

# Tidy Time Series & Forecasting in R

## 8. ARIMA models



# Outline

- 1 ARIMA models
- 2 Lab Session 16
- 3 Seasonal ARIMA models
- 4 Lab Session 17
- 5 Forecast ensembles

# Outline

- 1 ARIMA models
- 2 Lab Session 16
- 3 Seasonal ARIMA models
- 4 Lab Session 17
- 5 Forecast ensembles

# 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

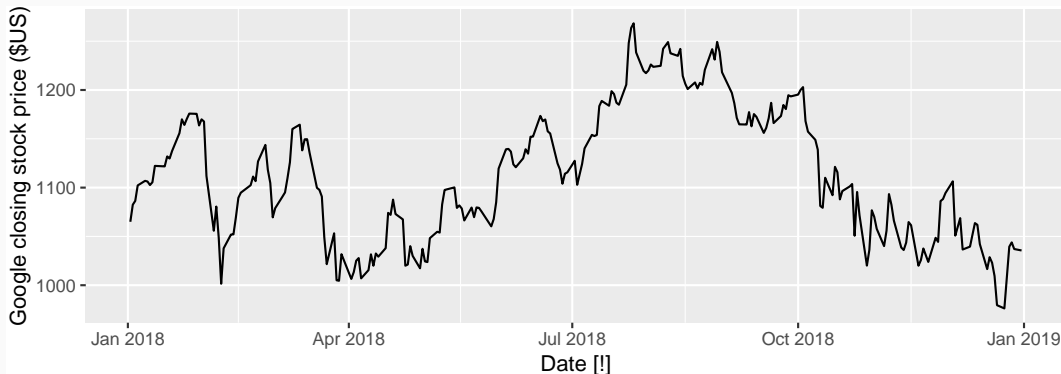
If  $\{y_t\}$  is a stationary time series, then for all  $s$ , the distribution of  $(y_t, \dots, y_{t+s})$  does not depend on  $t$ .

A **stationary series** is:

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

# Stationary?

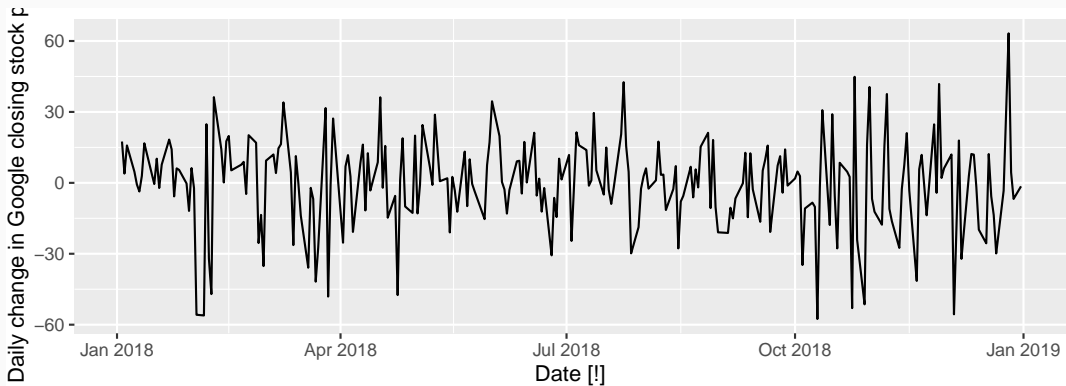
```
gafa_stock %>%  
  filter(Symbol == "GOOG", year(Date) == 2018) %>%  
  autoplot(Close) +  
  ylab("Google closing stock price ($US)")
```





# 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.

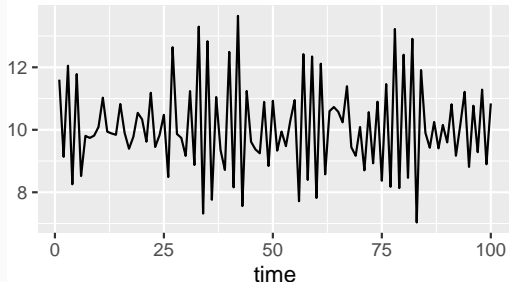
# Autoregressive models

## Autoregressive (AR) models:

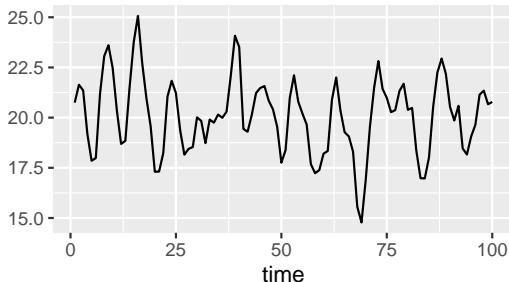
$$y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \varepsilon_t,$$

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

AR(1)



AR(2)



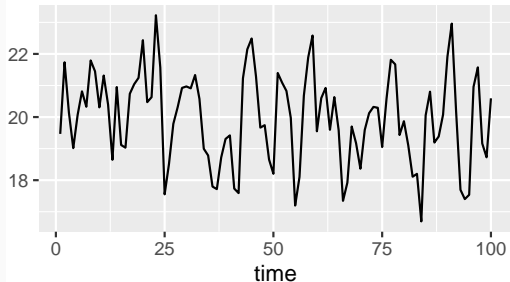
# Moving Average (MA) models

## Moving Average (MA) models:

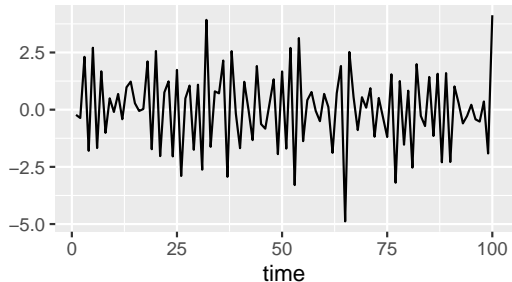
$$y_t = c + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \cdots + \theta_q \varepsilon_{t-q},$$

where  $\varepsilon_t$  is white noise. This is multiple regression with **lagged errors** as predictors. *Don't confuse 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.

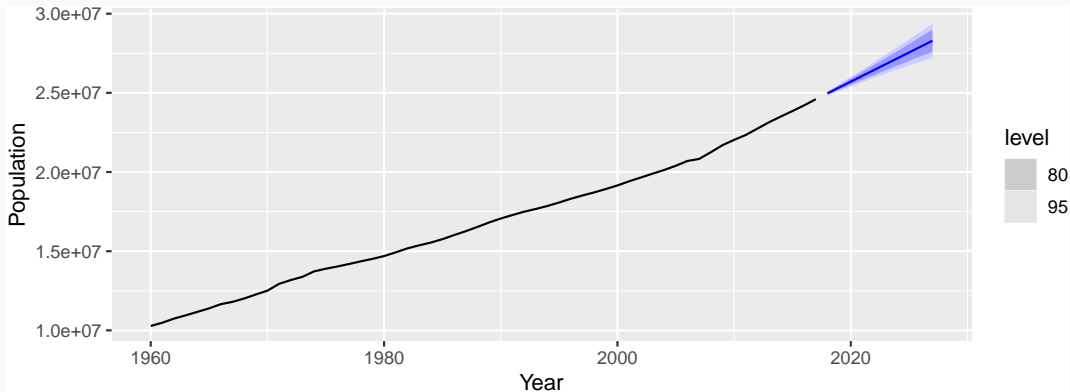
# Understanding ARIMA models

## 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?

**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.

# 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 ARIMA models
- 2 Lab Session 16
- 3 Seasonal ARIMA models
- 4 Lab Session 17
- 5 Forecast ensembles

## Lab Session 16

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

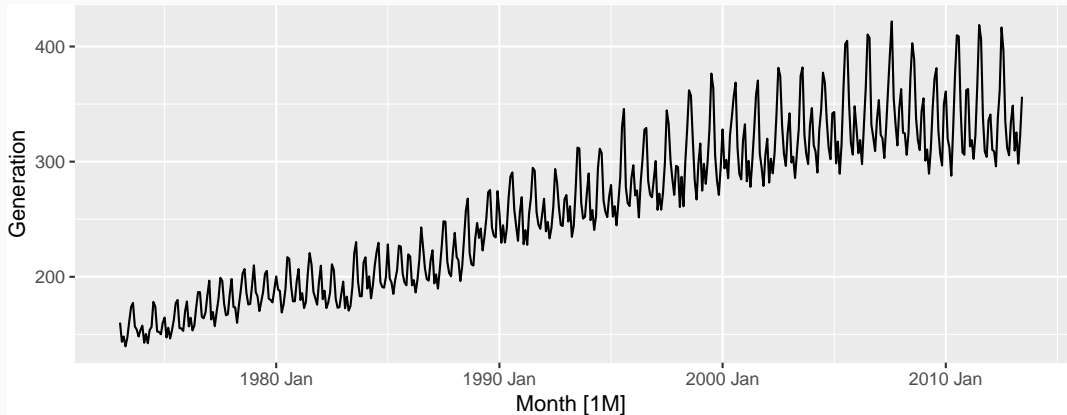
- Fit a suitable ARIMA model for the logged data.
- Produce forecasts of your fitted model. Do the forecasts look reasonable?

# Outline

- 1 ARIMA models
- 2 Lab Session 16
- 3 Seasonal ARIMA models**
- 4 Lab Session 17
- 5 Forecast ensembles

# Electricity production

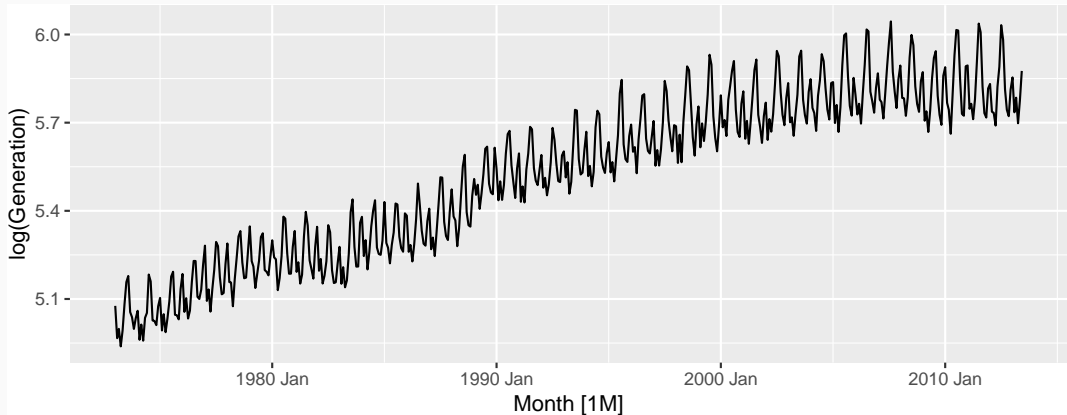
```
usmelec %>% autoplot(  
  Generation  
)
```





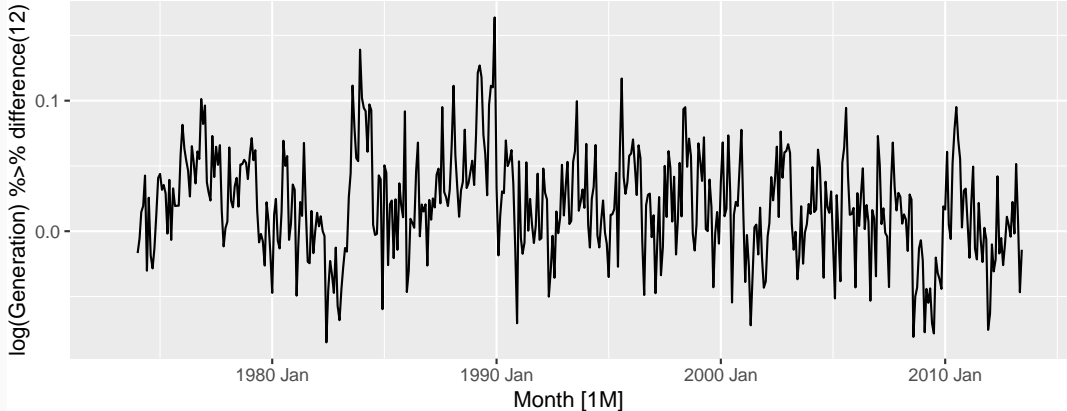
# Electricity production

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



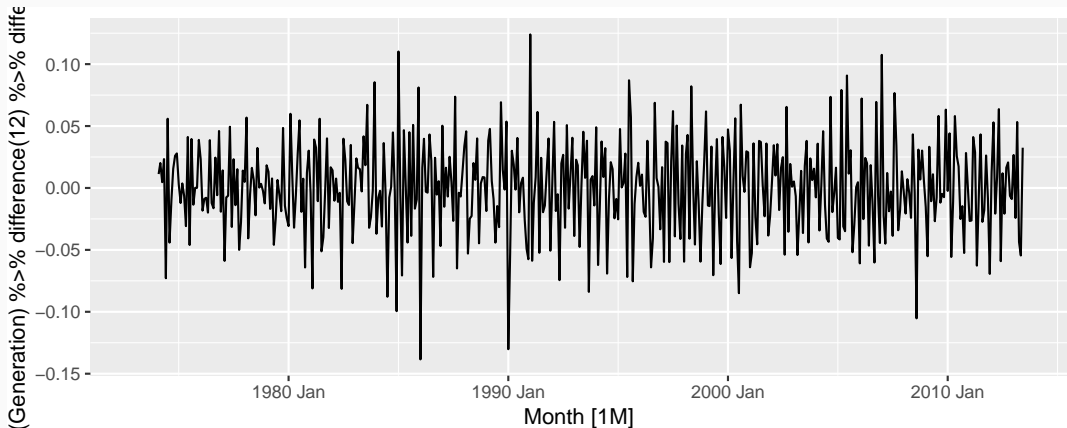
# Electricity production

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



# Electricity production

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



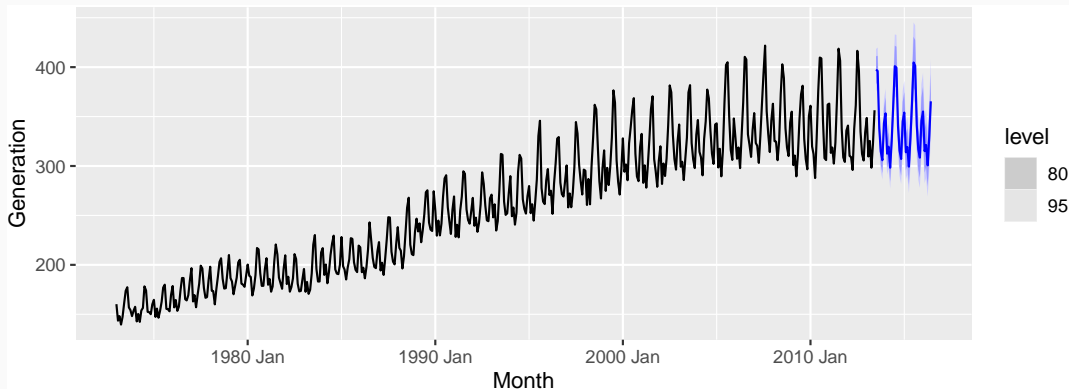
# Example: US electricity production

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

```
## Series: Generation  
## Model: ARIMA(1,1,1)(2,1,1)[12]  
## Transformation: log(Generation)  
##  
## 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
```

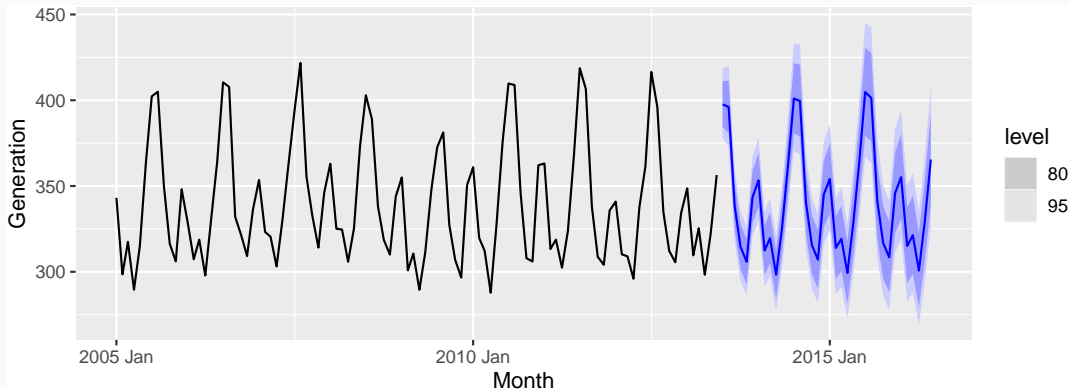
# Example: US electricity production

```
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	$(p, d, q)$	$(P, D, Q)_m$
	↑	↑
	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

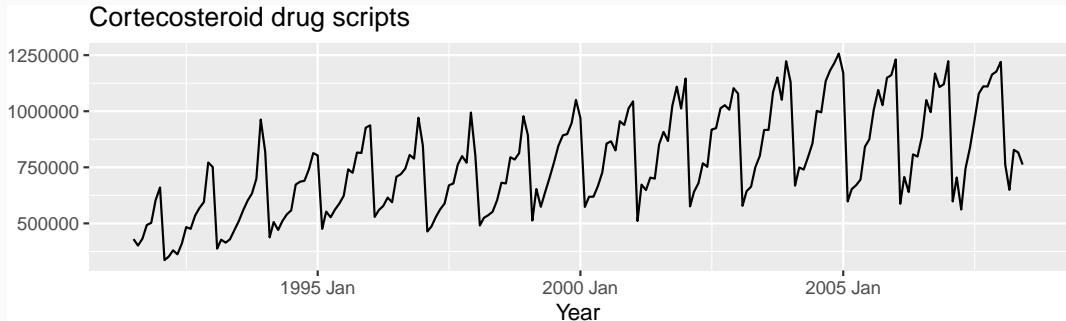
ARIMA(0,2,2)(0,1,1)<sub>m</sub> with log transformation

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



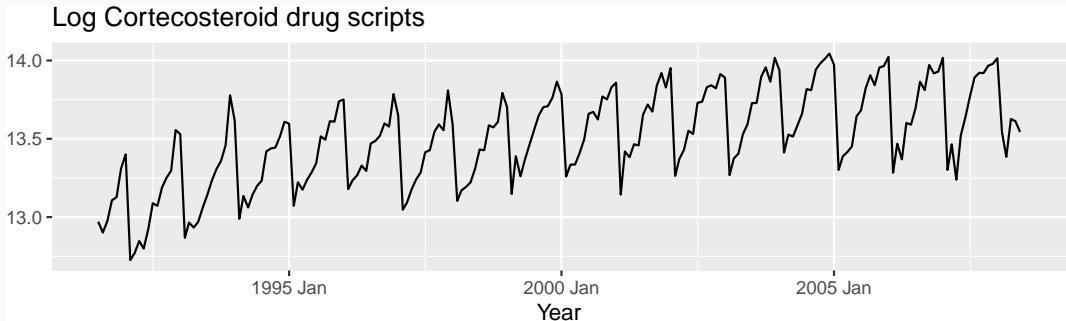
# Corticosteroid drug sales

```
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(Cost)  
##  
## 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
```

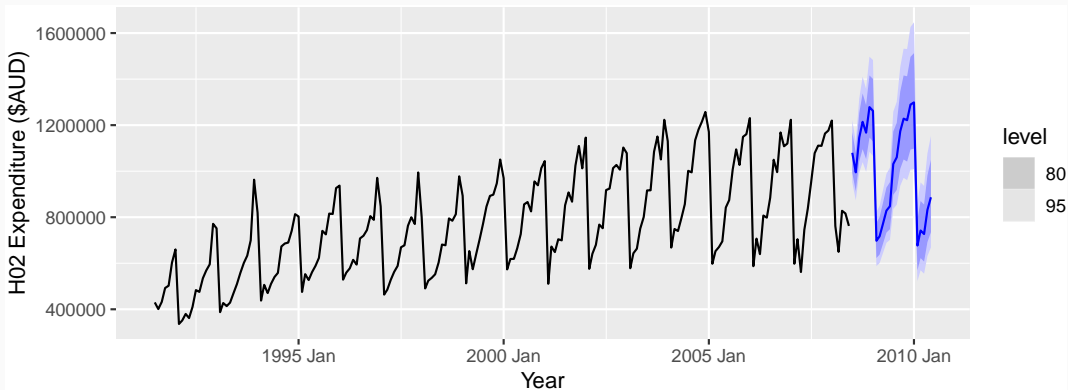
# Corticosteroid drug sales

```
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(Cost)  
##  
## Coefficients:  
##          ar1      ar2      ar3      ar4      ma1      sar1      sar2      sma1      sma2  
##      -0.0426  0.210  0.202  -0.227  -0.742  0.621  -0.383  -1.202  0.496  
## s.e.   0.2167  0.181  0.114   0.081   0.207  0.242   0.118   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 ARIMA models
- 2 Lab Session 16
- 3 Seasonal ARIMA models
- 4 Lab Session 17
- 5 Forecast ensembles

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 ARIMA models
- 2 Lab Session 16
- 3 Seasonal ARIMA models
- 4 Lab Session 17
- 5 Forecast ensembles



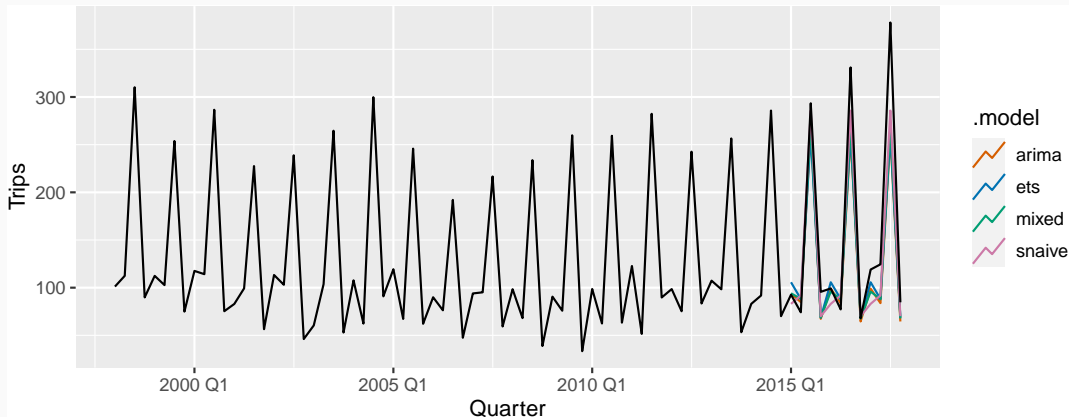
# Forecast ensembles

```
train <- tourism %>%  
  filter(year(Quarter) <= 2014)  
fit <- train %>%  
  model(  
    ets = ETS(Trips),  
    arima = ARIMA(Trips),  
    snaive = SNAIVE(Trips)  
  ) %>%  
  mutate(mixed = (ets + arima + snaive) / 3)
```

- Ensemble forecast `mixed` is a simple average of the three fitted models.
- `forecast()` will produce distributional forecasts taking into account the correlations between the forecast errors of the component models.

# Forecast ensembles

```
fc <- fit %>% forecast(h = "3 years")  
fc %>% filter(Region == "Snowy Mountains", Purpose == "Holiday") %>%  
  autoplot(tourism, level = NULL)
```



# Forecast ensembles

```
accuracy(fc, tourism) %>%  
  group_by(.model) %>%  
  summarise(  
    RMSE = mean(RMSE),  
    MAE = mean(MAE),  
    MASE = mean(MASE)  
  ) %>%  
  arrange(RMSE)
```

```
## # A tibble: 4 x 4  
##   .model  RMSE    MAE  MASE  
##   <chr>  <dbl> <dbl> <dbl>  
## 1 mixed   19.8  16.0  0.997  
## 2 ets     20.2  16.4  1.00  
## 3 snaive  21.5  17.3  1.17  
## 4 arima   21.9  17.8  1.06
```

# Forecast ensembles

**Can we do better than equal weights?**

# Forecast ensembles

## Can we do better than equal weights?

- Hard to find weights that improve forecast accuracy.
- Known as the “forecast combination puzzle”.
- Solution: FFORMA

# Forecast ensembles

## Can we do better than equal weights?

- Hard to find weights that improve forecast accuracy.
- Known as the “forecast combination puzzle”.
- Solution: FFORMA

## FFORMA (Feature-based FOREcast Model Averaging)

- Vector of time series features used to predict best weights.
- A modification of xgboost is used.
- Method came 2nd in the 2018 M4 international forecasting competition.
- Main author: Pablo Montero-Manso (Monash U)
- Not (yet) available for fable.