MONASH University

MONASH BUSINESS SCHOOL

ETC3550/ETC5550
Applied forecasting

Ch9. ARIMA models

OTexts.org/fpp3/

Outline

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.

Outline

Stationarity

Definition

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

Stationarity

Definition

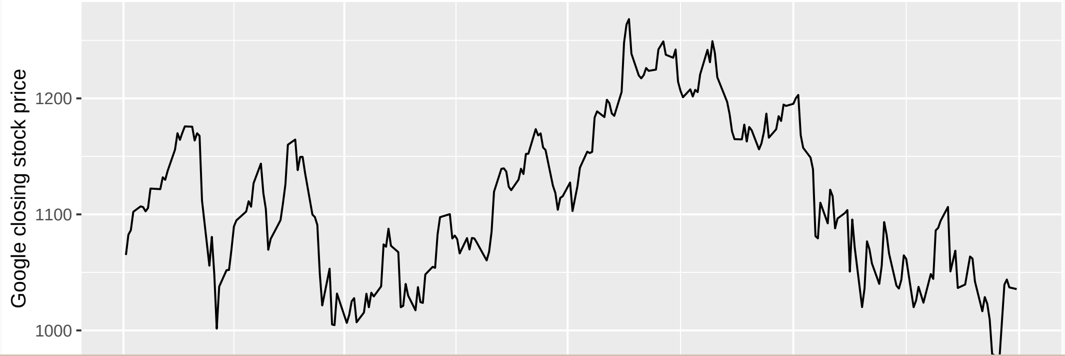
If $\{y_t\}$ is a stationary time series, then for all s, the distribution of $(y_t, \dots, y_{t+s})$ does not depend on t.

A **stationary series** is:

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

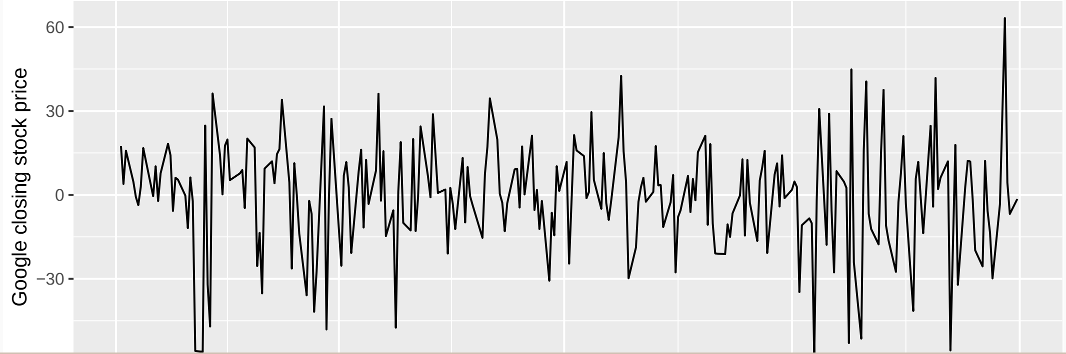
Stationary?

```
gafa_stock %>%
  filter(Symbol == "GOOG", year(Date) == 2018) %>%
  autoplot(Close) +
  labs(y = "Google closing stock price", x = "Day")
```



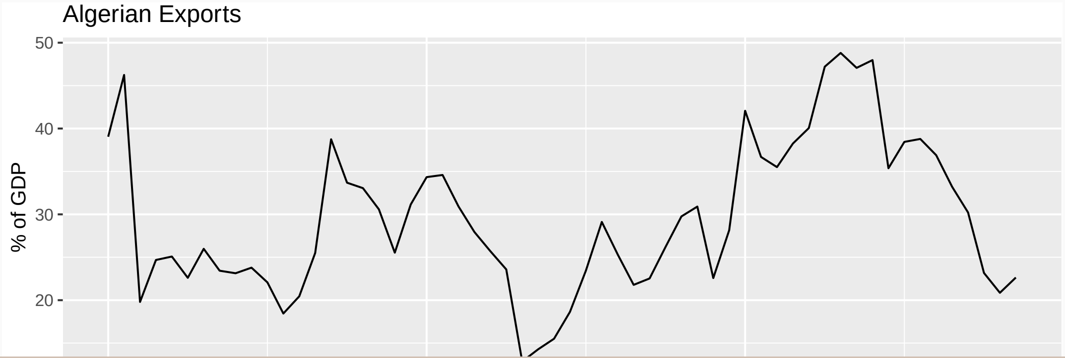
Stationary?

```
gafa_stock %>%
  filter(Symbol == "GOOG", year(Date) == 2018) %>%
  autoplot(difference(Close)) +
  labs(y = "Google closing stock price", x = "Day")
```



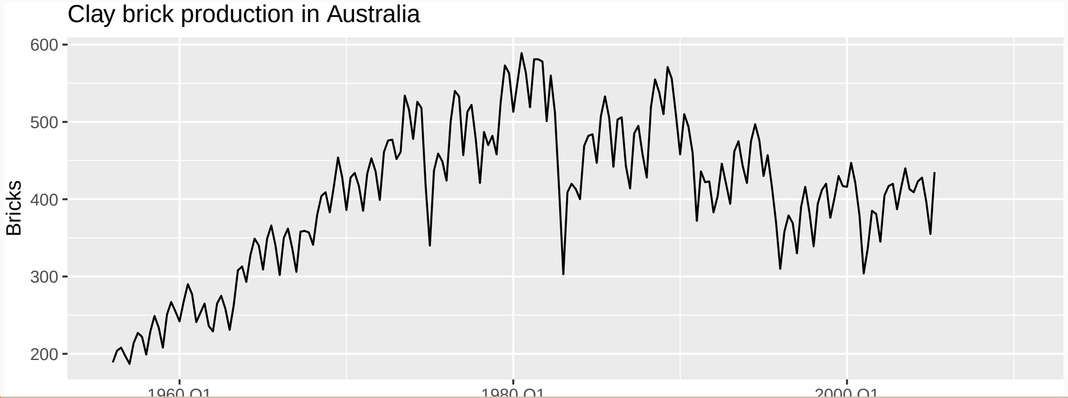
Stationary?

```
global_economy %>%
  filter(Country == "Algeria") %>%
  autoplot(Exports) +
  labs(y = "% of GDP", title = "Algerian Exports")
```



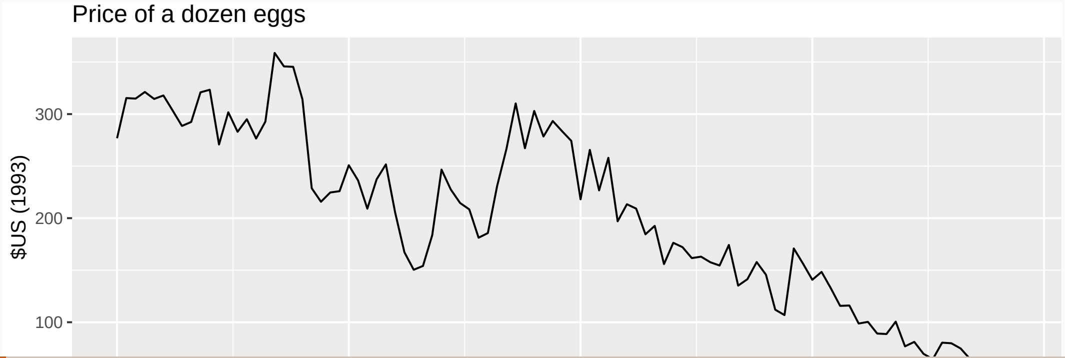
Stationary?

```
aus_production %>%
  autoplot(Bricks) +
  labs(title = "Clay brick production in Australia")
```



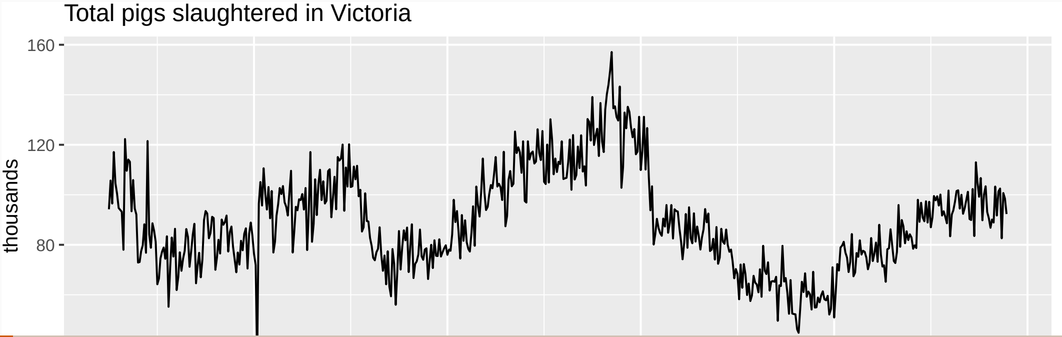
Stationary?

```
prices %>%
  filter(year >= 1900) %>%
  autoplot(eggs) +
  labs(y="$US (1993)", title="Price of a dozen eggs")
```



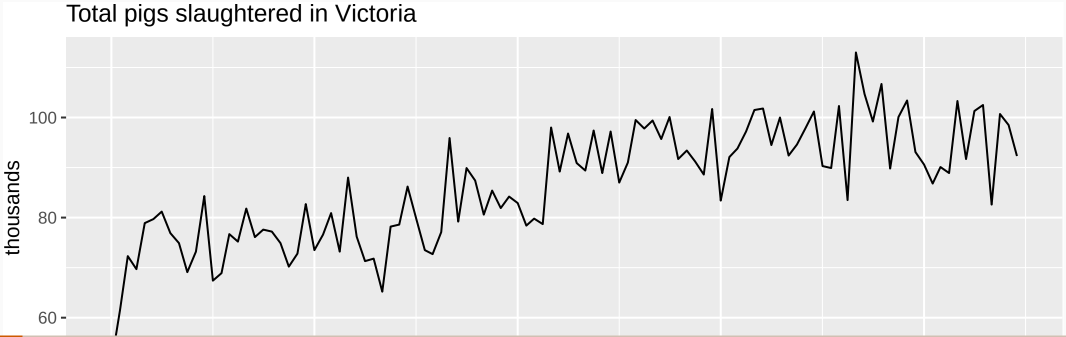
Stationary?

```
aus_livestock %>%
  filter(
    Animal == "Pigs", State == "Victoria",
  ) %>%
  autoplot(Count/1e3) +
  labs(y = "thousands", title = "Total pigs slaughtered in Victoria")
```



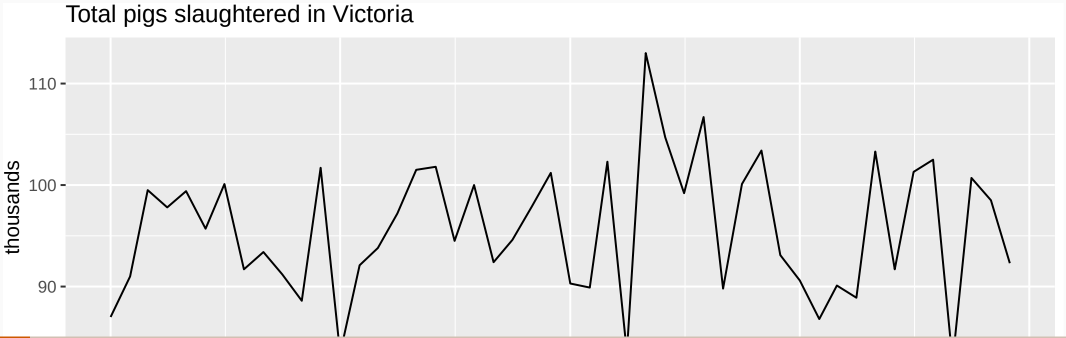
Stationary?

```
aus_livestock %>%
  filter(
    Animal == "Pigs", State == "Victoria", year(Month) >= 2010
  ) %>%
  autoplot(Count/1e3) +
  labs(y = "thousands", title = "Total pigs slaughtered in Victoria")
```



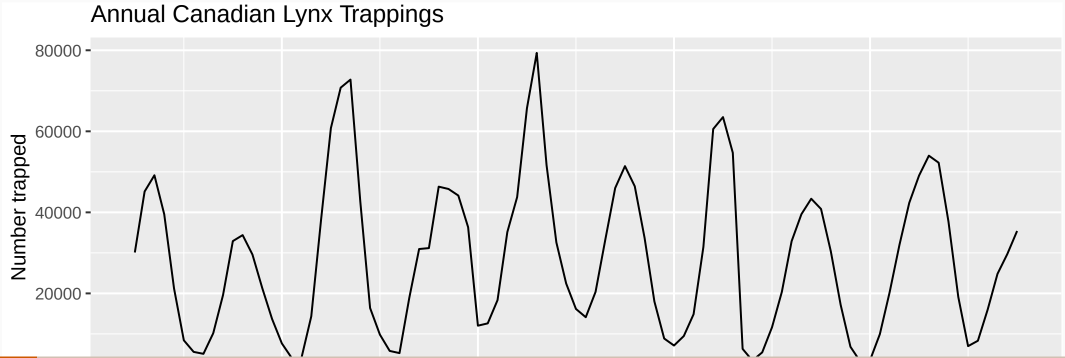
Stationary?

```
aus_livestock %>%
  filter(
    Animal == "Pigs", State == "Victoria", year(Month) >= 2015
  ) %>%
  autoplot(Count/1e3) +
  labs(y = "thousands", title = "Total pigs slaughtered in Victoria")
```



Stationary?

```r
pelt %>%
  autoplot(Lynx) +
  labs(y = "Number trapped",
       title = "Annual Canadian Lynx Trappings")
```



Stationarity

Definition

If $\{y_t\}$ is a stationary time series, then for all s, the distribution of $(y_t, \ldots, y_{t+s})$ 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.

Transformations help to **stabilize the variance**.

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

Non-stationarity in the mean

Identifying non-stationary series

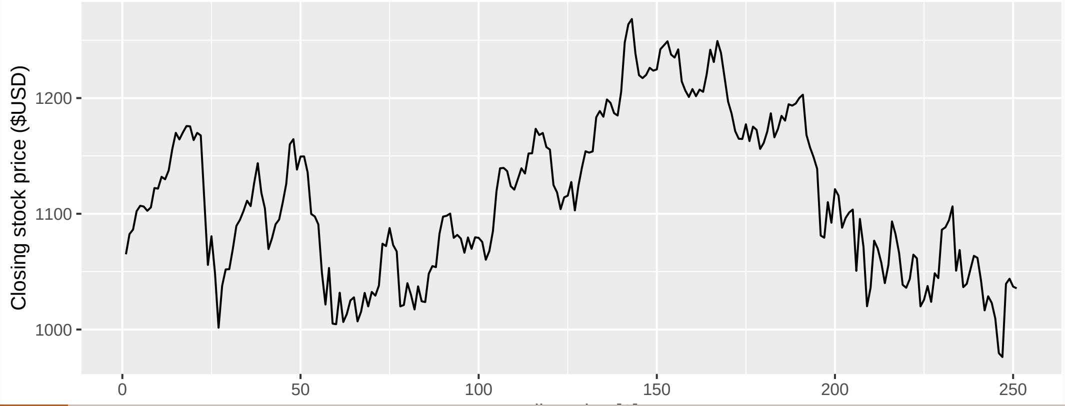
- time plot.
- The ACF of stationary data drops to zero relatively quickly
- The ACF of non-stationary data decreases slowly.
- For non-stationary data, the value of r_1 is often large and positive.

Example: Google stock price

```
google_2018 <- gafa_stock %>%
  filter(Symbol == "GOOG", year(Date) == 2018) %>%
  mutate(trading_day = row_number()) %>%
  update_tsibble(index = trading_day, regular = TRUE)
```

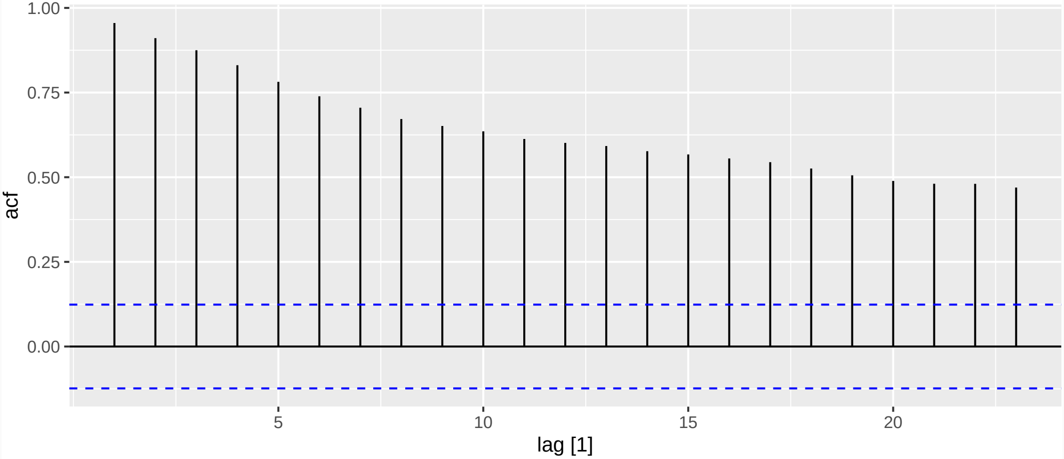
Example: Google stock price

```
google_2018 %>%
  autoplot(Close) +
  labs(y = "Closing stock price ($USD)")
```



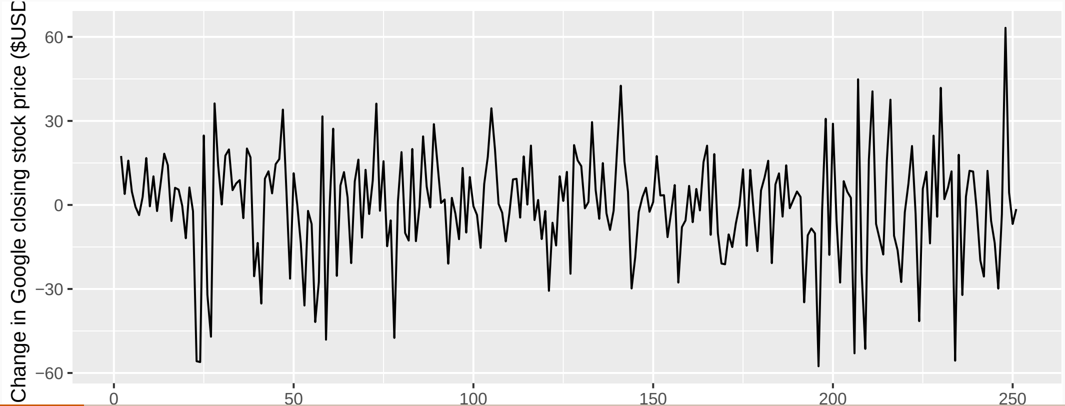
Example: Google stock price

```
google_2018 %>% ACF(Close) %>% autoplot()
```



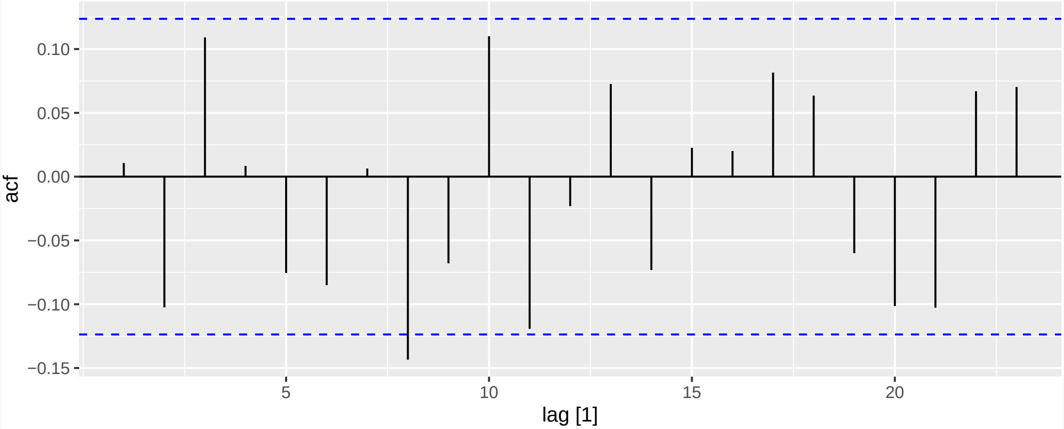
Example: Google stock price

```
google_2018 %>%
  autoplot(difference(Close)) +
  labs(y = "Change in Google closing stock price ($USD)")
```



Example: Google stock price

```
google_2018 %>% ACF(difference(Close)) %>% autoplot()
```



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:

Second-order differencing

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

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

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.

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.

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.

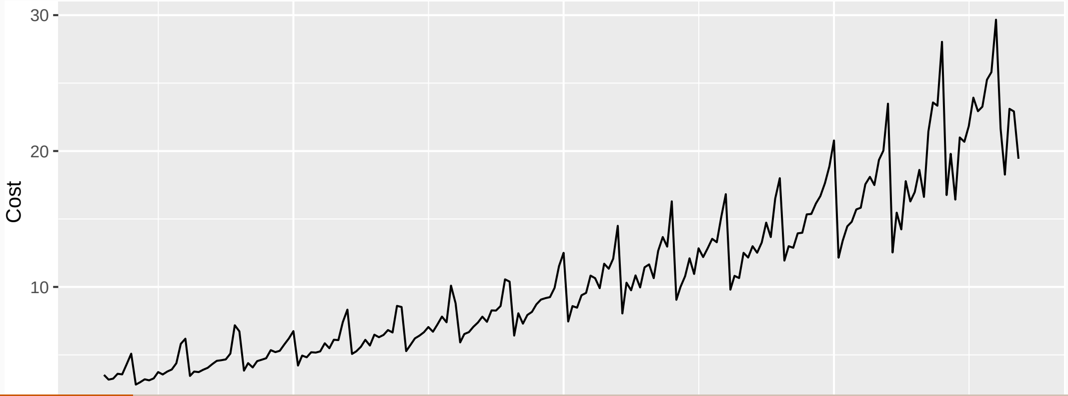
- For monthly data $m = 12$.
- For quarterly data $m = 4$.

Antidiabetic drug sales

```
a10 <- PBS %>%
  filter(ATC2 == "A10") %>%
  summarise(Cost = sum(Cost)/1e6)
```

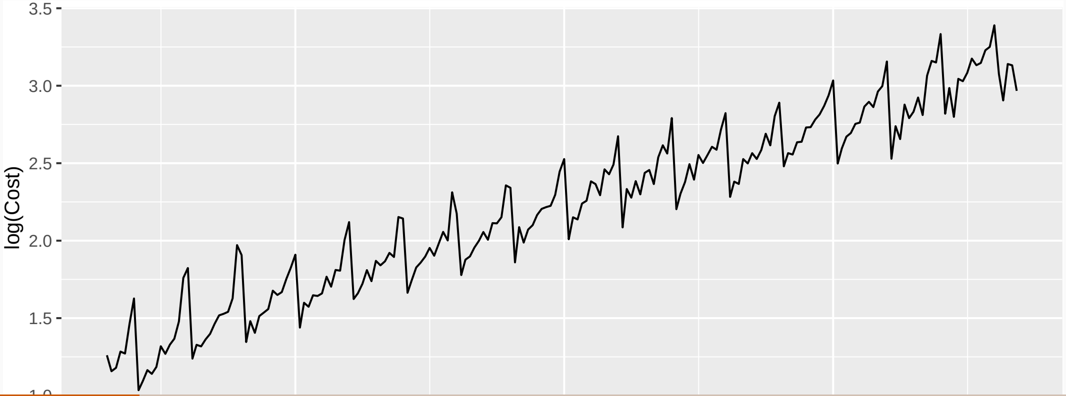
Antidiabetic drug sales

```
a10 %>% autoplot(
  Cost
)
```



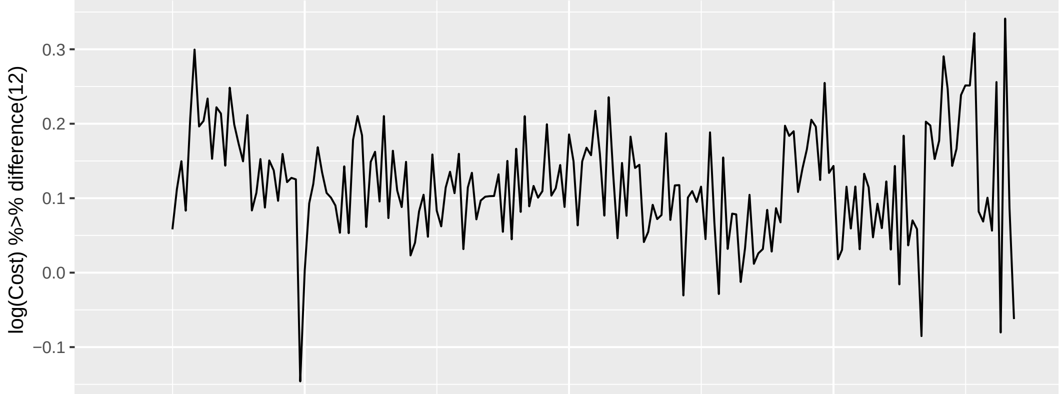
Antidiabetic drug sales

```
a10 %>% autoplot(
  log(Cost)
)
```



Antidiabetic drug sales

```r
a10 %>% autoplot(
  log(Cost) %>% difference(12)
)
```

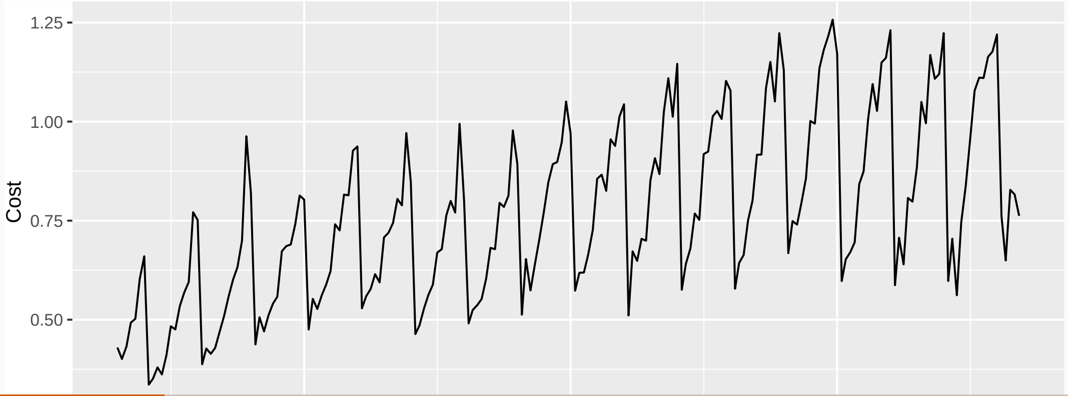


Cortecosteroid drug sales

```
h02 <- PBS %>%
  filter(ATC2 == "H02") %>%
  summarise(Cost = sum(Cost)/1e6)
```

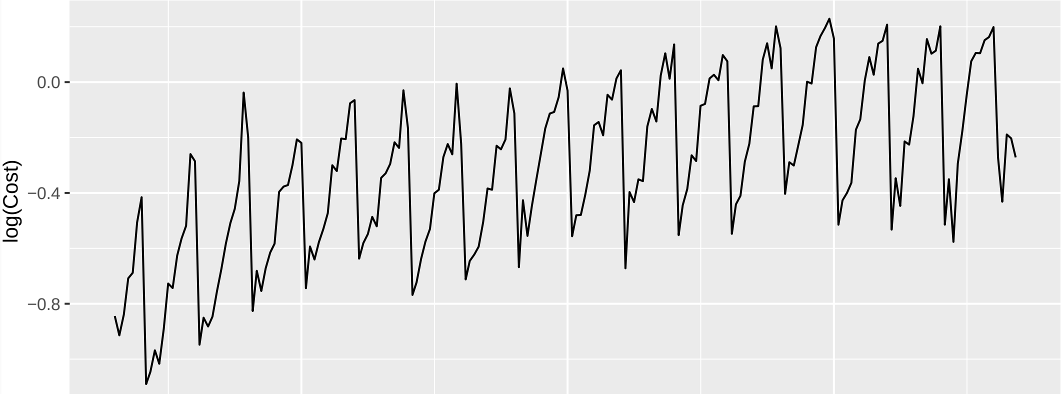
Cortecosteroid drug sales

```
h02 %>% autoplot(
  Cost
)
```



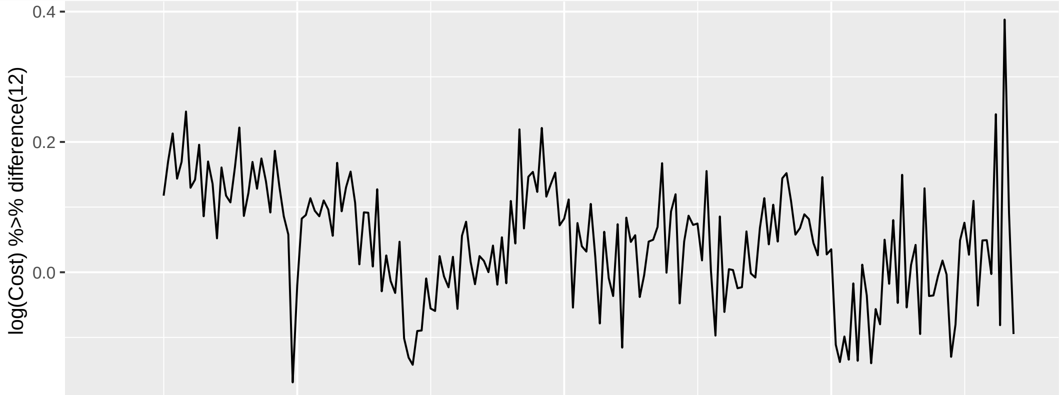
Cortecosteroid drug sales

```
h02 %>% autoplot(
  log(Cost)
)
```



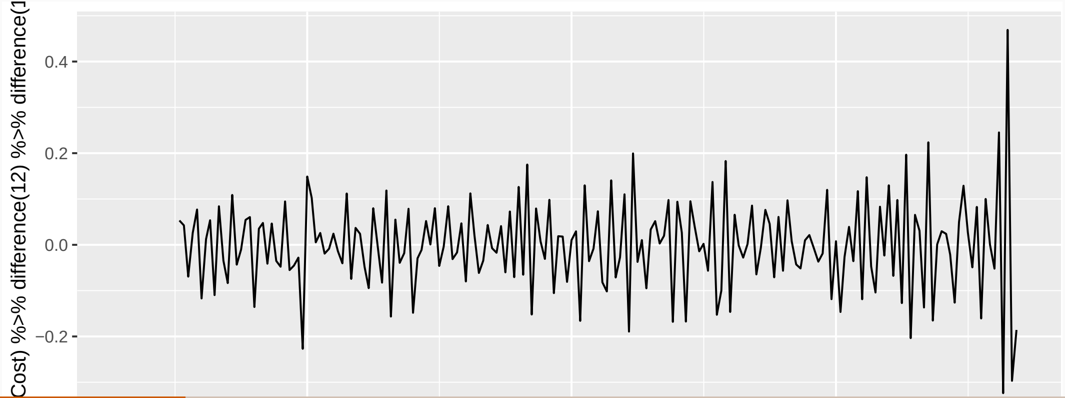
Cortecosteroid drug sales

```r
h02 %>% autoplot(
  log(Cost) %>% difference(12)
)
```



Cortecosteroid drug sales

```
h02 %>% autoplot(
  log(Cost) %>% difference(12) %>% difference(1)
)
```



Cortecosteroid drug sales

- Seasonally differenced series is closer to being stationary.
- Remaining non-stationarity can be removed with further first difference.

If $y'_t = y_t - y_{t-12}$ denotes seasonally differenced series, then twice-differenced series is

$$\begin{aligned} y^*_t &= y'_t - y'_{t-1} \\ &= (y_t - y_{t-12}) - (y_{t-1} - y_{t-13}) \\ &= y_t - y_{t-1} - y_{t-12} + y_{t-13}\,. \end{aligned}$$

Seasonal differencing

When both seasonal and first differences are applied…

Seasonal differencing

When both seasonal and first differences are applied…

- it makes no difference which is done first—the result will be the same.
- If seasonality is strong, we recommend that seasonal differencing be done first because sometimes the resulting series will be stationary and there will be no need for further first difference.

Seasonal differencing

When both seasonal and first differences are applied...

- it makes no difference which is done first—the result will be the same.
- If seasonality is strong, we recommend that seasonal differencing be done first because sometimes the resulting series will be stationary and there will be no need for further first difference.

It is important that if differencing is used, the differences are interpretable.

Interpretation of differencing

- first differences are the change between **one observation and the next**;
- seasonal differences are the change between **one year to the next**.

Interpretation of differencing

- first differences are the change between **one observation and the next**;
- seasonal differences are the change between **one year to the next**.

But taking lag 3 differences for yearly data, for example, results in a model which cannot be sensibly interpreted.

Unit root tests

Statistical tests to determine the required order of differencing.

1. Augmented Dickey Fuller test: null hypothesis is that the data are non-stationary and non-seasonal.
2. Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test: null hypothesis is that the data are stationary and non-seasonal.
3. Other tests available for seasonal data.

KPSS test

```
google_2018 %>%
  features(Close, unitroot_kpss)
```

```
## # A tibble: 1 x 3
##   Symbol kpss_stat kpss_pvalue
##   <chr>      <dbl>       <dbl>
## 1 GOOG       0.573      0.0252
```

KPSS test

```r
google_2018 %>%
  features(Close, unitroot_kpss)
```

```
## # A tibble: 1 x 3
##   Symbol kpss_stat kpss_pvalue
##   <chr>      <dbl>       <dbl>
## 1 GOOG       0.573      0.0252
```

```r
google_2018 %>%
  features(Close, unitroot_ndiffs)
```

```
## # A tibble: 1 x 2
##   Symbol ndiffs
##   <chr>   <int>
## 1 GOOG        1
```

Automatically selecting differences

STL decomposition: $y_t = T_t + S_t + R_t$

Seasonal strength $F_s = \max\left(0, 1 - \frac{\text{Var}(R_t)}{\text{Var}(S_t+R_t)}\right)$

If $F_s > 0.64$, do one seasonal difference.

```
h02 %>% mutate(log_sales = log(Cost)) %>%
  features(log_sales, list(unitroot_nsdiffs, feat_stl))
```

```
## # A tibble: 1 x 10
##   nsdiffs trend_strength seasonal_strength_ye~ seasonal_peak_y~ seasonal_trough~
##     <int>          <dbl>                 <dbl>            <dbl>            <dbl>
## 1       1          0.957                 0.955                6                8
## # ... with 5 more variables: spikiness <dbl>, linearity <dbl>, curvature <dbl>,
## #   stl_e_acf1 <dbl>, stl_e_acf10 <dbl>
```

Automatically selecting differences

```r
h02 %>% mutate(log_sales = log(Cost)) %>%
  features(log_sales, unitroot_nsdiffs)
```

```
## # A tibble: 1 x 1
##   nsdiffs
##     <int>
## 1       1
```

```r
h02 %>% mutate(d_log_sales = difference(log(Cost), 12)) %>%
  features(d_log_sales, unitroot_ndiffs)
```

```
## # A tibble: 1 x 1
##   ndiffs
##    <int>
## 1      1
```

Backshift notation

A very useful notational device is the backward shift operator, B, which is used as follows:

$$By_t = y_{t-1}$$

Backshift notation

A very useful notational device is the backward shift operator, B, which is used as follows:

$$By_t = y_{t-1}$$

In other words, B, operating on y_t, has the effect of **shifting the data back one period**.

Backshift notation

A very useful notational device is the backward shift operator, B, which is used as follows:

$$By_t = y_{t-1}$$

In other words, B, operating on y_t, has the effect of **shifting the data back one period**. Two applications of B to y_t **shifts the data back two periods**:

$$B(By_t) = B^2y_t = y_{t-2}$$

Backshift notation

A very useful notational device is the backward shift operator, B, which is used as follows:

$$By_t = y_{t-1}$$

In other words, B, operating on y_t, has the effect of **shifting the data back one period**. Two applications of B to y_t **shifts the data back two periods**:

$$B(By_t) = B^2y_t = y_{t-2}$$

For monthly data, if we wish to shift attention to "the same month last year", then B^{12} is used, and the notation is $B^{12}y_t = y_{t-12}$.

Backshift notation

The backward shift operator is convenient for describing the process of *differencing*.

Backshift notation

The backward shift operator is convenient for describing the process of *differencing*. A first difference can be written as

$$y'_t = y_t - y_{t-1} = y_t - By_t = (1 - B)y_t$$

Backshift notation

The backward shift operator is convenient for describing the process of *differencing*. A first difference can be written as

$$y'_t = y_t - y_{t-1} = y_t - By_t = (1 - B)y_t$$

Note that a first difference is represented by $(1 - B)$.

Backshift notation

The backward shift operator is convenient for describing the process of *differencing*. A first difference can be written as

$$y'_t = y_t - y_{t-1} = y_t - By_t = (1 - B)y_t$$

Note that a first difference is represented by $(1 - B)$.

Similarly, if second-order differences (i.e., first differences of first differences) have to be computed, then:

$$y''_t = y_t - 2y_{t-1} + y_{t-2} = (1 - B)^2 y_t$$

Backshift notation

- Second-order difference is denoted $(1 - B)^2$.
- *Second-order difference* is not the same as a *second difference*, which would be denoted $1 - B^2$;
- In general, a dth-order difference can be written as
$$(1 - B)^d y_t$$
- A seasonal difference followed by a first difference can be written as
$$(1 - B)(1 - B^m) y_t$$

Backshift notation

The "backshift" notation is convenient because the terms can be multiplied together to see the combined effect.

$$\begin{aligned}(1 - B)(1 - B^m)y_t &= (1 - B - B^m + B^{m+1})y_t \\ &= y_t - y_{t-1} - y_{t-m} + y_{t-m-1}.\end{aligned}$$

Backshift notation

The "backshift" notation is convenient because the terms can be multiplied together to see the combined effect.

$$\begin{aligned}(1-B)(1-B^m)y_t &= (1-B-B^m+B^{m+1})y_t \\ &= y_t - y_{t-1} - y_{t-m} + y_{t-m-1}.\end{aligned}$$

For monthly data, $m = 12$ and we obtain the same result as earlier.

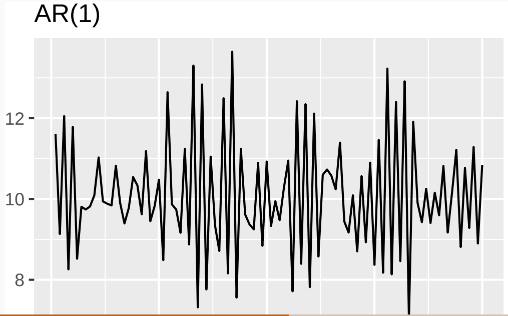
Outline

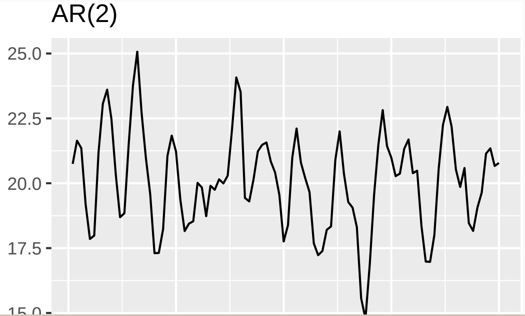
Autoregressive models

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.

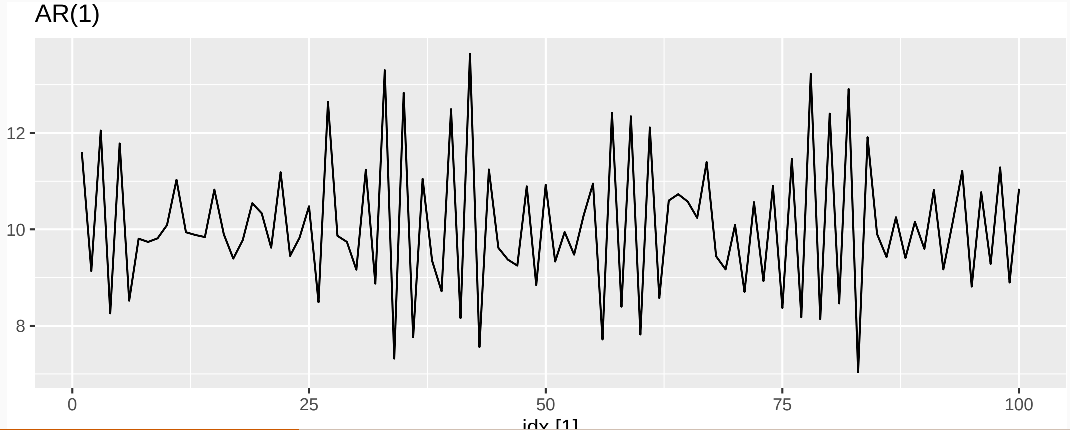




AR(1) model

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

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



AR(1) model

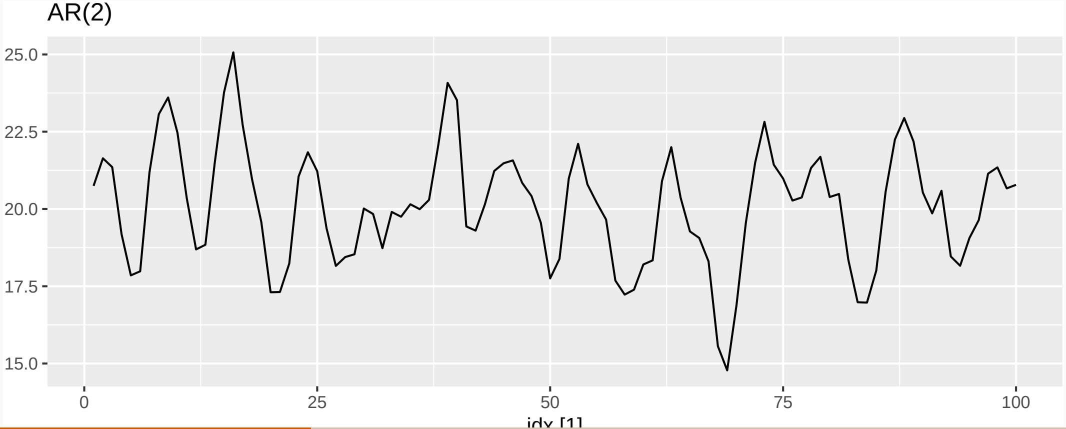
$$y_t = c + \phi_1 y_{t-1} + \varepsilon_t$$

- When $\phi_1 = 0$, y_t is **equivalent to WN**
- When $\phi_1 = 1$ and $c = 0$, y_t is **equivalent to a RW**
- When $\phi_1 = 1$ and $c \neq 0$, y_t is **equivalent to a RW with drift**
- When $\phi_1 < 0$, y_t tends to **oscillate between positive and negative values**.

AR(2) model

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

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



Stationarity conditions

We normally restrict autoregressive models to stationary data, and then some constraints on the values of the parameters are required.

General condition for stationarity

Complex roots of $1 - \phi_1 z - \phi_2 z^2 - \cdots - \phi_p z^p$ lie outside the unit circle on the complex plane.

Stationarity conditions

We normally restrict autoregressive models to stationary data, and then some constraints on the values of the parameters are required.

General condition for stationarity

Complex roots of $1 - \phi_1 z - \phi_2 z^2 - \dots - \phi_p z^p$ lie outside the unit circle on the complex plane.

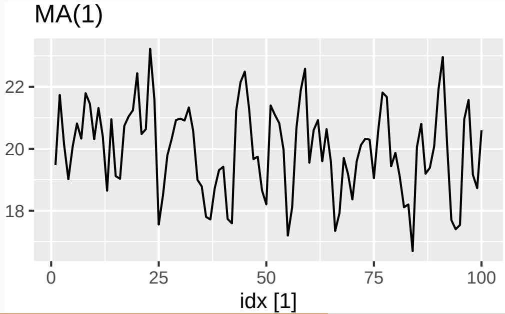
- For $p = 1$: $-1 < \phi_1 < 1$.
- For $p = 2$:
  $-1 < \phi_2 < 1 \qquad \phi_2 + \phi_1 < 1 \qquad \phi_2 - \phi_1 < 1.$
- More complicated conditions hold for $p \geq 3$.
- Estimation software takes care of this.

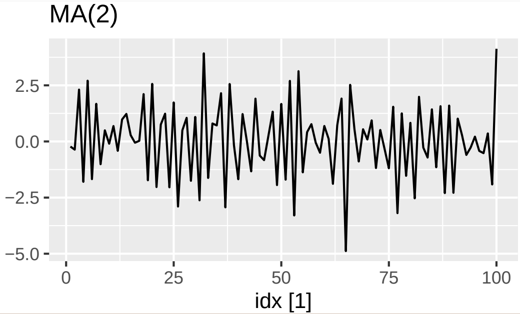
Moving Average (MA) models

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!*

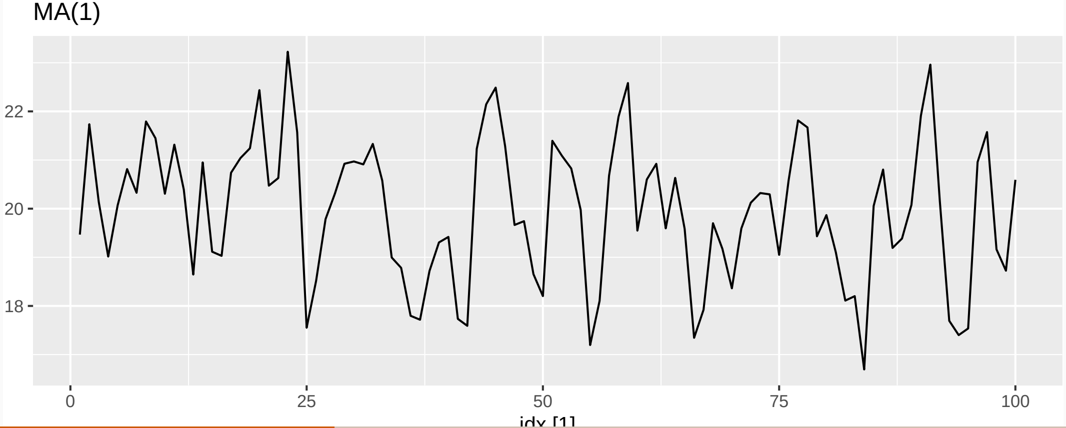




MA(1) model

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

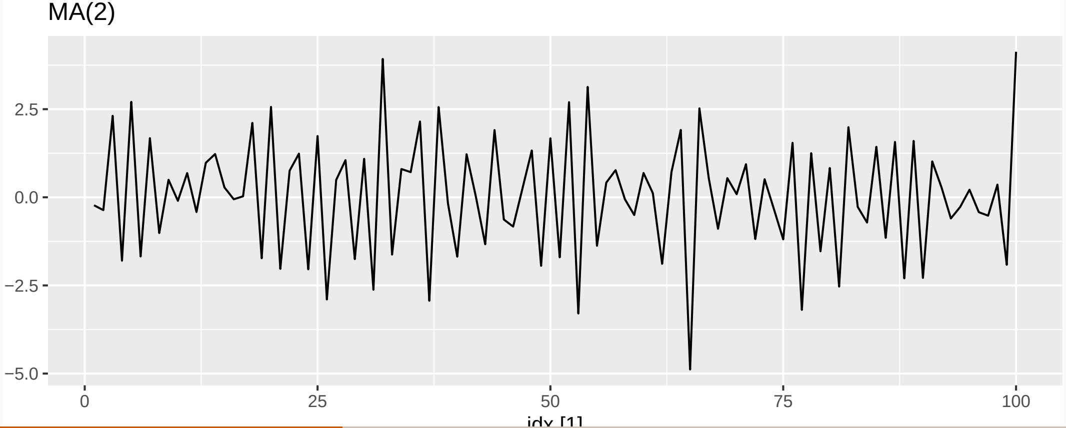
$$\varepsilon_t \sim N(0,1), \quad T = 100.$$



MA(2) model

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

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



MA(∞) models

It is possible to write any stationary AR(p) process as an MA(∞) process.

Example: AR(1)

$$\begin{aligned} y_t &= \phi_1 y_{t-1} + \varepsilon_t \\ &= \phi_1(\phi_1 y_{t-2} + \varepsilon_{t-1}) + \varepsilon_t \\ &= \phi_1^2 y_{t-2} + \phi_1 \varepsilon_{t-1} + \varepsilon_t \\ &= \phi_1^3 y_{t-3} + \phi_1^2 \varepsilon_{t-2} + \phi_1 \varepsilon_{t-1} + \varepsilon_t \\ &\dots \end{aligned}$$

MA(∞) models

It is possible to write any stationary AR(p) process as an MA(∞) process.

Example: AR(1)

$$
\begin{aligned}
y_t &= \phi_1 y_{t-1} + \varepsilon_t \\
&= \phi_1(\phi_1 y_{t-2} + \varepsilon_{t-1}) + \varepsilon_t \\
&= \phi_1^2 y_{t-2} + \phi_1 \varepsilon_{t-1} + \varepsilon_t \\
&= \phi_1^3 y_{t-3} + \phi_1^2 \varepsilon_{t-2} + \phi_1 \varepsilon_{t-1} + \varepsilon_t \\
&\dots
\end{aligned}
$$

Provided $-1 < \phi_1 < 1$:

Invertibility

- Any MA(q) process can be written as an AR(∞) process if we impose some constraints on the MA parameters.
- Then the MA model is called "invertible".
- Invertible models have some mathematical properties that make them easier to use in practice.
- Invertibility of an ARIMA model is equivalent to forecastability of an ETS model.

Invertibility

General condition for invertibility

Complex roots of $1 + \theta_1 z + \theta_2 z^2 + \cdots + \theta_q z^q$ lie outside the unit circle on the complex plane.

Invertibility

General condition for invertibility

Complex roots of $1 + \theta_1 z + \theta_2 z^2 + \dots + \theta_q z^q$ lie outside the unit circle on the complex plane.

- For $q = 1$: $-1 < \theta_1 < 1$.
- For $q = 2$:

  $$-1 < \theta_2 < 1 \qquad \theta_2 + \theta_1 > -1 \qquad \theta_1 - \theta_2 < 1.$$

- More complicated conditions hold for $q \geq 3$.
- Estimation software takes care of this.

ARIMA models

Autoregressive Moving Average models:

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

ARIMA models

Autoregressive Moving Average models:

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

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

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.**
- Conditions on coefficients ensure stationarity.
- Conditions on coefficients ensure invertibility.

Autoregressive Integrated Moving Average models

- Combine ARMA model with **differencing**.
- $(1 - B)^d y_t$ follows an ARMA model.

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)

Backshift notation for ARIMA

- ARMA model:

$$
\begin{aligned}
& y_t = c + \phi_1 B y_t + \cdots + \phi_p B^p y_t + \varepsilon_t + \theta_1 B \varepsilon_t + \cdots + \theta_q B^q \varepsilon_t \\
\text{or} \quad & (1 - \phi_1 B - \cdots - \phi_p B^p) y_t = c + (1 + \theta_1 B + \cdots + \theta_q B^q)\varepsilon_t
\end{aligned}
$$

- ARIMA(1,1,1) model:

$$
\underbrace{(1 - \phi_1 B)}_{\uparrow \atop \text{AR(1)}} \; \underbrace{(1 - B)}_{\uparrow \atop \text{First difference}} y_t \;=\; c + \underbrace{(1 + \theta_1 B)}_{\uparrow \atop \text{MA(1)}} \varepsilon_t
$$

Backshift notation for ARIMA

- ARMA model:

$$y_t = c + \phi_1 B y_t + \cdots + \phi_p B^p y_t + \varepsilon_t + \theta_1 B \varepsilon_t + \cdots + \theta_q B^q \varepsilon_t$$

$$\text{or} \quad (1 - \phi_1 B - \cdots - \phi_p B^p) y_t = c + (1 + \theta_1 B + \cdots + \theta_q B^q)\varepsilon_t$$

- ARIMA(1,1,1) model:

$$\underset{\substack{\uparrow \\ \text{AR(1)}}}{(1 - \phi_1 B)} \quad \underset{\substack{\uparrow \\ \text{First} \\ \text{difference}}}{(1 - B) y_t} \quad = \quad c + \underset{\substack{\uparrow \\ \text{MA(1)}}}{(1 + \theta_1 B)\varepsilon_t}$$

Written out:

$$y_t = c + y_{t-1} + \phi_1 y_{t-1} - \phi_1 y_{t-2} + \theta_1 \varepsilon_{t-1} + \varepsilon_t$$

R model

Intercept form

$$(1 - \phi_1 B - \cdots - \phi_p B^p) y'_t = c + (1 + \theta_1 B + \cdots + \theta_q B^q)\varepsilon_t$$

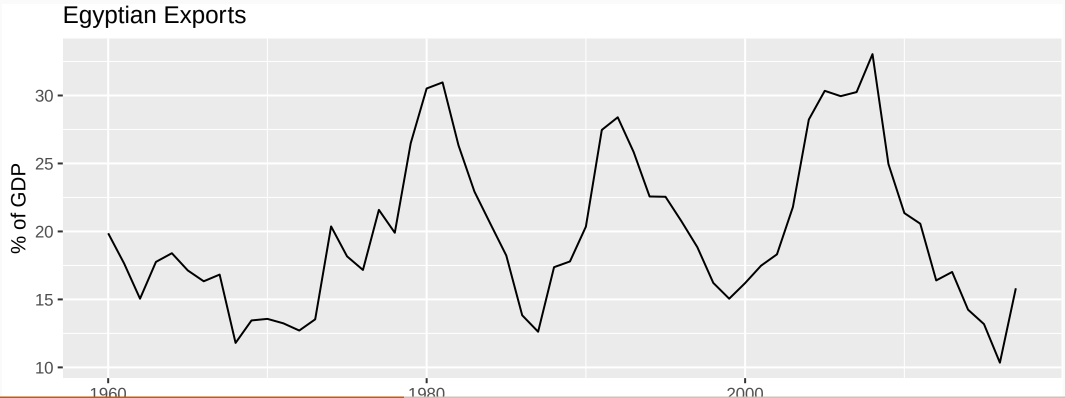
Mean form

$$(1 - \phi_1 B - \cdots - \phi_p B^p)(y'_t - \mu) = (1 + \theta_1 B + \cdots + \theta_q B^q)\varepsilon_t$$

- $y'_t = (1 - B)^d y_t$
- μ is the mean of y'_t.
- $c = \mu(1 - \phi_1 - \cdots - \phi_p)$.
- fable uses intercept form

Egyptian exports

```
global_economy %>%
  filter(Code == "EGY") %>%
  autoplot(Exports) +
  labs(y = "% of GDP", title = "Egyptian Exports")
```



Egyptian exports

```
fit <- global_economy %>% filter(Code == "EGY") %>%
  model(ARIMA(Exports))
report(fit)
```

```
## Series: Exports
## Model: ARIMA(2,0,1) w/ mean
##
## Coefficients:
##          ar1      ar2     ma1  constant
##        1.676  -0.8034  -0.690     2.562
## s.e.   0.111   0.0928   0.149     0.116
##
## sigma^2 estimated as 8.046:  log likelihood=-142
## AIC=293   AICc=294   BIC=303
```

Egyptian exports

```
fit <- global_economy %>% filter(Code == "EGY") %>%
  model(ARIMA(Exports))
report(fit)
```

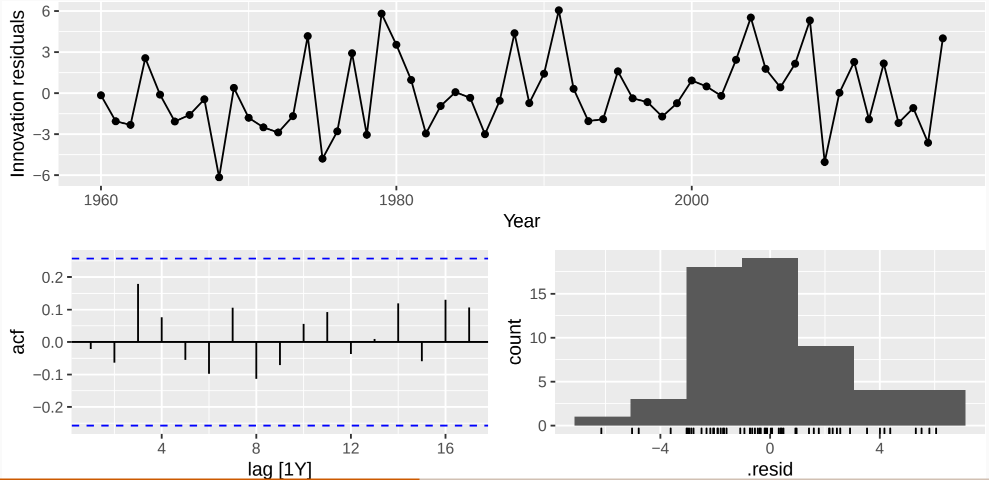
```
## Series: Exports
## Model: ARIMA(2,0,1) w/ mean
##
## Coefficients:
##          ar1      ar2     ma1  constant
##        1.676  -0.8034  -0.690     2.562
## s.e.   0.111   0.0928   0.149     0.116
##
## sigma^2 estimated as 8.046:  log likelihood=-142
## AIC=293   AICc=294   BIC=303
```

ARIMA(2,0,1) model:

$$y_t = 2.56 + 1.68 y_{t-1} - 0.80 y_{t-2} - 0.69 \varepsilon_{t-1} + \varepsilon_t,$$

Egyptian exports

```
gg_tsresiduals(fit)
```



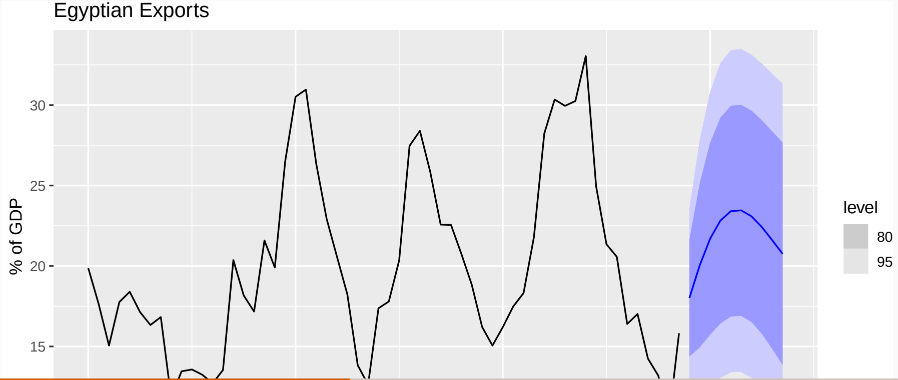
Egyptian exports

```r
augment(fit) %>%
  features(.innov, ljung_box, lag = 10, dof = 4)
```

```
## # A tibble: 1 x 4
##   Country          .model          lb_stat lb_pvalue
##   <fct>            <chr>             <dbl>     <dbl>
## 1 Egypt, Arab Rep. ARIMA(Exports)     5.78     0.448
```

Egyptian exports

```
fit %>% forecast(h=10) %>%
  autoplot(global_economy) +
  labs(y = "% of GDP", title = "Egyptian Exports")
```



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.

Cyclic behaviour

- For cyclic forecasts, $p \geq 2$ and some restrictions on coefficients are required.
- If $p = 2$, we need $\phi_1^2 + 4\phi_2 < 0$. Then average cycle of length

$$(2\pi)/\left[\text{arc cos}(-\phi_1(1 - \phi_2)/(4\phi_2))\right].$$

Outline

Maximum likelihood estimation

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

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 `ARIMA()` function allows CLS or MLE estimation.
- Non-linear optimization must be used in either case.
- Different software will give different estimates.

Partial autocorrelations

Partial autocorrelations measure relationship between y_t and y_{t-k}, when the effects of other time lags — $1, 2, 3, \dots, k-1$ — are removed.

Partial autocorrelations

Partial autocorrelations measure relationship between y_t and y_{t-k}, when the effects of other time lags — $1, 2, 3, \dots, k-1$ — are removed.

$$\begin{aligned}\alpha_k &= k\text{th partial autocorrelation coefficient}\\ &= \text{equal to the estimate of } \phi_k \text{ in regression:}\\ &\quad y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \dots + \phi_k y_{t-k} + \varepsilon_t.\end{aligned}$$

Partial autocorrelations

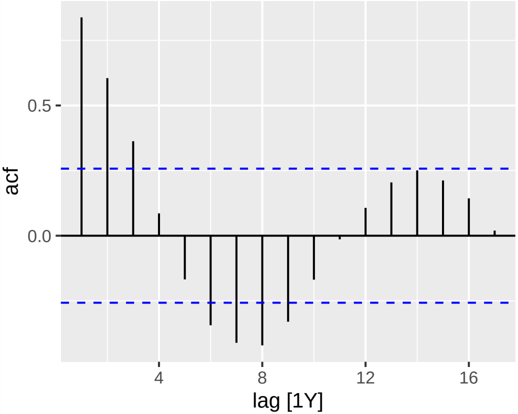
Partial autocorrelations measure relationship between y_t and y_{t-k}, when the effects of other time lags — $1, 2, 3, \dots, k-1$ — are removed.

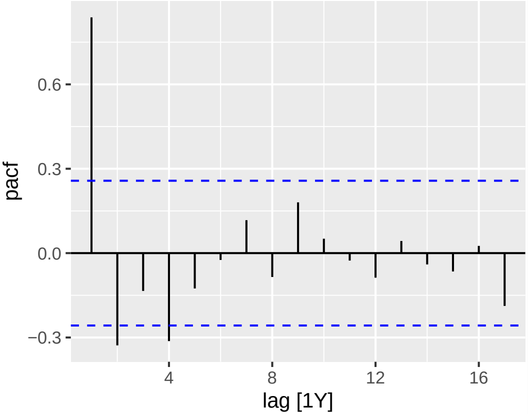
$$
\begin{aligned}
\alpha_k &= k\text{th partial autocorrelation coefficient} \\
&= \text{equal to the estimate of } \phi_k \text{ in regression:} \\
&\quad y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \dots + \phi_k y_{t-k} + \varepsilon_t.
\end{aligned}
$$

- Varying number of terms on RHS gives α_k for different values of k.
- $\alpha_1 = \rho_1$
- same critical values of $\pm 1.96/\sqrt{T}$ as for ACF.
- Last significant α_k indicates the order of an AR model.

Egyptian exports

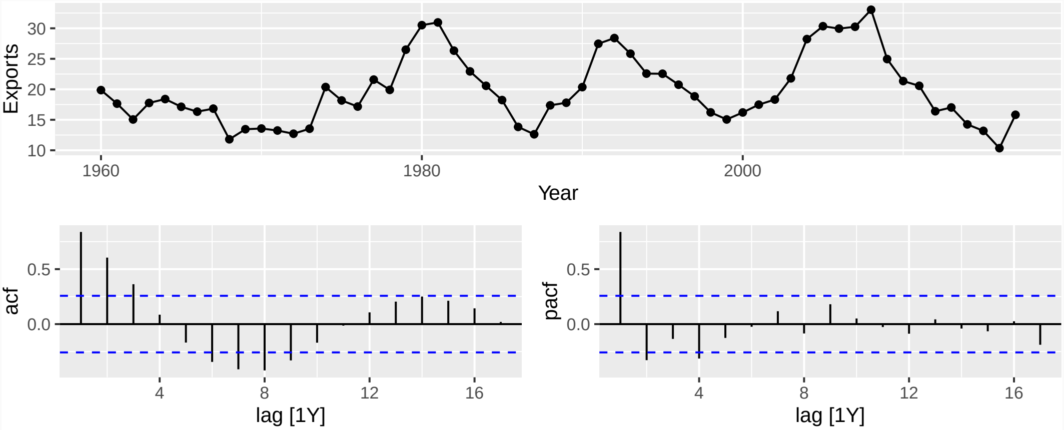
```
egypt <- global_economy %>% filter(Code == "EGY")
egypt %>% ACF(Exports) %>% autoplot()
egypt %>% PACF(Exports) %>% autoplot()
```





Egyptian exports

```
global_economy %>% filter(Code == "EGY") %>%
  gg_tsdisplay(Exports, plot_type='partial')
```



ACF and PACF interpretation

AR(1)

$$\rho_k = \phi_1^k \qquad \text{for } k = 1, 2, \dots;$$
$$\alpha_1 = \phi_1 \qquad \alpha_k = 0 \qquad \text{for } k = 2, 3, \dots.$$

So we have an AR(1) model when

- autocorrelations exponentially decay
- there is a single significant partial autocorrelation.

ACF and PACF interpretation

AR(p)

- ACF dies out in an exponential or damped sine-wave manner
- PACF has all zero spikes beyond the pth spike

So we have an AR(p) model when

- the ACF is exponentially decaying or sinusoidal
- there is a significant spike at lag p in PACF, but none beyond p

ACF and PACF interpretation

MA(1)

$$\rho_1 = \theta_1/(1+\theta_1^2) \qquad \rho_k = 0 \qquad \text{for } k = 2, 3, \dots;$$
$$\alpha_k = -(-\theta_1)^k/(1+\theta_1^2+\cdots+\theta_1^{2k})$$

So we have an MA(1) model when

- the PACF is exponentially decaying and
- there is a single significant spike in ACF

ACF and PACF interpretation

MA(q)

- PACF dies out in an exponential or damped sine-wave manner
- ACF has all zero spikes beyond the qth spike

So we have an MA(q) model when

- the PACF is exponentially decaying or sinusoidal
- there is a significant spike at lag q in ACF, but none beyond q

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 $c = 0$.

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:

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

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:

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

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.

Outline

How does ARIMA() work?

A non-seasonal ARIMA process

$$\phi(B)(1 - B)^d y_t = c + \theta(B)\varepsilon_t$$

Need to select appropriate orders: p, q, d

Hyndman and Khandakar (JSS, 2008) algorithm:

- Select no. differences d and D via KPSS test and seasonal strength measure.
- Select p, q by minimising AICc.
- Use stepwise search to traverse model space.

How does ARIMA() work?

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

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?

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

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

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?

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

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

Step1: Select current model (with smallest AICc) from:
ARIMA$(2, d, 2)$
ARIMA$(0, d, 0)$
ARIMA$(1, d, 0)$
ARIMA$(0, d, 1)$

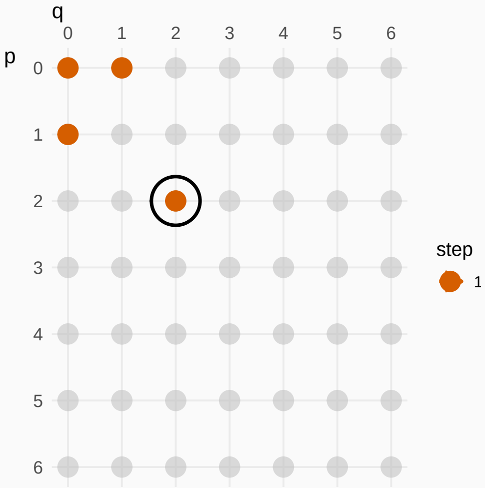
Step 2: Consider variations of current model:

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

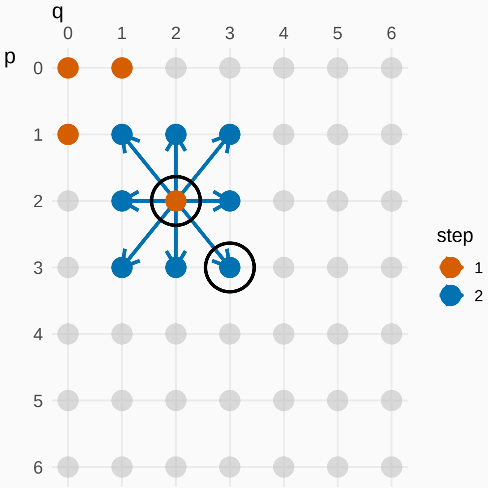
Model with lowest AICc becomes current model.

Repeat Step 2 until no lower AICc can be found.

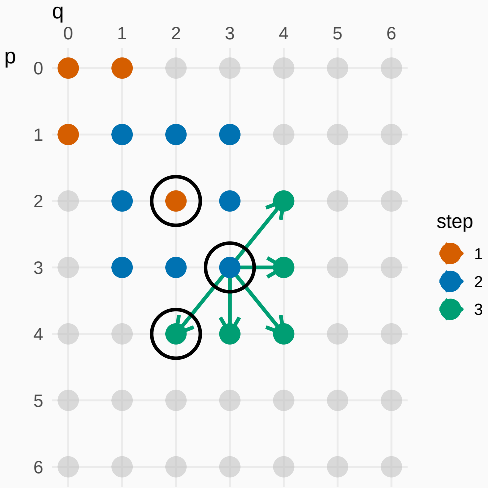
How does ARIMA() work?



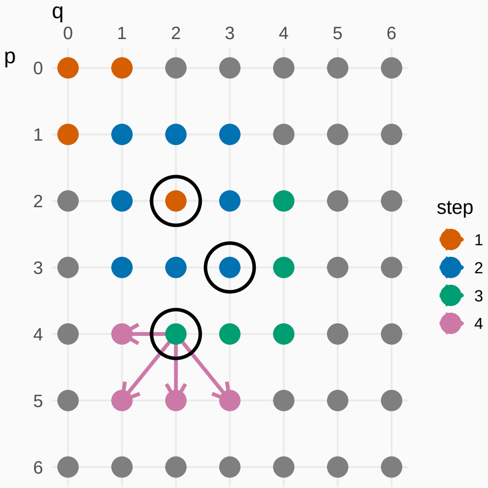
How does ARIMA() work?



How does ARIMA() work?

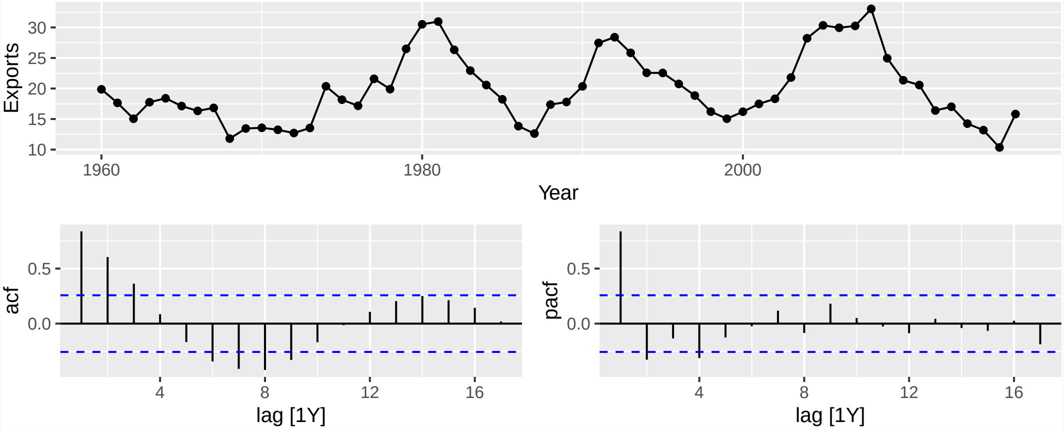


How does ARIMA() work?



Egyptian exports

```
global_economy %>% filter(Code == "EGY") %>%
  gg_tsdisplay(Exports, plot_type='partial')
```



Egyptian exports

```r
fit1 <- global_economy %>%
  filter(Code == "EGY") %>%
  model(ARIMA(Exports ~ pdq(4,0,0)))
report(fit1)
```

```
## Series: Exports
## Model: ARIMA(4,0,0) w/ mean
##
## Coefficients:
##          ar1      ar2    ar3      ar4  constant
##        0.986  -0.172  0.181  -0.328     6.692
## s.e.   0.125   0.186  0.186   0.127     0.356
##
## sigma^2 estimated as 7.885:  log likelihood=-141
## AIC=293   AICc=295   BIC=305
```

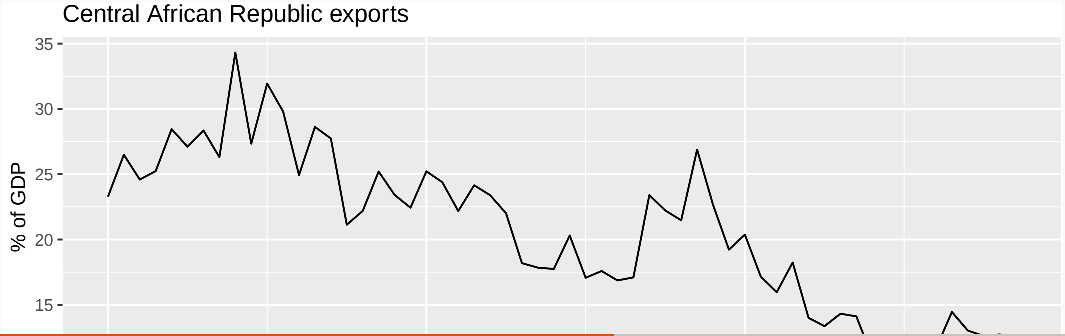
Egyptian exports

```
fit2 <- global_economy %>%
  filter(Code == "EGY") %>%
  model(ARIMA(Exports))
report(fit2)
```

```
## Series: Exports
## Model: ARIMA(2,0,1) w/ mean
##
## Coefficients:
##          ar1      ar2     ma1  constant
##        1.676  -0.8034  -0.690     2.562
## s.e.   0.111   0.0928   0.149     0.116
##
## sigma^2 estimated as 8.046:  log likelihood=-142
## AIC=293   AICc=294   BIC=303
```

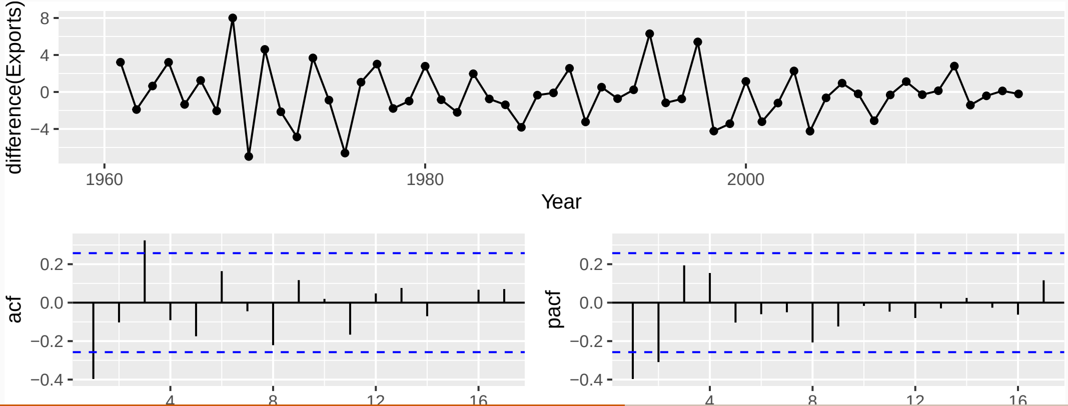
Central African Republic exports

```
global_economy %>%
  filter(Code == "CAF") %>%
  autoplot(Exports) +
  labs(title="Central African Republic exports",
       y="% of GDP")
```



Central African Republic exports

```
global_economy %>%
  filter(Code == "CAF") %>%
  gg_tsdisplay(difference(Exports), plot_type='partial')
```



Central African Republic exports

```r
caf_fit <- global_economy %>%
  filter(Code == "CAF") %>%
  model(arima210 = ARIMA(Exports ~ pdq(2,1,0)),
        arima013 = ARIMA(Exports ~ pdq(0,1,3)),
        stepwise = ARIMA(Exports),
        search = ARIMA(Exports, stepwise=FALSE))
```

Central African Republic exports

```
caf_fit %>% pivot_longer(!Country, names_to = "Model name",
                         values_to = "Orders")
```

```
## # A mable: 4 x 3
## # Key:     Country, Model name [4]
##   Country                  `Model name`        Orders
##   <fct>                    <chr>              <model>
## 1 Central African Republic arima210     <ARIMA(2,1,0)>
## 2 Central African Republic arima013     <ARIMA(0,1,3)>
## 3 Central African Republic stepwise     <ARIMA(2,1,2)>
## 4 Central African Republic search       <ARIMA(3,1,0)>
```

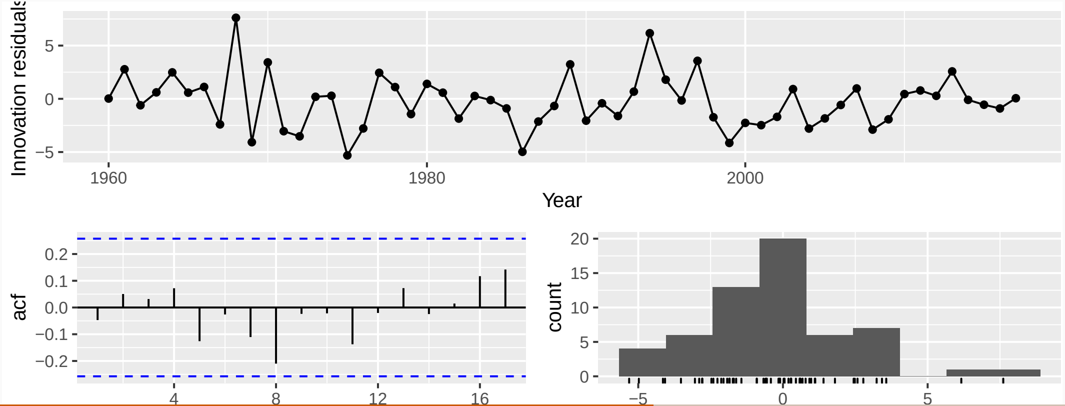
Central African Republic exports

```
glance(caf_fit) %>% arrange(AICc) %>% select(.model:BIC)
```

```
## # A tibble: 4 x 6
##   .model   sigma2 log_lik   AIC  AICc   BIC
##   <chr>     <dbl>   <dbl> <dbl> <dbl> <dbl>
## 1 search     6.52   -133.  274.  275.  282.
## 2 arima210   6.71   -134.  275.  275.  281.
## 3 arima013   6.54   -133.  274.  275.  282.
## 4 stepwise   6.42   -132.  274.  275.  284.
```

Central African Republic exports

```
caf_fit %>%
  select(search) %>%
  gg_tsresiduals()
```



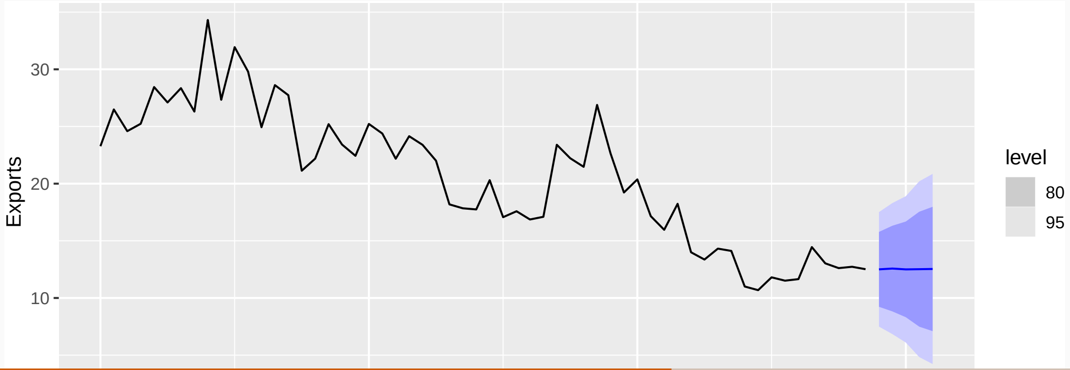
Central African Republic exports

```
augment(caf_fit) %>%
  filter(.model=='search') %>%
  features(.innov, ljung_box, lag = 10, dof = 3)
```

```
## # A tibble: 1 x 4
##   Country                  .model lb_stat lb_pvalue
##   <fct>                    <chr>    <dbl>     <dbl>
## 1 Central African Republic search    5.75     0.569
```

Central African Republic exports

```
caf_fit %>%
  forecast(h=5) %>%
  filter(.model=='search') %>%
  autoplot(global_economy)
```



Modelling procedure with `ARIMA()`

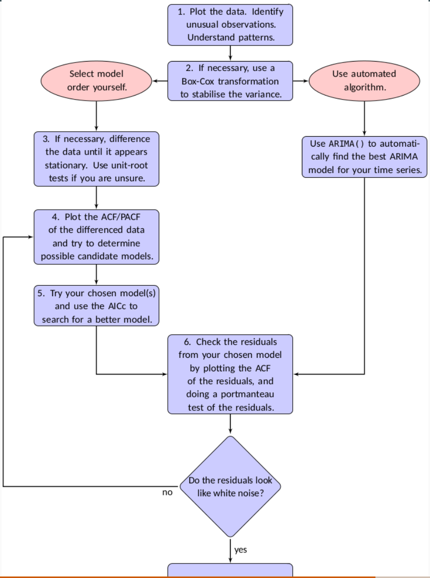
1. Plot the data. Identify any unusual observations.
2. If necessary, transform the data (using a Box-Cox transformation) to stabilize the variance.
3. If the data are non-stationary: take first differences of the data until the data are stationary.
4. Examine the ACF/PACF: Is an AR(p) or MA(q) model appropriate?
5. Try your chosen model(s), and use the AICc to search for a better model.
6. Check the residuals from your chosen model by plotting the ACF of the residuals, and doing a portmanteau test of the residuals. If they do not look like white noise, try a modified model.
7. Once the residuals look like white noise, calculate forecasts.

Automatic modelling procedure with `ARIMA()`

1. Plot the data. Identify any unusual observations.
2. If necessary, transform the data (using a Box-Cox transformation) to stabilize the variance.
3. Use `ARIMA` to automatically select a model.

6. Check the residuals from your chosen model by plotting the ACF of the residuals, and doing a portmanteau test of the residuals. If they do not look like white noise, try a modified model.
7. Once the residuals look like white noise, calculate forecasts.

Modelling procedure



Outline

1 Stationarity and differencing

2 Non-seasonal ARIMA models

3 Estimation and order selection

4 ARIMA modelling in R

5 Forecasting

6 Seasonal ARIMA models

7 ARIMA vs ETS

Point forecasts

1. Rearrange ARIMA equation so y_t is on LHS.
2. Rewrite equation by replacing t by $T + h$.
3. On RHS, replace future observations by their forecasts, future errors by zero, and past errors by corresponding residuals.

Start with $h = 1$. Repeat for $h = 2, 3, \ldots$.

Point forecasts

ARIMA(3,1,1) forecasts: Step 1

$$(1 - \phi_1 B - \phi_2 B^2 - \phi_3 B^3)(1 - B)y_t = (1 + \theta_1 B)\varepsilon_t,$$

Point forecasts

ARIMA(3,1,1) forecasts: Step 1

$$(1-\phi_1B-\phi_2B^2-\phi_3B^3)(1-B)y_t = (1+\theta_1B)\varepsilon_t,$$

$$\begin{aligned}\left[1-(1+\phi_1)B+(\phi_1-\phi_2)B^2+(\phi_2-\phi_3)B^3+\phi_3B^4\right]y_t \\ = (1+\theta_1B)\varepsilon_t,\end{aligned}$$

Point forecasts

ARIMA(3,1,1) forecasts: Step 1

$$(1 - \phi_1 B - \phi_2 B^2 - \phi_3 B^3)(1 - B)y_t = (1 + \theta_1 B)\varepsilon_t,$$

$$\begin{aligned}\left[1 - (1 + \phi_1)B + (\phi_1 - \phi_2)B^2 + (\phi_2 - \phi_3)B^3 + \phi_3 B^4\right] y_t \\ = (1 + \theta_1 B)\varepsilon_t,\end{aligned}$$

$$\begin{aligned}y_t - (1 + \phi_1)y_{t-1} + (\phi_1 - \phi_2)y_{t-2} + (\phi_2 - \phi_3)y_{t-3} \\ + \phi_3 y_{t-4} = \varepsilon_t + \theta_1 \varepsilon_{t-1}.\end{aligned}$$

Point forecasts

ARIMA(3,1,1) forecasts: Step 1

$$(1 - \phi_1 B - \phi_2 B^2 - \phi_3 B^3)(1 - B) y_t = (1 + \theta_1 B)\varepsilon_t,$$

$$\begin{aligned}\left[1 - (1+\phi_1)B + (\phi_1 - \phi_2)B^2 + (\phi_2 - \phi_3)B^3 + \phi_3 B^4\right] y_t \\ = (1 + \theta_1 B)\varepsilon_t,\end{aligned}$$

$$\begin{aligned}y_t - (1+\phi_1)y_{t-1} + (\phi_1 - \phi_2)y_{t-2} + (\phi_2 - \phi_3)y_{t-3} \\ + \phi_3 y_{t-4} = \varepsilon_t + \theta_1\varepsilon_{t-1}.\end{aligned}$$

$$\begin{aligned}y_t = (1+\phi_1)y_{t-1} - (\phi_1 - \phi_2)y_{t-2} - (\phi_2 - \phi_3)y_{t-3} \\ - \phi_3 y_{t-4} + \varepsilon_t + \theta_1\varepsilon_{t-1}.\end{aligned}$$

Point forecasts (h=1)

$$
\begin{aligned}
y_t = (1+\phi_1)y_{t-1} - (\phi_1 - \phi_2)y_{t-2} - (\phi_2 - \phi_3)y_{t-3} \\
- \phi_3 y_{t-4} + \varepsilon_t + \theta_1\varepsilon_{t-1}.
\end{aligned}
$$

Point forecasts (h=1)

$$
\begin{aligned}
y_t = (1+\phi_1)y_{t-1} - (\phi_1 - \phi_2)y_{t-2} - (\phi_2 - \phi_3)y_{t-3} \\
- \phi_3 y_{t-4} + \varepsilon_t + \theta_1\varepsilon_{t-1}.
\end{aligned}
$$

ARIMA(3,1,1) forecasts: Step 2

$$
\begin{aligned}
y_{T+1} = (1+\phi_1)y_T - (\phi_1 - \phi_2)y_{T-1} - (\phi_2 - \phi_3)y_{T-2} \\
- \phi_3 y_{T-3} + \varepsilon_{T+1} + \theta_1\varepsilon_T.
\end{aligned}
$$

Point forecasts (h=1)

$$y_t = (1+\phi_1)y_{t-1} - (\phi_1 - \phi_2)y_{t-2} - (\phi_2 - \phi_3)y_{t-3} - \phi_3 y_{t-4} + \varepsilon_t + \theta_1\varepsilon_{t-1}.$$

ARIMA(3,1,1) forecasts: Step 2

$$y_{T+1} = (1+\phi_1)y_T - (\phi_1 - \phi_2)y_{T-1} - (\phi_2 - \phi_3)y_{T-2} - \phi_3 y_{T-3} + \varepsilon_{T+1} + \theta_1\varepsilon_T.$$

ARIMA(3,1,1) forecasts: Step 3

$$\hat{y}_{T+1|T} = (1+\phi_1)y_T - (\phi_1 - \phi_2)y_{T-1} - (\phi_2 - \phi_3)y_{T-2} - \phi_3 y_{T-3} + \theta_1 e_T.$$

Point forecasts (h=2)

$$y_t = (1+\phi_1)y_{t-1} - (\phi_1 - \phi_2)y_{t-2} - (\phi_2 - \phi_3)y_{t-3} - \phi_3 y_{t-4} + \varepsilon_t + \theta_1\varepsilon_{t-1}.$$

Point forecasts (h=2)

$$y_t = (1 + \phi_1)y_{t-1} - (\phi_1 - \phi_2)y_{t-2} - (\phi_2 - \phi_3)y_{t-3} - \phi_3 y_{t-4} + \varepsilon_t + \theta_1\varepsilon_{t-1}.$$

ARIMA(3,1,1) forecasts: Step 2

$$y_{T+2} = (1 + \phi_1)y_{T+1} - (\phi_1 - \phi_2)y_T - (\phi_2 - \phi_3)y_{T-1} - \phi_3 y_{T-2} + \varepsilon_{T+2} + \theta_1\varepsilon_{T+1}.$$

Point forecasts (h=2)

$$y_t = (1 + \phi_1)y_{t-1} - (\phi_1 - \phi_2)y_{t-2} - (\phi_2 - \phi_3)y_{t-3} - \phi_3 y_{t-4} + \varepsilon_t + \theta_1\varepsilon_{t-1}.$$

ARIMA(3,1,1) forecasts: Step 2

$$y_{T+2} = (1 + \phi_1)y_{T+1} - (\phi_1 - \phi_2)y_T - (\phi_2 - \phi_3)y_{T-1} - \phi_3 y_{T-2} + \varepsilon_{T+2} + \theta_1\varepsilon_{T+1}.$$

ARIMA(3,1,1) forecasts: Step 3

$$\hat{y}_{T+2|T} = (1 + \phi_1)\hat{y}_{T+1|T} - (\phi_1 - \phi_2)y_T - (\phi_2 - \phi_3)y_{T-1} - \phi_3 y_{T-2}.$$

Prediction intervals

95% prediction interval

$$\hat{y}_{T+h|T} \pm 1.96\sqrt{v_{T+h|T}}$$

where $v_{T+h|T}$ is estimated forecast variance.

Prediction intervals

95% prediction interval

$$\hat{y}_{T+h|T} \pm 1.96\sqrt{v_{T+h|T}}$$

where $v_{T+h|T}$ is estimated forecast variance.

- $v_{T+1|T} = \hat{\sigma}^2$ for all ARIMA models regardless of parameters and orders.
- Multi-step prediction intervals for ARIMA(0,0,q):

$$y_t = \varepsilon_t + \sum_{i=1}^{q} \theta_i \varepsilon_{t-i}.$$

$$v_{T|T+h} = \hat{\sigma}^2 \left[1 + \sum_{i=1}^{h-1} \theta_i^2\right], \qquad \text{for } h = 2, 3, \dots.$$

Prediction intervals

95% prediction interval

$$\hat{y}_{T+h|T} \pm 1.96\sqrt{v_{T+h|T}}$$

where $v_{T+h|T}$ is estimated forecast variance.

- Multi-step prediction intervals for ARIMA(0,0,q):

$$y_t = \varepsilon_t + \sum_{i=1}^{q} \theta_i \varepsilon_{t-i}.$$

$$v_{T|T+h} = \hat{\sigma}^2 \left[1 + \sum_{i=1}^{h-1} \theta_i^2\right], \qquad \text{for } h = 2, 3, \dots.$$

Prediction intervals

95% prediction interval

$$\hat{y}_{T+h|T} \pm 1.96\sqrt{v_{T+h|T}}$$

where $v_{T+h|T}$ is estimated forecast variance.

- Multi-step prediction intervals for ARIMA(0,0,q):

$$y_t = \varepsilon_t + \sum_{i=1}^{q} \theta_i \varepsilon_{t-i}.$$

$$v_{T|T+h} = \hat{\sigma}^2 \left[1 + \sum_{i=1}^{h-1} \theta_i^2\right], \qquad \text{for } h = 2, 3, \dots.$$

- AR(1): Rewrite as MA(∞) and use above result.
- Other models beyond scope of this subject.

Prediction intervals

- Prediction intervals **increase in size with forecast horizon**.
- Prediction intervals can be difficult to calculate by hand
- Calculations assume residuals are **uncorrelated** and **normally distributed**.
- Prediction intervals tend to be too narrow.
  - the uncertainty in the parameter estimates has not been accounted for.
  - the ARIMA model assumes historical patterns will not change during the forecast period.
  - the ARIMA model assumes uncorrelated future errors

Outline

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.

Seasonal ARIMA models

E.g., ARIMA$(1, 1, 1)(1, 1, 1)_4$ model (without constant)

Seasonal ARIMA models

E.g., ARIMA$(1,1,1)(1,1,1)_4$ model (without constant)

$$(1-\phi_1 B)(1-\Phi_1 B^4)(1-B)(1-B^4)y_t = (1+\theta_1 B)(1+\Theta_1 B^4)\varepsilon_t.$$

Seasonal ARIMA models

E.g., ARIMA$(1, 1, 1)(1, 1, 1)_4$ model (without constant)

$$(1 - \phi_1 B)(1 - \Phi_1 B^4)(1 - B)(1 - B^4)y_t \;=\; (1 + \theta_1 B)(1 + \Theta_1 B^4)\varepsilon_t.$$

(Non-seasonal AR(1))

(Seasonal AR(1))

(Non-seasonal difference)

(Seasonal difference)

(Non-seasonal MA(1))

(Seasonal MA(1))

Seasonal ARIMA models

E.g., ARIMA$(1, 1, 1)(1, 1, 1)_4$ model (without constant)

$$(1 - \phi_1 B)(1 - \Phi_1 B^4)(1 - B)(1 - B^4)y_t = (1 + \theta_1 B)(1 + \Theta_1 B^4)\varepsilon_t.$$

All the factors can be multiplied out and the general model written as follows:

$$\begin{aligned} y_t = &(1 + \phi_1)y_{t-1} - \phi_1 y_{t-2} + (1 + \Phi_1)y_{t-4} \\ &- (1 + \phi_1 + \Phi_1 + \phi_1\Phi_1)y_{t-5} + (\phi_1 + \phi_1\Phi_1)y_{t-6} \\ &- \Phi_1 y_{t-8} + (\Phi_1 + \phi_1\Phi_1)y_{t-9} - \phi_1\Phi_1 y_{t-10} \\ &+ \varepsilon_t + \theta_1\varepsilon_{t-1} + \Theta_1\varepsilon_{t-4} + \theta_1\Theta_1\varepsilon_{t-5}. \end{aligned}$$

Common ARIMA models

The US Census Bureau uses the following models most often:

ARIMA(0,1,1)(0,1,1)$_m$	with log transformation
ARIMA(0,1,2)(0,1,1)$_m$	with log transformation
ARIMA(2,1,0)(0,1,1)$_m$	with log transformation
ARIMA(0,2,2)(0,1,1)$_m$	with log transformation
ARIMA(2,1,2)(0,1,1)$_m$	with no transformation

Seasonal ARIMA models

The seasonal part of an AR or MA model will be seen in the seasonal lags of the PACF and ACF.

ARIMA(0,0,0)(0,0,1)$_{12}$ will show:

- a spike at lag 12 in the ACF but no other significant spikes.
- The PACF will show exponential decay in the seasonal lags; that is, at lags 12, 24, 36, ...

ARIMA(0,0,0)(1,0,0)$_{12}$ will show:

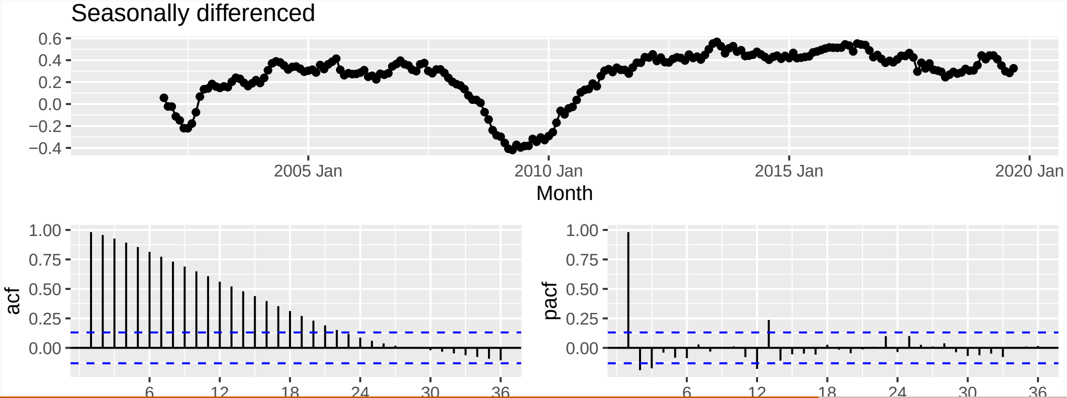
- exponential decay in the seasonal lags of the ACF
- a single significant spike at lag 12 in the PACF.

US leisure employment

```r
leisure <- us_employment %>%
  filter(Title == "Leisure and Hospitality",
         year(Month) > 2000) %>%
  mutate(Employed = Employed/1000) %>%
  select(Month, Employed)
autoplot(leisure, Employed) +
  labs(title = "US employment: leisure and hospitality",
       y="Number of people (millions)")
```

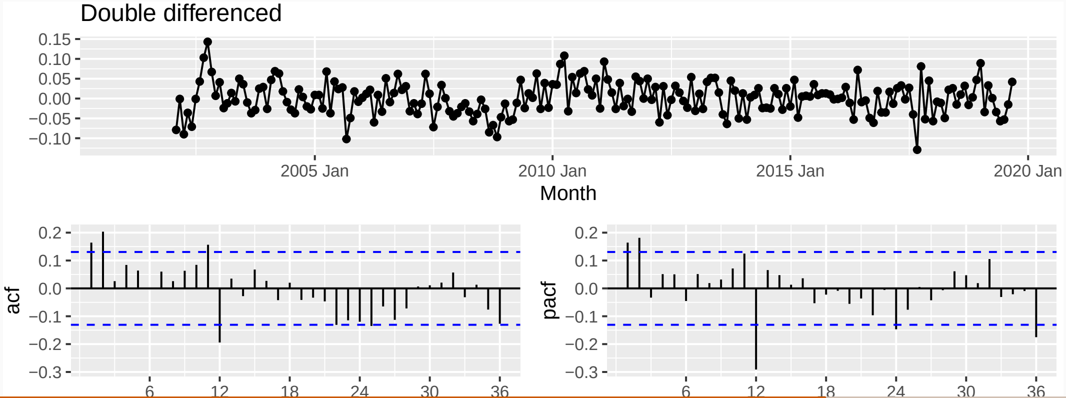
US leisure employment

```
leisure %>%
  gg_tsdisplay(difference(Employed, 12),
               plot_type='partial', lag=36) +
  labs(title="Seasonally differenced", y="")
```



US leisure employment

```
leisure %>%
  gg_tsdisplay(difference(Employed, 12) %>% difference(),
               plot_type='partial', lag=36) +
  labs(title = "Double differenced", y="")
```



US leisure employment

```r
fit <- leisure %>%
  model(
    arima012011 = ARIMA(Employed ~ pdq(0,1,2) + PDQ(0,1,1)),
    arima210011 = ARIMA(Employed ~ pdq(2,1,0) + PDQ(0,1,1)),
    auto = ARIMA(Employed, stepwise = FALSE, approx = FALSE)
  )
fit %>% pivot_longer(everything(), names_to = "Model name",
                     values_to = "Orders")
```

```
## # A mable: 3 x 2
## # Key:     Model name [3]
##   `Model name`                    Orders
##   <chr>                          <model>
## 1 arima012011  <ARIMA(0,1,2)(0,1,1)[12]>
## 2 arima210011  <ARIMA(2,1,0)(0,1,1)[12]>
## 3 auto         <ARIMA(2,1,0)(1,1,1)[12]>
```

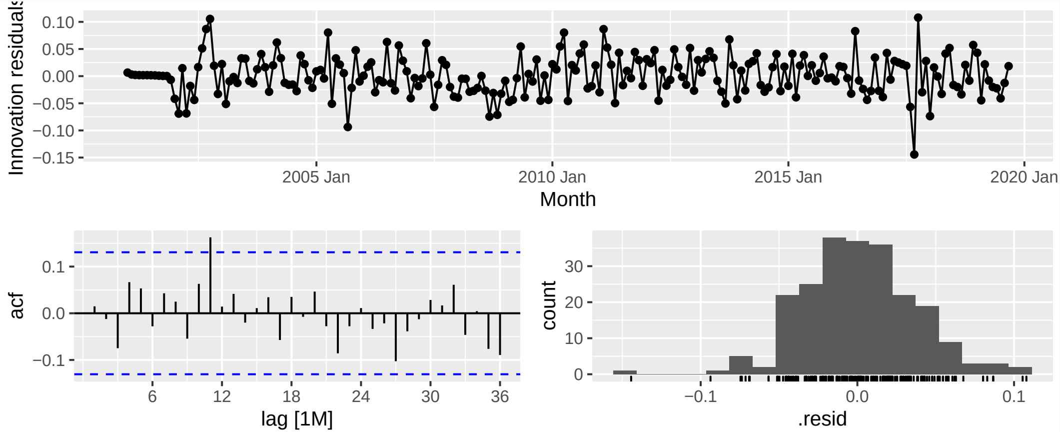
US leisure employment

```
glance(fit) %>% arrange(AICc) %>% select(.model:BIC)
```

```
## # A tibble: 3 x 6
##   .model       sigma2 log_lik   AIC  AICc   BIC
##   <chr>         <dbl>   <dbl> <dbl> <dbl> <dbl>
## 1 auto        0.00142    395. -780. -780. -763.
## 2 arima210011 0.00145    392. -776. -776. -763.
## 3 arima012011 0.00146    391. -775. -775. -761.
```

US leisure employment

```
fit %>% select(auto) %>% gg_tsresiduals(lag=36)
```



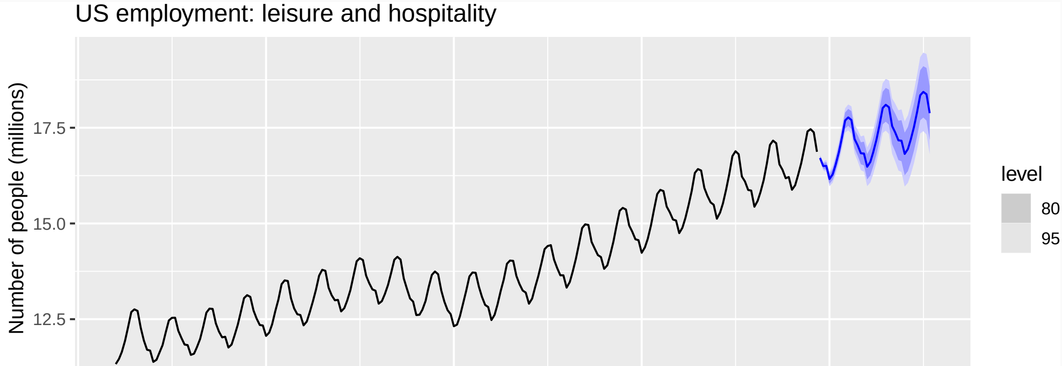
US leisure employment

```r
augment(fit) %>% features(.innov, ljung_box, lag=24, dof=4)
```

```
## # A tibble: 3 x 3
##   .model      lb_stat lb_pvalue
##   <chr>         <dbl>     <dbl>
## 1 arima012011    22.4     0.320
## 2 arima210011    18.9     0.527
## 3 auto           16.6     0.680
```

US leisure employment

```
forecast(fit, h=36) %>%
  filter(.model=='auto') %>%
  autoplot(leisure) +
  labs(title = "US employment: leisure and hospitality",
       y="Number of people (millions)")
```

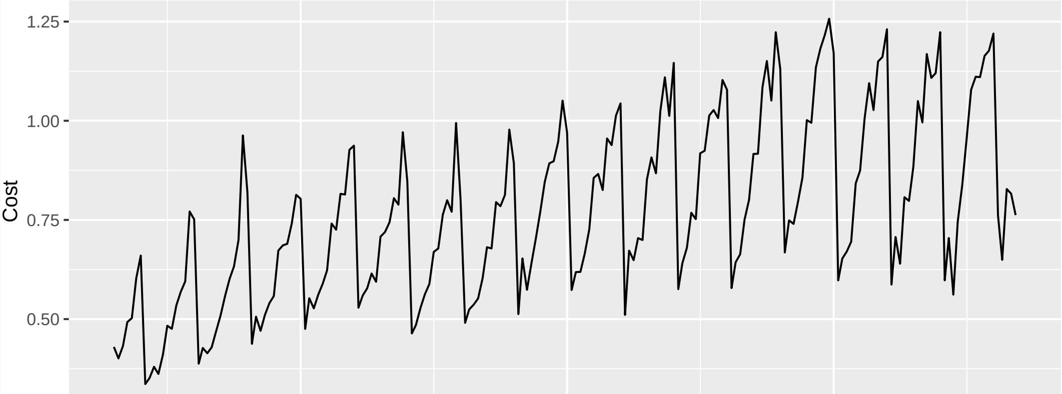


Cortecosteroid drug sales

```r
h02 <- PBS %>%
  filter(ATC2 == "H02") %>%
  summarise(Cost = sum(Cost)/1e6)
```

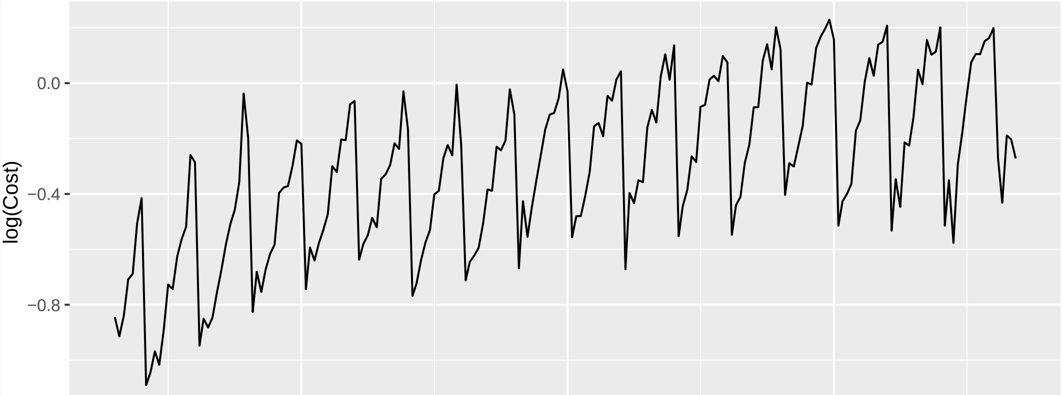
Cortecosteroid drug sales

```
h02 %>% autoplot(
  Cost
)
```



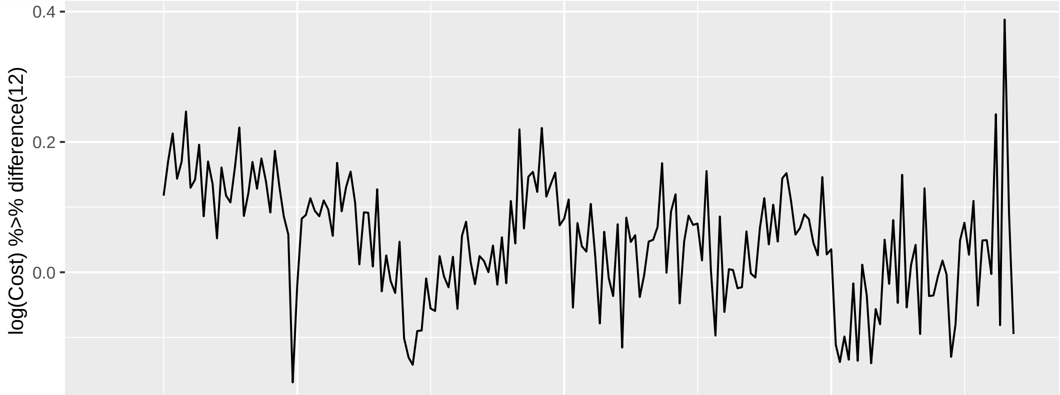
Cortecosteroid drug sales

```
h02 %>% autoplot(
  log(Cost)
)
```



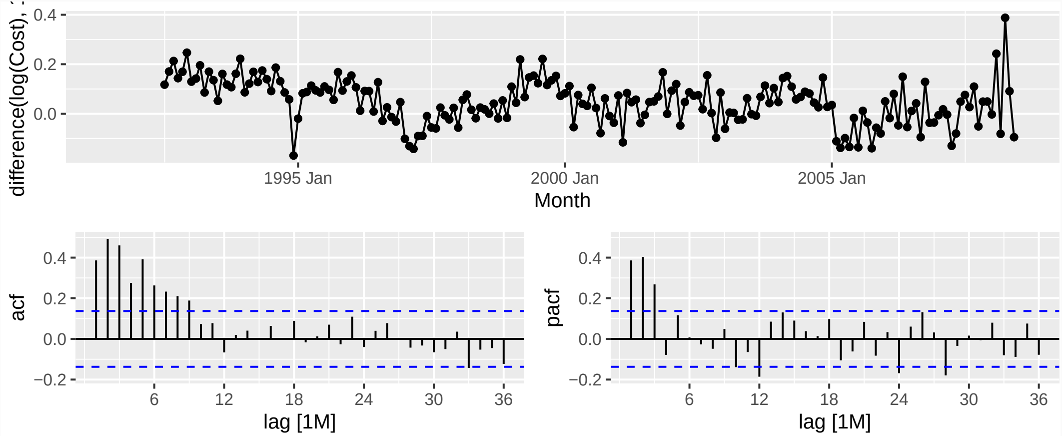
Cortecosteroid drug sales

```
h02 %>% autoplot(
  log(Cost) %>% difference(12)
)
```



Cortecosteroid drug sales

```
h02 %>% gg_tsdisplay(difference(log(Cost),12),
                     lag_max = 36, plot_type = 'partial')
```



Cortecosteroid drug sales

- Choose $D = 1$ and $d = 0$.
- Spikes in PACF at lags 12 and 24 suggest seasonal AR(2) term.
- Spikes in PACF suggests possible non-seasonal AR(3) term.
- Initial candidate model: ARIMA$(3,0,0)(2,1,0)_{12}$.

Cortecosteroid drug sales

.model	AICc
ARIMA(3,0,1)(0,1,2)[12]	-485
ARIMA(3,0,1)(1,1,1)[12]	-484
ARIMA(3,0,1)(0,1,1)[12]	-484
ARIMA(3,0,1)(2,1,0)[12]	-476
ARIMA(3,0,0)(2,1,0)[12]	-475
ARIMA(3,0,2)(2,1,0)[12]	-475
ARIMA(3,0,1)(1,1,0)[12]	-463

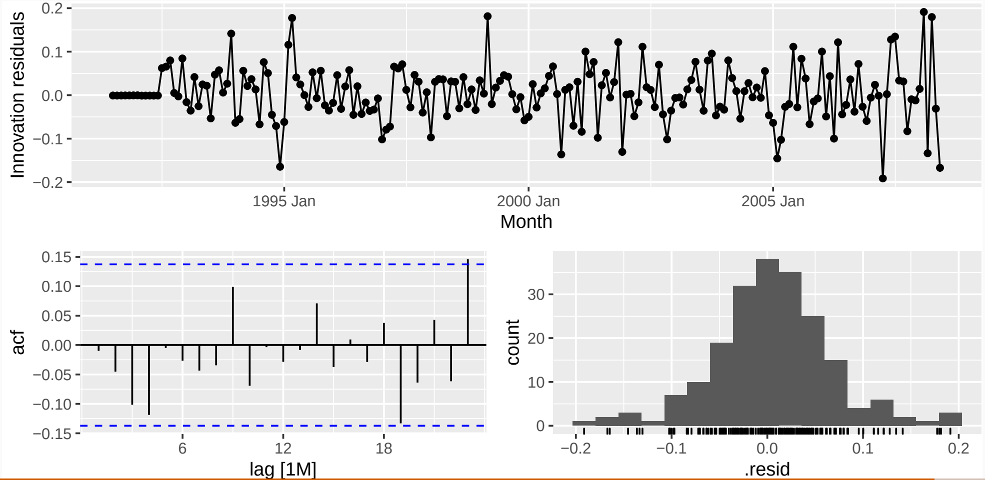
Cortecosteroid drug sales

```r
fit <- h02 %>%
  model(best = ARIMA(log(Cost) ~ 0 + pdq(3,0,1) + PDQ(0,1,2)))
report(fit)
```

```
## Series: Cost
## Model: ARIMA(3,0,1)(0,1,2)[12]
## Transformation: log(Cost)
##
## Coefficients:
##          ar1     ar2     ar3    ma1     sma1     sma2
##       -0.160  0.5481  0.5678  0.383  -0.5222  -0.1768
## s.e.   0.164  0.0878  0.0942  0.190   0.0861   0.0872
##
## sigma^2 estimated as 0.004278:  log likelihood=250
## AIC=-486   AICc=-485   BIC=-463
```

Cortecosteroid drug sales

```
gg_tsresiduals(fit)
```



Cortecosteroid drug sales

```r
augment(fit) %>%
  features(.innov, ljung_box, lag = 36, dof = 6)
```

```
## # A tibble: 1 x 3
##   .model lb_stat lb_pvalue
##   <chr>    <dbl>     <dbl>
## 1 best      50.7    0.0104
```

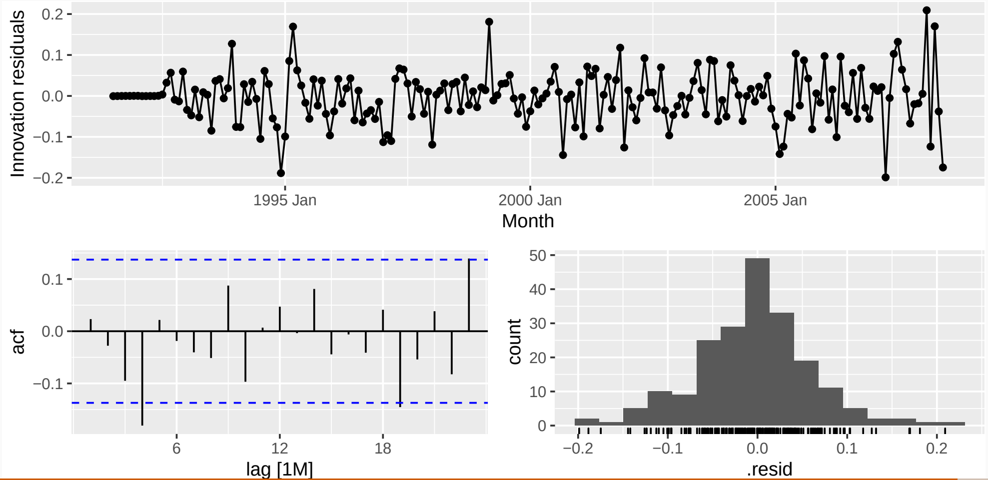
Cortecosteroid drug sales

```
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 0.004387:  log likelihood=245
## AIC=-483   AICc=-483   BIC=-470
```

Cortecosteroid drug sales

```
gg_tsresiduals(fit)
```



Cortecosteroid drug sales

```r
augment(fit) %>%
  features(.innov, ljung_box, lag = 36, dof = 3)
```

```
## # A tibble: 1 x 3
##   .model lb_stat lb_pvalue
##   <chr>    <dbl>     <dbl>
## 1 auto      59.3   0.00332
```

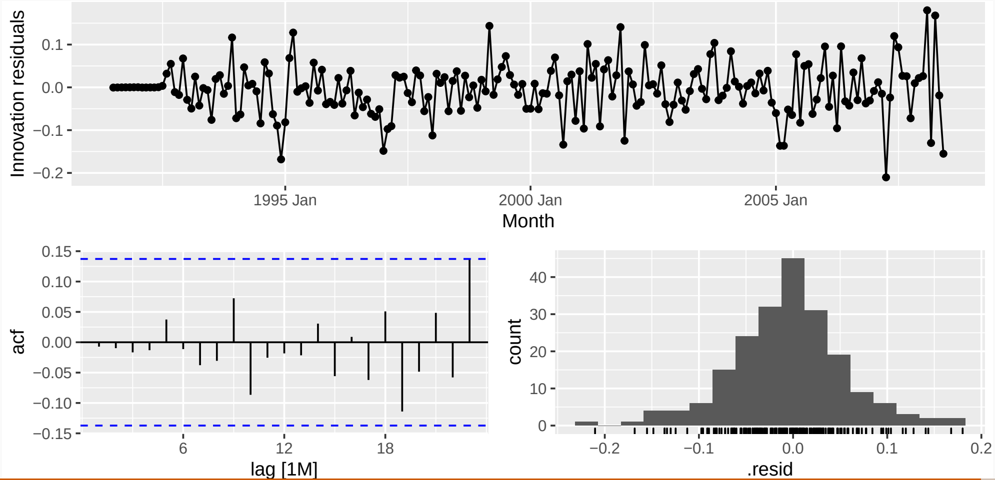
Cortecosteroid drug sales

```
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
##        -0.0425  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  0.213
##
## sigma^2 estimated as 0.004049:  log likelihood=254
## AIC=-489   AICc=-487   BIC=-456
```

Cortecosteroid drug sales

```
gg_tsresiduals(fit)
```



Cortecosteroid drug sales

```r
augment(fit) %>%
  features(.innov, ljung_box, lag = 36, dof = 9)
```

```
## # A tibble: 1 x 3
##   .model lb_stat lb_pvalue
##   <chr>    <dbl>     <dbl>
## 1 best      36.5     0.106
```

Cortecosteroid drug sales

Training data: July 1991 to June 2006

Test data: July 2006–June 2008

```
fit <- h02 %>%
  filter_index(~ "2006 Jun") %>%
  model(
    ARIMA(log(Cost) ~ 0 + pdq(3, 0, 0) + PDQ(2, 1, 0)),
    ARIMA(log(Cost) ~ 0 + pdq(3, 0, 1) + PDQ(2, 1, 0)),
    ARIMA(log(Cost) ~ 0 + pdq(3, 0, 2) + PDQ(2, 1, 0)),
    ARIMA(log(Cost) ~ 0 + pdq(3, 0, 1) + PDQ(1, 1, 0))
    # ... #
  )

fit %>%
  forecast(h = "2 years") %>%
```

Cortecosteroid drug sales

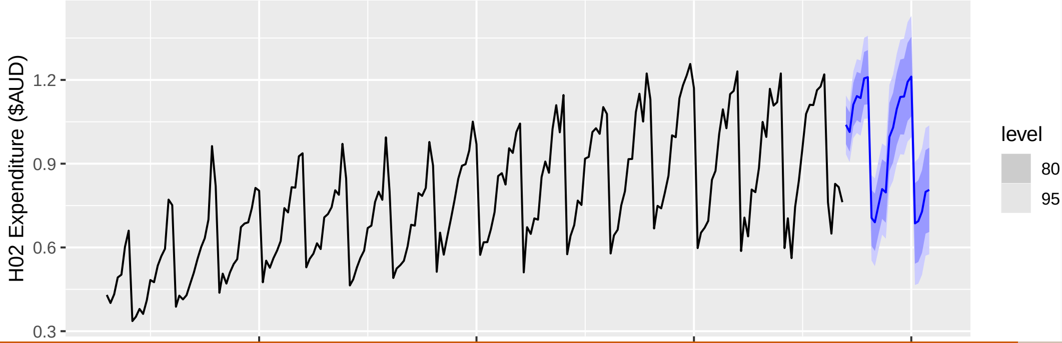
.model	RMSE
ARIMA(3,0,1)(1,1,1)[12]	0.0619
ARIMA(3,0,1)(0,1,2)[12]	0.0621
ARIMA(3,0,1)(0,1,1)[12]	0.0630
ARIMA(2,1,0)(0,1,1)[12]	0.0630
ARIMA(4,1,1)(2,1,2)[12]	0.0631
ARIMA(3,0,2)(2,1,0)[12]	0.0651
ARIMA(3,0,1)(2,1,0)[12]	0.0653
ARIMA(3,0,1)(1,1,0)[12]	0.0666
ARIMA(3,0,0)(2,1,0)[12]	0.0668

Cortecosteroid drug sales

- Models with lowest AICc values tend to give slightly better results than the other models.
- AICc comparisons must have the same orders of differencing. But RMSE test set comparisons can involve any models.
- Use the best model available, even if it does not pass all tests.

Cortecosteroid drug sales

```
fit <- h02 %>%
  model(ARIMA(Cost ~ 0 + pdq(3,0,1) + PDQ(0,1,2)))
fit %>% forecast %>% autoplot(h02) +
  labs(y = "H02 Expenditure ($AUD)")
```

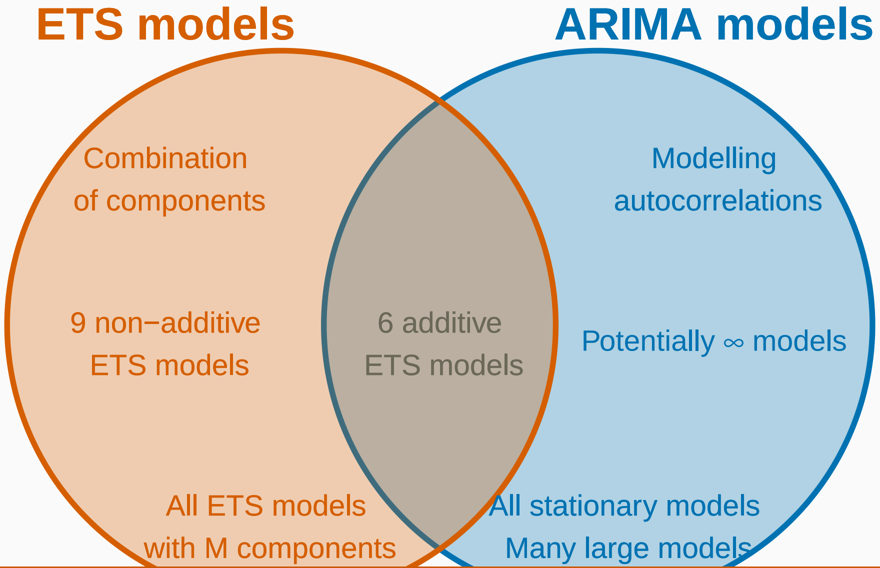


Outline

ARIMA vs ETS

- Myth that ARIMA models are more general than exponential smoothing.
- Linear exponential smoothing models all special cases of ARIMA models.
- Non-linear exponential smoothing models have no equivalent ARIMA counterparts.
- Many ARIMA models have no exponential smoothing counterparts.
- ETS models all non-stationary. Models with seasonality or non-damped trend (or both) have two unit roots; all other models have one unit root.

ARIMA vs ETS


ETS models
ARIMA models
Combination
of components
9 non−additive
ETS models
All ETS models
with M components
6 additive
ETS models
Modelling
autocorrelations
Potentially ∞ models
All stationary models
Many large models

Equivalences

ETS model	**ARIMA model**	**Parameters**
ETS(A,N,N)	ARIMA(0,1,1)	$\theta_1 = \alpha - 1$
ETS(A,A,N)	ARIMA(0,2,2)	$\theta_1 = \alpha + \beta - 2$
		$\theta_2 = 1 - \alpha$
ETS(A,A_d,N)	ARIMA(1,1,2)	$\phi_1 = \phi$
		$\theta_1 = \alpha + \phi\beta - 1 - \phi$
		$\theta_2 = (1 - \alpha)\phi$
ETS(A,N,A)	ARIMA(0,0,m)(0,1,0)$_m$	
ETS(A,A,A)	ARIMA(0,1,$m+1$)(0,1,0)$_m$	
ETS(A,A_d,A)	ARIMA(1,0,$m+1$)(0,1,0)$_m$	

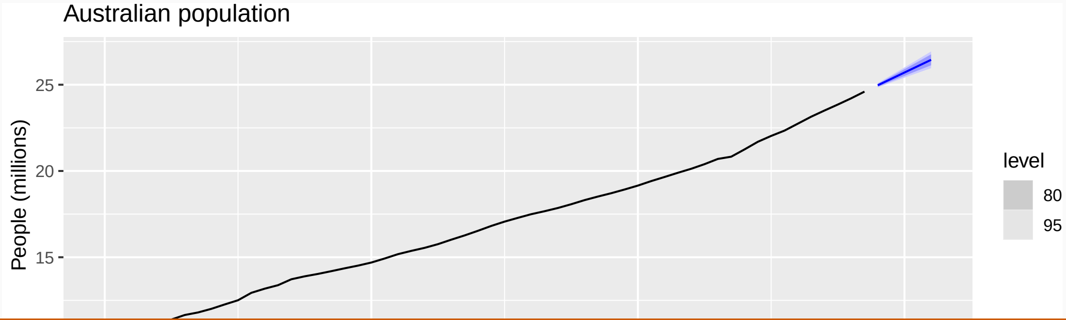
Example: Australian population

```
aus_economy <- global_economy %>% filter(Code == "AUS") %>%
  mutate(Population = Population/1e6)
aus_economy %>%
  slice(-n()) %>%
  stretch_tsibble(.init = 10) %>%
  model(eta = ETS(Population),
        arima = ARIMA(Population)
  ) %>%
  forecast(h = 1) %>%
  accuracy(aus_economy) %>%
  select(.model, ME:RMSSE)
```

```
## # A tibble: 2 x 8
##   .model      ME   RMSE    MAE    MPE   MAPE   MASE RMSSE
##   <chr>    <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl> <dbl>
## 1 arima   0.0420 0.194  0.0789 0.277  0.509  0.317 0.746
```

Example: Australian population

```
aus_economy %>%
  model(ETS(Population)) %>%
  forecast(h = "5 years") %>%
  autoplot(aus_economy) +
  labs(title = "Australian population",
       y = "People (millions)")
```



Example: Cement production

```r
cement <- aus_production %>%
  select(Cement) %>%
  filter_index("1988 Q1" ~ .)
train <- cement %>% filter_index(. ~ "2007 Q4")
fit <- train %>%
  model(
    arima = ARIMA(Cement),
    ets = ETS(Cement)
  )
```

Example: Cement production

```
fit %>%
  select(arima) %>%
  report()
```

```
## Series: Cement
## Model: ARIMA(1,0,1)(2,1,1)[4] w/ drift
##
## Coefficients:
##          ar1     ma1   sar1    sar2    sma1  constant
##       0.8886  -0.237  0.081  -0.234  -0.898      5.39
## s.e.  0.0842   0.133  0.157   0.139   0.178      1.48
##
## sigma^2 estimated as 11456:  log likelihood=-464
## AIC=941   AICc=943   BIC=957
```

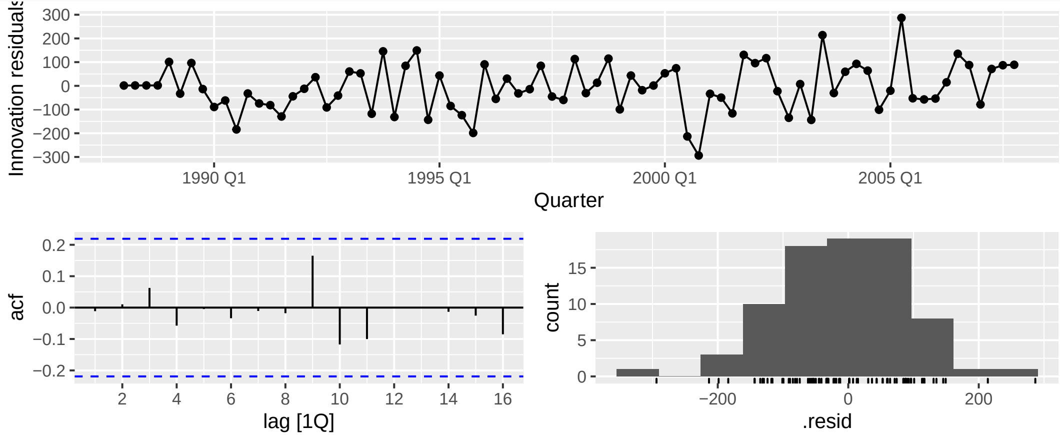
Example: Cement production

```
fit %>%
  select(ets) %>%
  report()
```

```
## Series: Cement
## Model: ETS(M,N,M)
##   Smoothing parameters:
##     alpha = 0.753
##     gamma = 1e-04
##
##   Initial states:
##  l[0] s[0] s[-1] s[-2] s[-3]
##  1695 1.03  1.05  1.01 0.912
##
##   sigma^2:  0.0034
##
##  AIC AICc  BIC
```

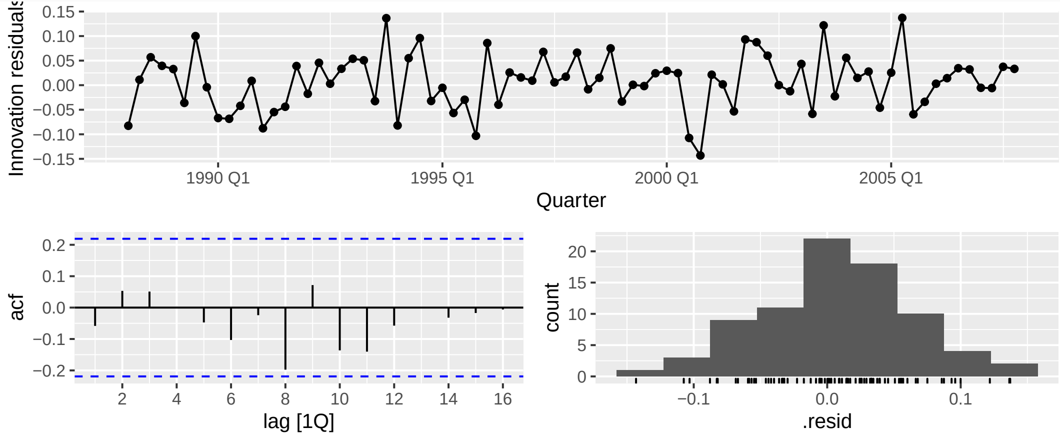
Example: Cement production

```
gg_tsresiduals(fit %>% select(arima), lag_max = 16)
```



Example: Cement production

```
gg_tsresiduals(fit %>% select(ets), lag_max = 16)
```



Example: Cement production

```
fit %>%
  select(arima) %>%
  augment() %>%
  features(.innov, ljung_box, lag = 16, dof = 6)
```

```
## # A tibble: 1 x 3
##   .model lb_stat lb_pvalue
##   <chr>    <dbl>     <dbl>
## 1 arima     6.37     0.783
```

Example: Cement production

```
fit %>%
  select(ets) %>%
  augment() %>%
  features(.innov, ljung_box, lag = 16, dof = 6)
```

```
## # A tibble: 1 x 3
##   .model lb_stat lb_pvalue
##   <chr>    <dbl>     <dbl>
## 1 ets       10.0     0.438
```

Example: Cement production

```
fit %>%
  forecast(h = "2 years 6 months") %>%
  accuracy(cement) %>%
  select(-ME, -MPE, -ACF1)
```

```
## # A tibble: 2 x 7
##   .model .type  RMSE   MAE  MAPE  MASE RMSSE
##   <chr>  <chr> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 arima  Test   216.  186.  8.68  1.27  1.26
## 2 ets    Test   222.  191.  8.85  1.30  1.29
```

Example: Cement production

```
fit %>%
  select(arima) %>%
  forecast(h="3 years") %>%
  autoplot(cement) +
  labs(title = "Cement production in Australia",
       y="Tonnes ('000)")
```