{t-m} + \varepsilon_t$$

**Multiplicatively?**

$$y_t = \ell_{t-1} b_{t-1} s_{t-m} (1 + \varepsilon_t)$$

**Perhaps a mix of both?**

$$y_t = (\ell_{t-1} + b_{t-1}) s_{t-m} + \varepsilon_t$$



# A model for levels, trends, and seasonalities

We want a model that captures the level ( $\ell_t$ ), trend ( $b_t$ ) and seasonality ( $s_t$ ).

How do we combine these elements?

Additively?

$$y_t = \ell_{t-1} + b_{t-1} + s_{t-m} + \varepsilon_t$$

Multiplicatively?

$$y_t = \ell_{t-1} b_{t-1} s_{t-m} (1 + \varepsilon_t)$$

Perhaps a mix of both?

$$y_t = (\ell_{t-1} + b_{t-1}) s_{t-m} + \varepsilon_t$$

How do the level, trend and seasonal components evolve over time?

# ETS models

General notation

ETS : ExponenTial Smoothing



Error Trend Season

**Error:** Additive ("A") or multiplicative ("M")

# ETS models

General notation

ETS : ExponenTial Smoothing



Error Trend Season

**Error:** Additive ("A") or multiplicative ("M")

**Trend:** None ("N"), additive ("A"), multiplicative ("M"), or damped ("Ad" or "Md").

# ETS models

General notation

ETS : ExponenTial Smoothing



Error Trend Season

**Error:** Additive ("A") or multiplicative ("M")

**Trend:** None ("N"), additive ("A"), multiplicative ("M"), or damped ("Ad" or "Md").

**Seasonality:** None ("N"), additive ("A") or multiplicative ("M")

## ETS(A,N,N): SES with additive errors

Forecast equation

$$\hat{y}_{T+h|T} = \ell_T$$

Measurement equation

$$y_t = \ell_{t-1} + \varepsilon_t$$

State equation

$$\ell_t = \ell_{t-1} + \alpha \varepsilon_t$$

where  $\varepsilon_t \sim \text{NID}(0, \sigma^2)$ .

# ETS(A,N,N): SES with additive errors

Forecast equation

$$\hat{y}_{T+h|T} = \ell_T$$

Measurement equation

$$y_t = \ell_{t-1} + \varepsilon_t$$

State equation

$$\ell_t = \ell_{t-1} + \alpha \varepsilon_t$$

where  $\varepsilon_t \sim \text{NID}(0, \sigma^2)$ .

- “innovations” or “single source of error” because equations have the same error process,  $\varepsilon_t$ .
- Measurement equation: relationship between observations and states.
- Transition/state equation(s): evolution of the state(s) through time.

# ETS(M,N,N): SES with multiplicative errors

Forecast equation	$\hat{y}_{T+h T} = \ell_T$
Measurement equation	$y_t = \ell_{t-1}(1 + \varepsilon_t)$
State equation	$\ell_t = \ell_{t-1}(1 + \alpha\varepsilon_t)$

where  $\varepsilon_t \sim \text{NID}(0, \sigma^2)$ .

## ETS(M,N,N): SES with multiplicative errors

Forecast equation	$\hat{y}_{T+h T} = \ell_T$
Measurement equation	$y_t = \ell_{t-1}(1 + \varepsilon_t)$
State equation	$\ell_t = \ell_{t-1}(1 + \alpha\varepsilon_t)$

where  $\varepsilon_t \sim \text{NID}(0, \sigma^2)$ .

- Models with additive and multiplicative errors with the same parameters generate the same point forecasts but different prediction intervals.



# ETS(A,A,N): Holt's linear trend

## Additive errors

Forecast equation  $\hat{y}_{T+h|T} = \ell_T + hb_T$

Measurement equation  $y_t = \ell_{t-1} + b_{t-1} + \varepsilon_t$

State equations  $\ell_t = \ell_{t-1} + b_{t-1} + \alpha\varepsilon_t$

$$b_t = b_{t-1} + \beta\varepsilon_t$$

# ETS(A,A,N): Holt's linear trend

## Additive errors

Forecast equation  $\hat{y}_{T+h|T} = \ell_T + hb_T$

Measurement equation  $y_t = \ell_{t-1} + b_{t-1} + \varepsilon_t$

State equations  $\ell_t = \ell_{t-1} + b_{t-1} + \alpha\varepsilon_t$

$$b_t = b_{t-1} + \beta\varepsilon_t$$

## Multiplicative errors

Forecast equation  $\hat{y}_{T+h|T} = \ell_T + hb_T$

Measurement equation  $y_t = (\ell_{t-1} + b_{t-1})(1 + \varepsilon_t)$

State equations  $\ell_t = (\ell_{t-1} + b_{t-1})(1 + \alpha\varepsilon_t)$

$$b_t = b_{t-1} + \beta\varepsilon_t$$

# Example: Australian population

```
aus_economy <- global_economy %>% filter(Code == "AUS") %>%  
  mutate(Pop = Population/1e6)  
fit <- aus_economy %>% model(AAN = ETS(Pop))  
report(fit)
```

```
## Series: Pop  
## Model: ETS(A,A,N)  
## Smoothing parameters:  
##   alpha = 1  
##   beta  = 0.327  
##  
## Initial states:  
##   l      b  
## 10.1 0.222  
##  
## sigma^2: 0.0041  
##  
## AIC  AICc  BIC  
## -77.0 -75.8 -66.7
```

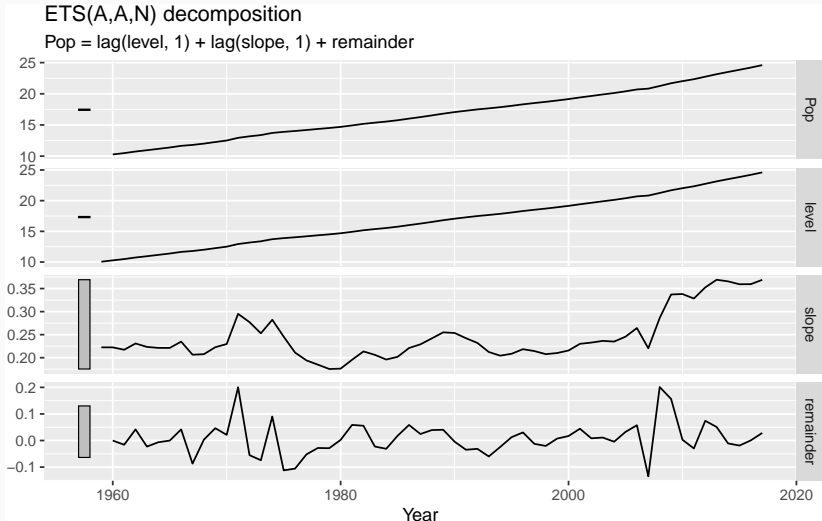
# Example: Australian population

```
components(fit)
```

```
## # A dable:                59 x 7 [1Y]
## # Key:                    Country, .model [1]
## # ETS(A,A,N) Decomposition: Pop = lag(level, 1) + lag(slope, 1)
## #   remainder
##   Country   .model Year   Pop level slope remainder
##   <fct>     <chr>  <dbl> <dbl> <dbl> <dbl>      <dbl>
## 1 Australia AAN    1959  NA    10.1 0.222 NA
## 2 Australia AAN    1960  10.3  10.3 0.222 -0.000145
## 3 Australia AAN    1961  10.5  10.5 0.217 -0.0159
## 4 Australia AAN    1962  10.7  10.7 0.231  0.0418
## 5 Australia AAN    1963  11.0  11.0 0.223 -0.0229
## 6 Australia AAN    1964  11.2  11.2 0.221 -0.00641
## 7 Australia AAN    1965  11.4  11.4 0.221 -0.000314
## 8 Australia AAN    1966  11.7  11.7 0.235  0.0418
## 9 Australia AAN    1967  11.8  11.8 0.206 -0.0869
## 10 Australia AAN    1968  12.0  12.0 0.208  0.00350
```

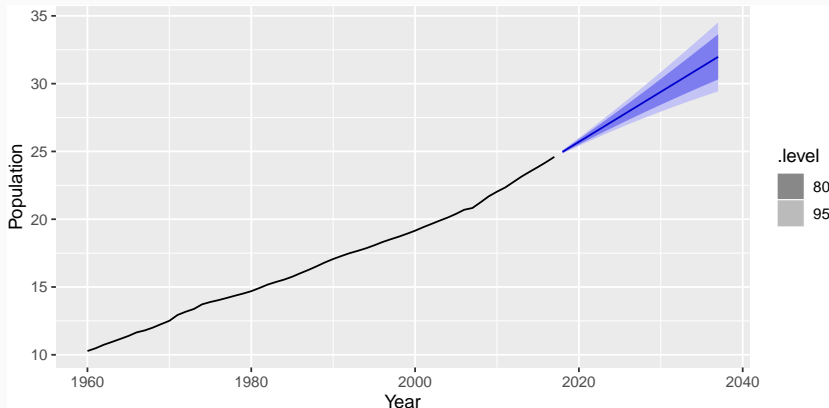
# Example: Australian population

```
components(fit) %>% autoplot()
```



# Example: Australian population

```
fit %>%  
  forecast(h = 20) %>%  
  autoplot(aus_economy) +  
  ylab("Population") + xlab("Year")
```



# ETS(A,Ad,N): Damped trend method

## Additive errors

Forecast equation  $\hat{y}_{T+h|T} = \ell_T + (\phi + \dots + \phi^{h-1})b_T$

Measurement equation  $y_t = (\ell_{t-1} + \phi b_{t-1}) + \varepsilon_t$

State equations  $\ell_t = (\ell_{t-1} + \phi b_{t-1}) + \alpha \varepsilon_t$

$$b_t = \phi b_{t-1} + \beta \varepsilon_t$$

# ETS(A,Ad,N): Damped trend method

## Additive errors

Forecast equation  $\hat{y}_{T+h|T} = \ell_T + (\phi + \dots + \phi^{h-1})b_T$

Measurement equation  $y_t = (\ell_{t-1} + \phi b_{t-1}) + \varepsilon_t$

State equations  $\ell_t = (\ell_{t-1} + \phi b_{t-1}) + \alpha \varepsilon_t$

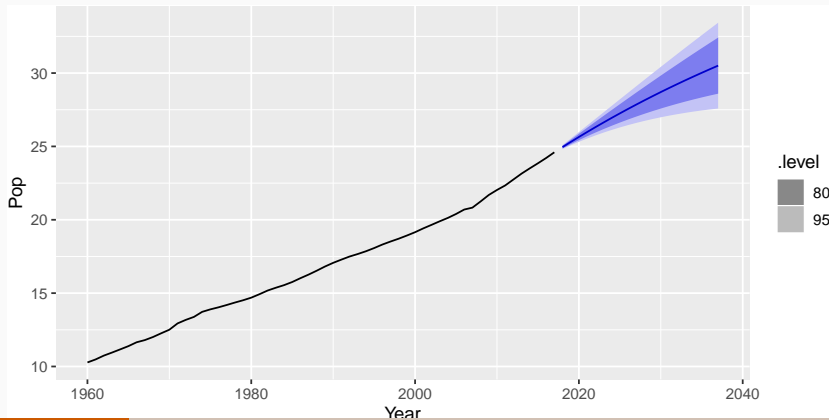
$$b_t = \phi b_{t-1} + \beta \varepsilon_t$$

- Damping parameter  $0 < \phi < 1$ .
- If  $\phi = 1$ , identical to Holt's linear trend.
- As  $h \rightarrow \infty$ ,  $\hat{y}_{T+h|T} \rightarrow \ell_T + \phi b_T / (1 - \phi)$ .
- Short-run forecasts trended, long-run forecasts constant.



# Example: Australian population

```
aus_economy %>%  
  model(holt = ETS(Pop ~ trend("Ad"))) %>%  
  forecast(h = 20) %>%  
  autoplot(aus_economy)
```



# Example: National populations

```
fit <- global_economy %>%  
  mutate(Pop = Population/1e6) %>%  
  model(ets = ETS(Pop))  
fit
```

```
## # A mable: 263 x 2  
## # Key:      Country [263]  
##   Country      ets  
##   <fct>        <model>  
## 1 Afghanistan <ETS(A,A,N)>  
## 2 Albania     <ETS(M,A,N)>  
## 3 Algeria     <ETS(M,A,N)>  
## 4 American Samoa <ETS(M,A,N)>  
## 5 Andorra     <ETS(M,A,N)>  
## 6 Angola      <ETS(M,A,N)>  
## 7 Antigua and Barbuda <ETS(M,A,N)>  
## 8 Arab World  <ETS(M,A,N)>  
## 9 Argentina  <ETS(A,A,N)>  
## 10 Armenia    <ETS(M,A,N)>  
## # ... with 253 more rows
```

# Example: National populations

```
fit %>%  
  forecast(h = 5)
```

```
## # A tibble: 1,315 x 5 [1Y]  
## # Key:      Country, .model [263]  
##   Country      .model Year   Pop .distribution  
##   <fct>        <chr>   <dbl> <dbl> <dist>  
## 1 Afghanistan ets      2018  36.4 N(36, 0.012)  
## 2 Afghanistan ets      2019  37.3 N(37, 0.059)  
## 3 Afghanistan ets      2020  38.2 N(38, 0.164)  
## 4 Afghanistan ets      2021  39.0 N(39, 0.351)  
## 5 Afghanistan ets      2022  39.9 N(40, 0.644)  
## 6 Albania     ets      2018   2.87 N(2.9, 0.00012)  
## 7 Albania     ets      2019   2.87 N(2.9, 0.00060)  
## 8 Albania     ets      2020   2.87 N(2.9, 0.00169)  
## 9 Albania     ets      2021   2.86 N(2.9, 0.00362)
```

## ETS(A,A,A): Holt-Winters additive method

Forecast equation  $\hat{y}_{t+h|t} = \ell_t + hb_t + s_{t+h-m(k+1)}$

Observation equation  $y_t = \ell_{t-1} + b_{t-1} + s_{t-m} + \varepsilon_t$

State equations  $\ell_t = \ell_{t-1} + b_{t-1} + \alpha\varepsilon_t$

$$b_t = b_{t-1} + \beta\varepsilon_t$$

$$s_t = s_{t-m} + \gamma\varepsilon_t$$

- $k = \text{integer part of } (h - 1)/m$ .
- $\sum_i s_i \approx 0$ .
- Parameters:  $0 \leq \alpha \leq 1$ ,  $0 \leq \beta \leq 1$ ,  $0 \leq \gamma \leq 1 - \alpha$  and  $m = \text{period of seasonality (e.g. } m = 4 \text{ for quarterly data)}$ .

# ETS(M,A,M): Holt-Winters multiplicative method

Forecast equation  $\hat{y}_{t+h|t} = (\ell_t + hb_t)s_{t+h-m(k+1)}$

Observation equation  $y_t = (\ell_{t-1} + b_{t-1})s_{t-m}(1 + \varepsilon_t)$

State equations  $\ell_t = (\ell_{t-1} + b_{t-1})(1 + \alpha\varepsilon_t)$

$$b_t = b_{t-1}(1 + \beta\varepsilon_t)$$

$$s_t = s_{t-m}(1 + \gamma\varepsilon_t)$$

- $k$  is integer part of  $(h - 1)/m$ .
- $\sum_i s_i \approx m$ .
- Parameters:  $0 \leq \alpha \leq 1$ ,  $0 \leq \beta^* \leq 1$ ,  $0 \leq \gamma \leq 1 - \alpha$  and  $m$  = period of seasonality (e.g.  $m = 4$  for quarterly data).

# Example: Australian holiday tourism

```
holidays <- tourism %>%  
  filter(Purpose == "Holiday")  
fit <- holidays %>% model(ets = ETS(Trips))  
fit
```

```
## # A mable: 76 x 4  
## # Key:      Region, State, Purpose [76]  
##   Region                State                Purpose ets  
##   <chr>                 <chr>                 <chr>  <model>  
## 1 Adelaide             South Australia    Holiday <ETS(A,N,A~  
## 2 Adelaide Hills       South Australia    Holiday <ETS(A,A,N~  
## 3 Alice Springs        Northern Territo~ Holiday <ETS(M,N,A~  
## 4 Australia's Coral Coast Western Australia Holiday <ETS(M,N,A~  
## 5 Australia's Golden Outba~ Western Australia Holiday <ETS(M,N,M~  
## 6 Australia's North West Western Australia Holiday <ETS(A,N,A~  
## 7 Australia's South West Western Australia Holiday <ETS(M,N,M~  
## 8 Ballarat             Victoria          Holiday <ETS(M,N,A~  
## 9 Barkly               Northern Territo~ Holiday <ETS(A,N,A~  
## 10 Barossa             South Australia    Holiday <ETS(A,N,N~
```

# Example: Australian holiday tourism

```
fit %>% filter(Region=="Snowy Mountains") %>% report()
```

```
## Series: Trips
## Model: ETS(M,N,A)
## Smoothing parameters:
##   alpha = 0.157
##   gamma = 1e-04
##
## Initial states:
##   l   s1  s2    s3    s4
## 142 -61 131 -42.2 -27.7
##
## sigma^2: 0.0388
##
## AIC AICc BIC
## 852 854 869
```

# Example: Australian holiday tourism

```
fit %>% filter(Region=="Snowy Mountains") %>% components(fit)
```

```
## # A dable:                84 x 9 [1Q]
## # Key:                    Region, State, Purpose, .model [1]
## # ETS(M,N,A) Decomposition: Trips = (lag(level, 1) + lag(season,
## #   4)) * (1 + remainder)
##   Region State Purpose .model   Quarter Trips level season
##   <chr>  <chr> <chr>  <chr>      <qtr>  <dbl>  <dbl>  <dbl>
## 1 Snowy~ New ~ Holiday ets      1997 Q1   NA      NA    -27.7
## 2 Snowy~ New ~ Holiday ets      1997 Q2   NA      NA    -42.2
## 3 Snowy~ New ~ Holiday ets      1997 Q3   NA      NA    131.
## 4 Snowy~ New ~ Holiday ets      1997 Q4   NA     142.   -61.0
## 5 Snowy~ New ~ Holiday ets      1998 Q1  101.    140.   -27.7
## 6 Snowy~ New ~ Holiday ets      1998 Q2  112.    142.   -42.2
## 7 Snowy~ New ~ Holiday ets      1998 Q3  310.    148.    131.
## 8 Snowy~ New ~ Holiday ets      1998 Q4   89.8   148.   -61.0
## 9 Snowy~ New ~ Holiday ets      1999 Q1  112.    147.   -27.7
## 10 Snowy~ New ~ Holiday ets      1999 Q2  103.    147.   -42.2
## # ...with 74 more rows, and 1 more variable: remainder <dbl>
```

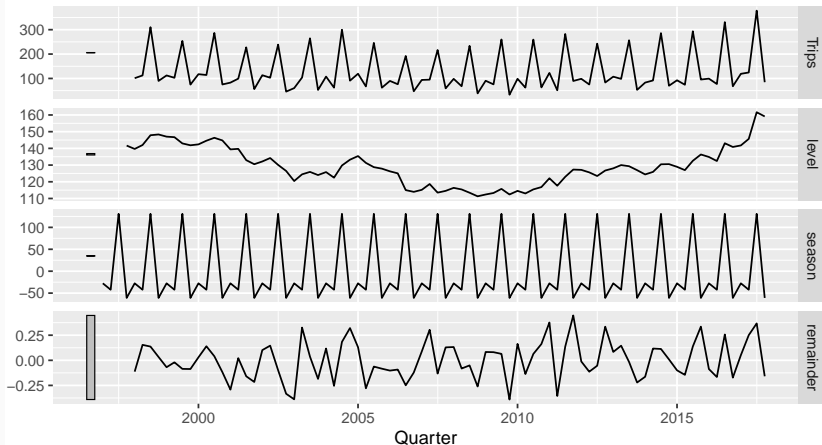


# Example: Australian holiday tourism

```
fit %>% filter(Region=="Snowy Mountains") %>%  
  components(fit) %>% autoplot()
```

ETS(M,N,A) decomposition

Trips = (lag(level, 1) + lag(season, 4)) \* (1 + remainder)



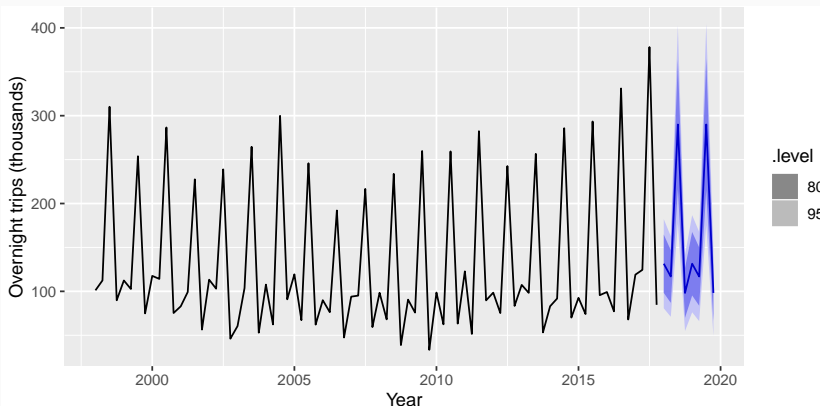
# Example: Australian holiday tourism

```
fit %>% forecast()
```

```
## # A tibble: 608 x 7 [1Q]
## # Key:      Region, State, Purpose, .model [76]
##   Region    State    Purpose .model    Quarter Trips .distribution
##   <chr>     <chr>     <chr>  <chr>      <qtr>  <dbl>  <dist>
## 1 Adelaide South A~ Holiday ets      2018 Q1  210.  N(210, 457)
## 2 Adelaide South A~ Holiday ets      2018 Q2  173.  N(173, 473)
## 3 Adelaide South A~ Holiday ets      2018 Q3  169.  N(169, 489)
## 4 Adelaide South A~ Holiday ets      2018 Q4  186.  N(186, 505)
## 5 Adelaide South A~ Holiday ets      2019 Q1  210.  N(210, 521)
## 6 Adelaide South A~ Holiday ets      2019 Q2  173.  N(173, 537)
## 7 Adelaide South A~ Holiday ets      2019 Q3  169.  N(169, 553)
## 8 Adelaide South A~ Holiday ets      2019 Q4  186.  N(186, 569)
## 9 Adelaide~ South A~ Holiday ets      2018 Q1   19.4 N(19, 36)
##10 Adelaide~ South A~ Holiday ets      2018 Q2   19.6 N(20, 36)
## # ... with 598 more rows
```

# Example: Australian holiday tourism

```
fit %>% forecast() %>%  
  filter(Region=="Snowy Mountains") %>%  
  autoplot(holidays) +  
    xlab("Year") + ylab("Overnight trips (thousands)")
```



# Exponential smoothing models

## Additive Error

		Seasonal Component		
		N (None)	A (Additive)	M (Multiplicative)
Trend Component	N (None)	A,N,N	A,N,A	<del>A,N,M</del>
	A (Additive)	A,A,N	A,A,A	<del>A,A,M</del>
	A <sub>d</sub> (Additive damped)	A,A <sub>d</sub> ,N	A,A <sub>d</sub> ,A	<del>A,A<sub>d</sub>,M</del>

## Multiplicative Error

		Seasonal Component		
		N (None)	A (Additive)	M (Multiplicative)
Trend Component	N (None)	M,N,N	M,N,A	M,N,M
	A (Additive)	M,A,N	M,A,A	M,A,M
	A <sub>d</sub> (Additive damped)	M,A <sub>d</sub> ,N	M,A <sub>d</sub> ,A	M,A <sub>d</sub> ,M

# Estimating ETS models

- Smoothing parameters  $\alpha$ ,  $\beta$ ,  $\gamma$  and  $\phi$ , and the initial states  $\ell_0$ ,  $b_0$ ,  $s_0$ ,  $s_{-1}$ ,  $\dots$ ,  $s_{-m+1}$  are estimated by maximising the “likelihood” = the probability of the data arising from the specified model.
- For models with additive errors equivalent to minimising SSE.
- For models with multiplicative errors, **not** equivalent to minimising SSE.

# Model selection

## Akaike's Information Criterion

$$\text{AIC} = -2 \log(L) + 2k$$

where  $L$  is the likelihood and  $k$  is the number of parameters initial states estimated in the model.

# Model selection

## Akaike's Information Criterion

$$\text{AIC} = -2 \log(L) + 2k$$

where  $L$  is the likelihood and  $k$  is the number of parameters initial states estimated in the model.

## Corrected AIC

$$\text{AIC}_c = \text{AIC} + \frac{2(k+1)(k+2)}{T-k}$$

which is the AIC corrected (for small sample bias).

# Model selection

## Akaike's Information Criterion

$$\text{AIC} = -2 \log(L) + 2k$$

where  $L$  is the likelihood and  $k$  is the number of parameters initial states estimated in the model.

## Corrected AIC

$$\text{AIC}_c = \text{AIC} + \frac{2(k+1)(k+2)}{T-k}$$

which is the AIC corrected (for small sample bias).

## Bayesian Information Criterion

$$\text{BIC} = \text{AIC} + k(\log(T) - 2).$$



# AIC and cross-validation

Minimizing the AIC assuming Gaussian residuals is asymptotically equivalent to minimizing one-step time series cross validation MSE.

# Automatic forecasting

## From Hyndman et al. (IJF, 2002):

- 1 Apply each model that is appropriate to the data. Optimize parameters and initial values using MLE.
  - 2 Select best method using AICc.
  - 3 Produce forecasts using best method.
  - 4 Obtain forecast intervals using underlying state space model.
- Method performed very well in M3 competition.
  - Used as a benchmark in the M4 competition.

# Outline

- 1 Exponential smoothing
- 2 Lab Session 7
- 3 ARIMA models
- 4 Lab Session 8
- 5 Seasonal ARIMA models
- 6 Lab Session 9
- 7 Forecast accuracy measures

Find an ETS model for the Gas data from `aus_production`.

- Why is multiplicative seasonality necessary here?
- Experiment with making the trend damped.

# Outline

- 1 Exponential smoothing
- 2 Lab Session 7
- 3 ARIMA models
- 4 Lab Session 8
- 5 Seasonal ARIMA models
- 6 Lab Session 9
- 7 Forecast accuracy measures

# ARIMA models

- AR:** autoregressive (lagged observations as inputs)
- I:** integrated (differencing to make series stationary)
- MA:** moving average (lagged errors as inputs)

# ARIMA models

**AR:** autoregressive (lagged observations as inputs)

**I:** integrated (differencing to make series stationary)

**MA:** moving average (lagged errors as inputs)

An ARIMA model is rarely interpretable in terms of visible data structures like trend and seasonality. But it can capture a huge range of time series patterns.

# 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  $t$ .



# Stationarity

## Definition

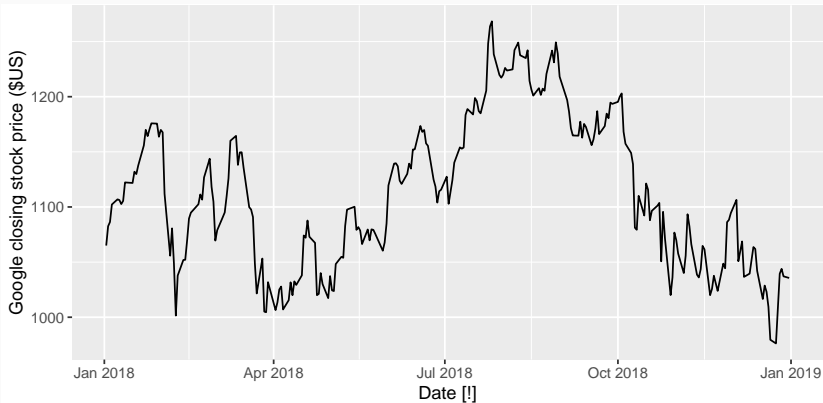
If  $\{y_t\}$  is a stationary time series, then for all  $s$ , the distribution of  $(y_t, \dots, y_{t+s})$  does not depend on  $t$ .

A **stationary series** is:

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

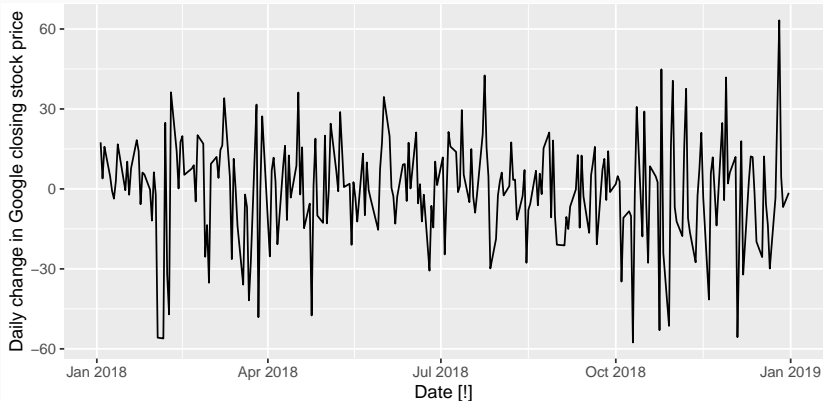
# Stationary?

```
gafa_stock %>%  
  filter(Symbol == "GOOG", year(Date) == 2018) %>%  
  autoplot(Close) +  
    ylab("Google closing stock price ($US)")
```



# Stationary?

```
gafa_stock %>%  
  filter(Symbol == "GOOG", year(Date) == 2018) %>%  
  autoplot(difference(Close)) +  
  ylab("Daily change in Google closing stock price")
```



# Differencing

- Differencing helps to **stabilize the mean**.
- The differenced series is the *change* between each observation in the original series.
- Occasionally the differenced data will not appear stationary and it may be necessary to difference the data a second time.
- In practice, it is almost never necessary to go beyond second-order differences.

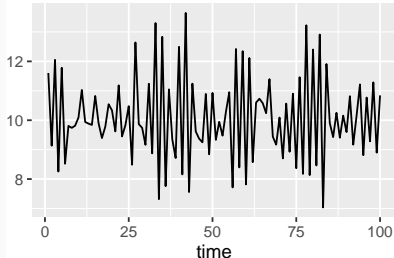
# Autoregressive models

## Autoregressive (AR) models:

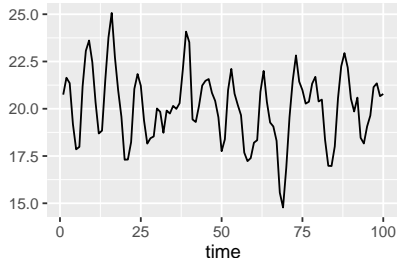
$$y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \varepsilon_t,$$

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

AR(1)



AR(2)



- Cyclic behaviour is possible when  $p \geq 2$ .

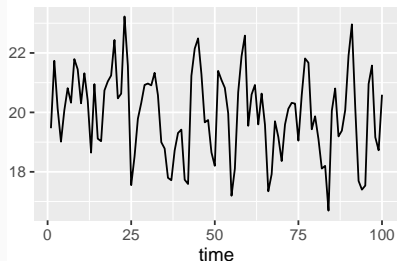
# Moving Average (MA) models

## Moving Average (MA) models:

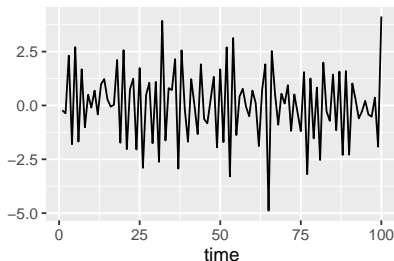
$$y_t = c + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \cdots + \theta_q \varepsilon_{t-q},$$

where  $\varepsilon_t$  is white noise. This is a multiple regression with **lagged errors** as predictors. *Don't confuse this with moving average smoothing!*

MA(1)



MA(2)



# ARIMA models

## Autoregressive Moving Average models:

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

# ARIMA 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**.



# ARIMA 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**.

## Autoregressive Integrated Moving Average models

- Combine ARMA model with **differencing**.
- $d$ -differenced series follows an ARMA model.
- Need to choose  $p, d, q$  and whether or not to include  $c$ .

# ARIMA 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, $q$ )

# Example: National populations

```
fit <- global_economy %>%  
  model(arima = ARIMA(Population))  
fit
```

```
## # A mable: 263 x 2  
## # Key:      Country [263]  
##   Country      arima  
##   <fct>        <model>  
## 1 Afghanistan <ARIMA(4,2,1)>  
## 2 Albania     <ARIMA(0,2,2)>  
## 3 Algeria     <ARIMA(2,2,2)>  
## 4 American Samoa <ARIMA(2,2,2)>  
## 5 Andorra     <ARIMA(2,1,2) w/ drift>  
## 6 Angola      <ARIMA(4,2,1)>  
## 7 Antigua and Barbuda <ARIMA(2,1,2) w/ drift>  
## 8 Arab World  <ARIMA(0,2,1)>
```

# Example: National populations

```
fit %>% filter(Country=="Australia") %>% report()
```

```
## Series: Population
## Model: ARIMA(0,2,1)
##
## Coefficients:
##          ma1
##      -0.661
## s.e.    0.107
##
## sigma^2 estimated as 4.063e+09:  log likelihood=-699
## AIC=1401   AICc=1402   BIC=1405
```

# Example: National populations

```
fit %>% filter(Country=="Australia") %>% report()
```

```
## Series: Population
```

```
## Model: ARIMA(0,2,1)
```

```
##
```

```
## Coefficients:
```

```
##          ma1
```

```
##        -0.661
```

```
## s.e.    0.107
```

```
##
```

```
## sigma^2 estimated as 4.063e+09:  log likelihood=-699
```

```
## AIC=1401   AICc=1402   BIC=1405
```

$$y_t = 2y_{t-1} - y_{t-2} - 0.7\varepsilon_{t-1} + \varepsilon_t$$
$$\varepsilon_t \sim \text{NID}(0, 4 \times 10^9)$$

# Understanding ARIMA models

- If  $c = 0$  and  $d = 0$ , the long-term forecasts will go to zero.
- If  $c = 0$  and  $d = 1$ , the long-term forecasts will go to a non-zero constant.
- If  $c = 0$  and  $d = 2$ , the long-term forecasts will follow a straight line.
- If  $c \neq 0$  and  $d = 0$ , the long-term forecasts will go to the mean of the data.
- If  $c \neq 0$  and  $d = 1$ , the long-term forecasts will follow a straight line.
- If  $c \neq 0$  and  $d = 2$ , the long-term forecasts will follow a quadratic trend.

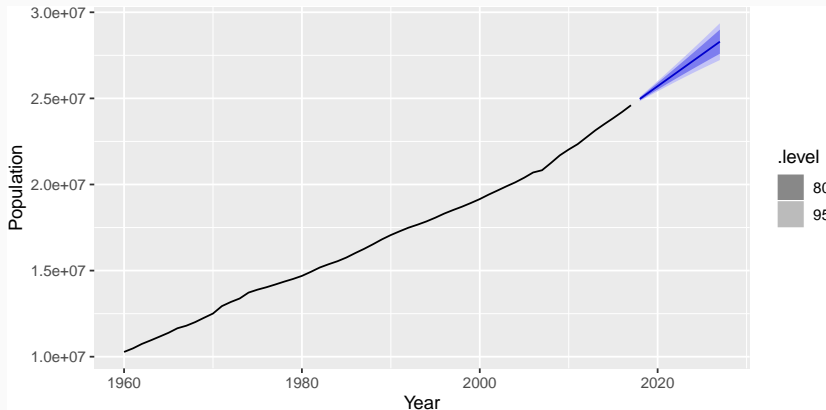
# Understanding ARIMA models

## Forecast variance and $d$

- The higher the value of  $d$ , the more rapidly the prediction intervals increase in size.
- For  $d = 0$ , the long-term forecast standard deviation will go to the standard deviation of the historical data.

# Example: National populations

```
fit %>% forecast(h=10) %>%  
  filter(Country=="Australia") %>%  
  autoplot(global_economy)
```





# How does ARIMA() work?

## Hyndman and Khandakar (JSS, 2008) algorithm:

- Select no. differences  $d$  via KPSS test.
- Select  $p, q$  and inclusion of  $c$  by minimising AICc.
- Use stepwise search to traverse model space.

# How does ARIMA() work?

## Hyndman and Khandakar (JSS, 2008) algorithm:

- Select no. differences  $d$  via KPSS test.
- Select  $p, q$  and inclusion of  $c$  by minimising AICc.
- Use stepwise search to traverse model space.

$$\text{AICc} = -2 \log(L) + 2(p+q+k+1) \left[ 1 + \frac{(p+q+k+2)}{T-p-q-k-2} \right].$$

where  $L$  is the maximised likelihood fitted to the *differenced* data,  $k = 1$  if  $c \neq 0$  and  $k = 0$  otherwise.

# How does ARIMA() work?

## Hyndman and Khandakar (JSS, 2008) algorithm:

- Select no. differences  $d$  via KPSS test.
- Select  $p$ ,  $q$  and inclusion of  $c$  by minimising AICc.
- Use stepwise search to traverse model space.

$$\text{AICc} = -2 \log(L) + 2(p+q+k+1) \left[ 1 + \frac{(p+q+k+2)}{T-p-q-k-2} \right].$$

where  $L$  is the maximised likelihood fitted to the *differenced* data,  $k = 1$  if  $c \neq 0$  and  $k = 0$  otherwise.

Note: Can't compare AICc for different values of  $d$ .

# How does ARIMA() work?

**Step1:** Select current model (with smallest AICc) from:

ARIMA(2,  $d$ , 2)

ARIMA(0,  $d$ , 0)

ARIMA(1,  $d$ , 0)

ARIMA(0,  $d$ , 1)

# How does ARIMA() work?

**Step 1:** Select current model (with smallest AICc) from:

ARIMA(2,  $d$ , 2)

ARIMA(0,  $d$ , 0)

ARIMA(1,  $d$ , 0)

ARIMA(0,  $d$ , 1)

**Step 2:** Consider variations of current model:

- vary one of  $p$ ,  $q$ , from current model by  $\pm 1$ ;
- $p$ ,  $q$  both vary from current model by  $\pm 1$ ;
- Include/exclude  $c$  from current model.

Model with lowest AICc becomes current model.

# How does ARIMA() work?

**Step1:** Select current model (with smallest AICc) from:

ARIMA(2,  $d$ , 2)

ARIMA(0,  $d$ , 0)

ARIMA(1,  $d$ , 0)

ARIMA(0,  $d$ , 1)

**Step 2:** Consider variations of current model:

- vary one of  $p$ ,  $q$ , from current model by  $\pm 1$ ;
- $p$ ,  $q$  both vary from current model by  $\pm 1$ ;
- Include/exclude  $c$  from current model.

Model with lowest AICc becomes current model.

**Repeat Step 2 until no lower AICc can be found.**

# Outline

- 1 Exponential smoothing
- 2 Lab Session 7
- 3 ARIMA models
- 4 Lab Session 8
- 5 Seasonal ARIMA models
- 6 Lab Session 9
- 7 Forecast accuracy measures

For the United States GDP data (from `global_economy`):

- Fit a suitable ARIMA model for the logged data.
- Produce forecasts of your fitted model. Do the forecasts look reasonable?

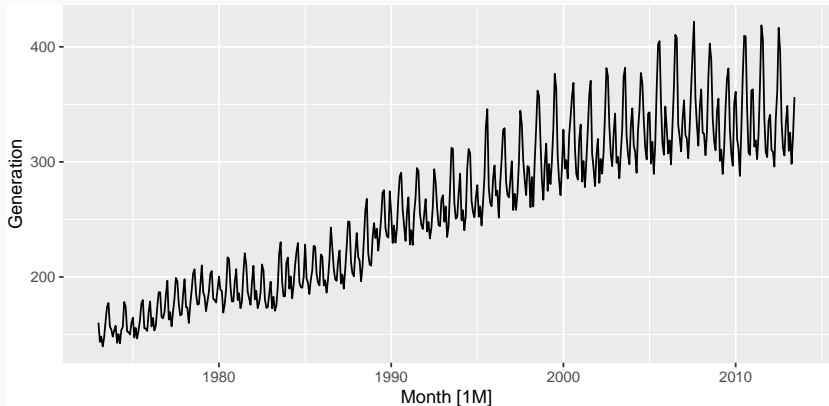


# Outline

- 1 Exponential smoothing
- 2 Lab Session 7
- 3 ARIMA models
- 4 Lab Session 8
- 5 Seasonal ARIMA models**
- 6 Lab Session 9
- 7 Forecast accuracy measures

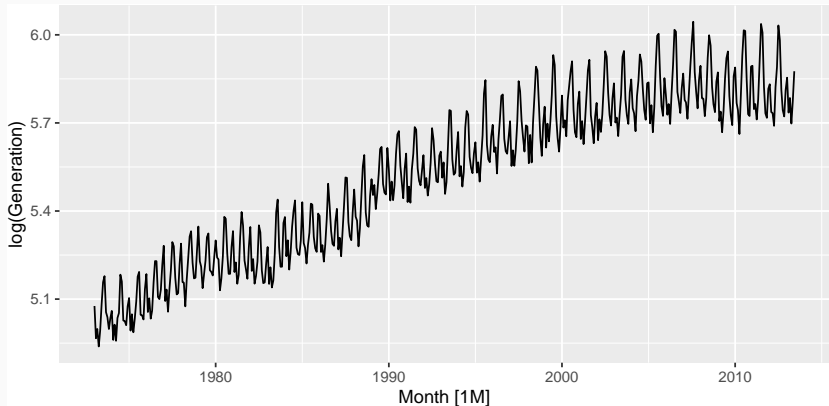
# Electricity production

```
usmelec %>% autoplot(  
  Generation  
)
```



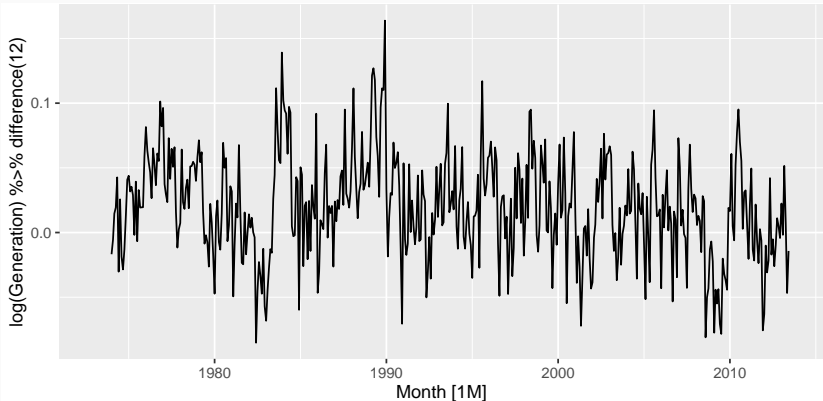
# Electricity production

```
usmelec %>% autoplot(  
  log(Generation)  
)
```



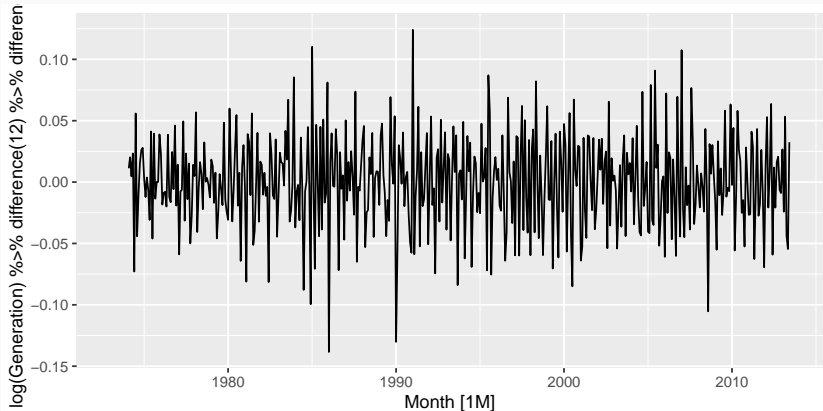
# Electricity production

```
usmelec %>% autoplot(  
  log(Generation) %>% difference(12)  
)
```



# Electricity production

```
usmelec %>% autoplot(  
  log(Generation) %>% difference(12) %>% difference()  
)
```



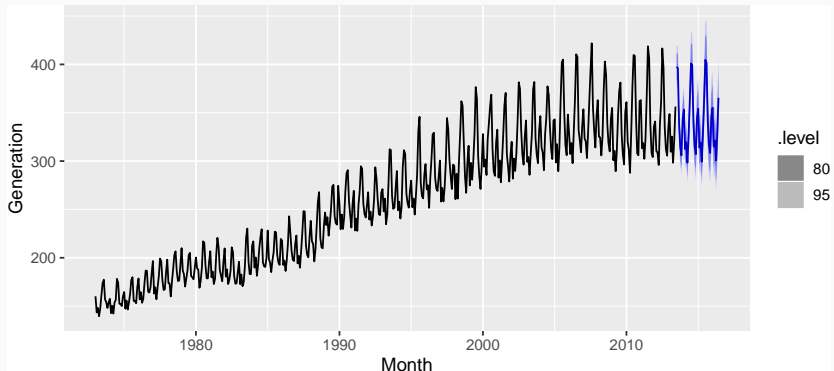
# Example: US electricity production

```
usmelec %>%  
  model(arima = ARIMA(log(Generation))) %>%  
  report()
```

```
## Series: Generation  
## Model: ARIMA(1,1,1)(2,1,1)[12]  
## Transformation: log(.x)  
##  
## Coefficients:  
##          ar1          ma1          sar1          sar2          sma1  
##      0.4116   -0.8483    0.0100   -0.1017   -0.8204  
## s.e.  0.0617    0.0348    0.0561    0.0529    0.0357  
##  
## sigma^2 estimated as 0.0006841:  log likelihood=1047  
## AIC=-2082    AICc=-2082    BIC=-2057
```

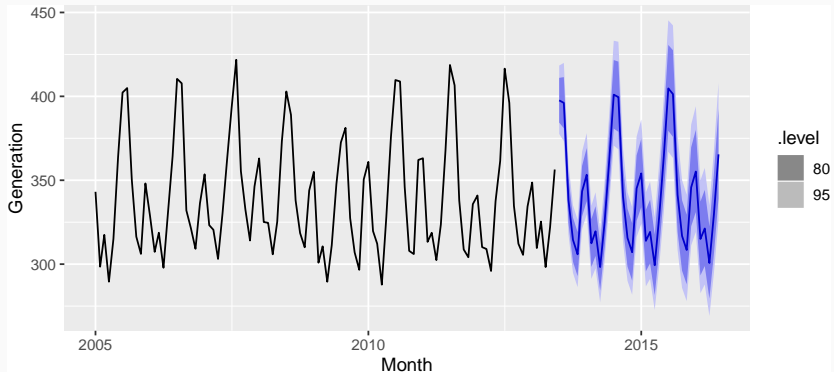
# Example: US electricity production

```
usmelec %>%  
  model(arima = ARIMA(log(Generation))) %>%  
  forecast(h="3 years") %>%  
  autoplot(usmelec)
```



# Example: US electricity production

```
usmelec %>%  
  model(arima = ARIMA(log(Generation))) %>%  
  forecast(h="3 years") %>%  
  autoplot(filter_index(usmelec, 2005 ~ .))
```





# Seasonal ARIMA models

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

- $m$  = number of observations per year.
- $d$  first differences,  $D$  seasonal differences
- $p$  AR lags,  $q$  MA lags
- $P$  seasonal AR lags,  $Q$  seasonal MA lags

Seasonal and non-seasonal terms combine multiplicatively

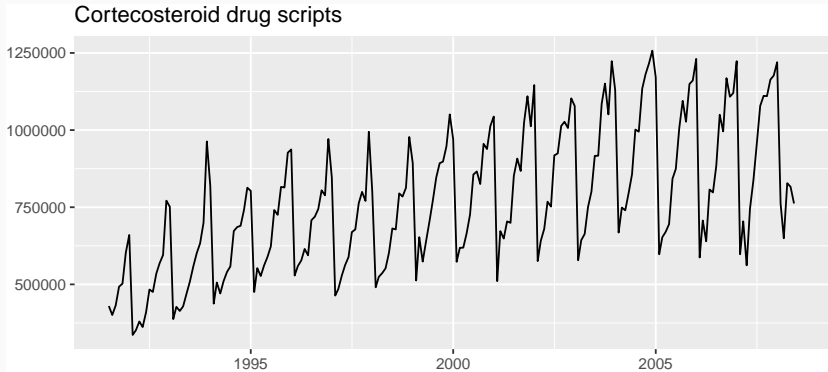
# Common ARIMA models

The US Census Bureau uses the following models most often:

ARIMA(0,1,1)(0,1,1) <sub>m</sub>	with log transformation
ARIMA(0,1,2)(0,1,1) <sub>m</sub>	with log transformation
ARIMA(2,1,0)(0,1,1) <sub>m</sub>	with log transformation
ARIMA(0,2,2)(0,1,1) <sub>m</sub>	with log transformation
ARIMA(2,1,2)(0,1,1) <sub>m</sub>	with no transformation

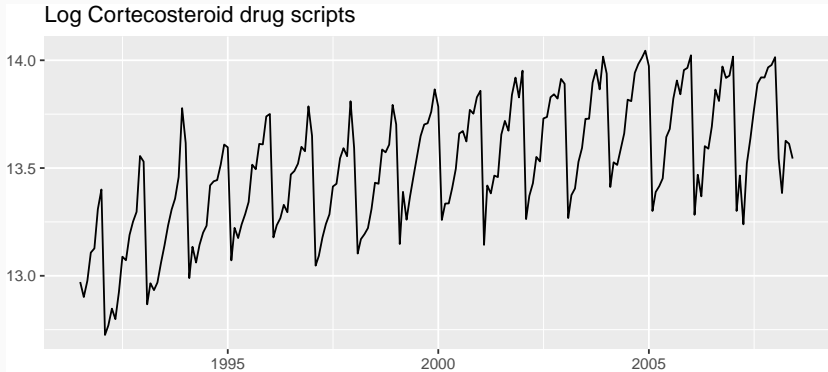
# Corticosteroid drug sales

```
h02 <- PBS %>% filter(ATC2 == "H02") %>%  
  summarise(Cost = sum(Cost))  
h02 %>% autoplot(Cost) +  
  xlab("Year") + ylab("") +  
  ggtitle("Corticosteroid drug scripts")
```



# Corticosteroid drug sales

```
h02 <- PBS %>% filter(ATC2 == "H02") %>%  
  summarise(Cost = sum(Cost))  
h02 %>% autoplot(log(Cost)) +  
  xlab("Year") + ylab("") +  
  ggtitle("Log Corticosteroid drug scripts")
```



# Corticosteroid drug sales

```
fit <- h02 %>%  
  model(auto = ARIMA(log(Cost)))  
report(fit)
```

```
## Series: Cost  
## Model: ARIMA(2,1,0)(0,1,1)[12]  
## Transformation: log(.x)  
##  
## Coefficients:  
##          ar1      ar2      sma1  
##      -0.8491  -0.4207  -0.6401  
## s.e.   0.0712   0.0714   0.0694  
##  
## sigma^2 estimated as 0.004399:  log likelihood=245  
## AIC=-483   AICc=-483   BIC=-470
```

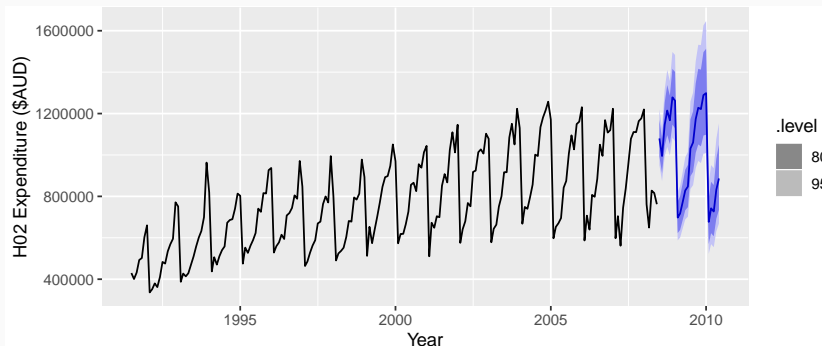
# Corticosteroid drug sales

```
fit <- h02 %>%  
  model(best = ARIMA(log(Cost), stepwise = FALSE,  
    approximation = FALSE,  
    order_constraint = p + q + P + Q <= 9))  
report(fit)
```

```
## Series: Cost  
## Model: ARIMA(4,1,1)(2,1,2)[12]  
## Transformation: log(.x)  
##  
## Coefficients:  
##          ar1      ar2      ar3      ar4      ma1      sar1      sar2  
##      -0.0426  0.210   0.202  -0.227  -0.742   0.621  -0.383  
## s.e.   0.2167  0.181   0.114   0.081   0.207   0.242   0.118  
##          sma1      sma2  
##      -1.202   0.496  
## s.e.   0.249   0.214  
##  
## sigma^2 estimated as 0.004061:  log likelihood=254  
## AIC=-489   AICc=-487   BIC=-456
```

# Corticosteroid drug sales

```
fit %>% forecast %>% autoplot(h02) +  
  ylab("H02 Expenditure ($AUD)") + xlab("Year")
```



# Outline

- 1 Exponential smoothing
- 2 Lab Session 7
- 3 ARIMA models
- 4 Lab Session 8
- 5 Seasonal ARIMA models
- 6 Lab Session 9
- 7 Forecast accuracy measures



For the Australian tourism data (from `tourism`):

- Fit a suitable ARIMA model for all data.
- Produce forecasts of your fitted models.
- Check the forecasts for the “Snowy Mountains” and “Melbourne” regions. Do they look reasonable?

# Outline

- 1 Exponential smoothing
- 2 Lab Session 7
- 3 ARIMA models
- 4 Lab Session 8
- 5 Seasonal ARIMA models
- 6 Lab Session 9
- 7 Forecast accuracy measures

# Training and test sets



- A model which fits the training data well will not necessarily forecast well.
- Forecast accuracy is based only on the test set.

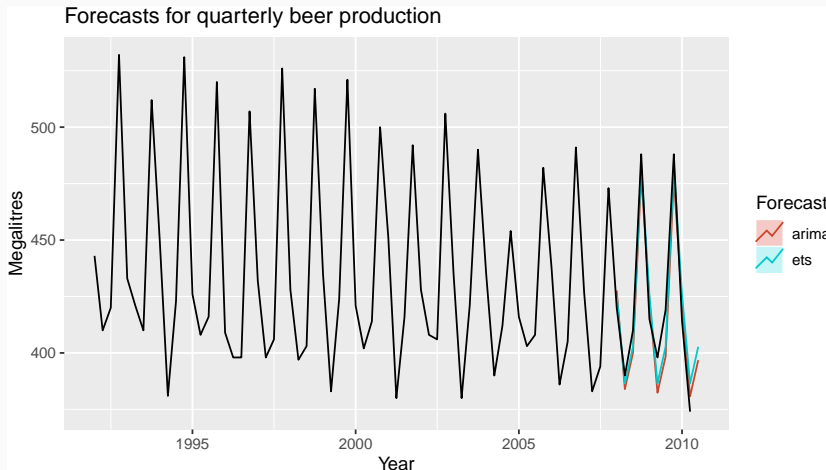
## Forecast errors

Forecast “error”: the difference between an observed value and its forecast.

$$e_{T+h} = y_{T+h} - \hat{y}_{T+h|T},$$

where the training data is given by  $\{y_1, \dots, y_T\}$

# Measures of forecast accuracy



# Measures of forecast accuracy

$y_{T+h}$  =  $(T + h)$ th observation,  $h = 1, \dots, H$

$\hat{y}_{T+h|T}$  = its forecast based on data up to time  $T$ .

$$e_{T+h} = y_{T+h} - \hat{y}_{T+h|T}$$

$$\text{MAE} = \text{mean}(|e_{T+h}|)$$

$$\text{MSE} = \text{mean}(e_{T+h}^2)$$

$$\text{RMSE} = \sqrt{\text{mean}(e_{T+h}^2)}$$

$$\text{MAPE} = 100\text{mean}(|e_{T+h}|/|y_{T+h}|)$$

# Measures of forecast accuracy

$y_{T+h}$  =  $(T + h)$ th observation,  $h = 1, \dots, H$

$\hat{y}_{T+h|T}$  = its forecast based on data up to time  $T$ .

$$e_{T+h} = y_{T+h} - \hat{y}_{T+h|T}$$

$$\text{MAE} = \text{mean}(|e_{T+h}|)$$

$$\text{MSE} = \text{mean}(e_{T+h}^2)$$

$$\text{RMSE} = \sqrt{\text{mean}(e_{T+h}^2)}$$

$$\text{MAPE} = 100\text{mean}(|e_{T+h}|/|y_{T+h}|)$$

- MAE, MSE, RMSE are all scale dependent.
- MAPE is scale independent but is only sensible if  $y_t \gg 0$  for all  $t$ , and  $y$  has a natural zero.

# Measures of forecast accuracy

## Mean Absolute Scaled Error

$$\text{MASE} = \text{mean}(|e_{T+h}|/Q)$$

where  $Q$  is a stable measure of the scale of the time series  $\{y_t\}$ .

Proposed by Hyndman and Koehler (IJF, 2006).

For non-seasonal time series,

$$Q = (T - 1)^{-1} \sum_{t=2}^T |y_t - y_{t-1}|$$

works well. Then MASE is equivalent to MAE relative to a naïve method.

# Measures of forecast accuracy

## Mean Absolute Scaled Error

$$\text{MASE} = \text{mean}(|e_{T+h}|/Q)$$

where  $Q$  is a stable measure of the scale of the time series  $\{y_t\}$ .

Proposed by Hyndman and Koehler (IJF, 2006).

For seasonal time series,

$$Q = (T - m)^{-1} \sum_{t=m+1}^T |y_t - y_{t-m}|$$

works well. Then MASE is equivalent to MAE relative to a seasonal naïve method.



# Measures of forecast accuracy

```
recent_production <- aus_production %>%  
  filter(year(Quarter) >= 1992)  
train <- recent_production %>% filter(year(Quarter) <= 2007)  
beer_fit <- train %>%  
  model(  
    ets = ETS(Beer),  
    arima = ARIMA(Beer)  
  )  
beer_fc <- forecast(beer_fit, h="4 years")  
accuracy(beer_fc, aus_production)
```

```
## # A tibble: 2 x 9  
##   .model .type    ME  RMSE   MAE   MPE  MAPE  MASE  ACF1  
##   <chr>  <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 arima Test   4.18  11.2  10.4  0.940  2.47  0.657  0.145  
## 2 ets   Test   0.854  9.80  8.99  0.151  2.18  0.568  0.207
```