Applied Economic Forecasting

7. ARIMA Models

1. Stationarity and Differencing
2. AR Models
3. MA Models
4. ARIMA Modeling
5. Nonseasonal ARIMA Models
6. Seasonal ARIMA Models

Section 1

Stationarity and Differencing

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.

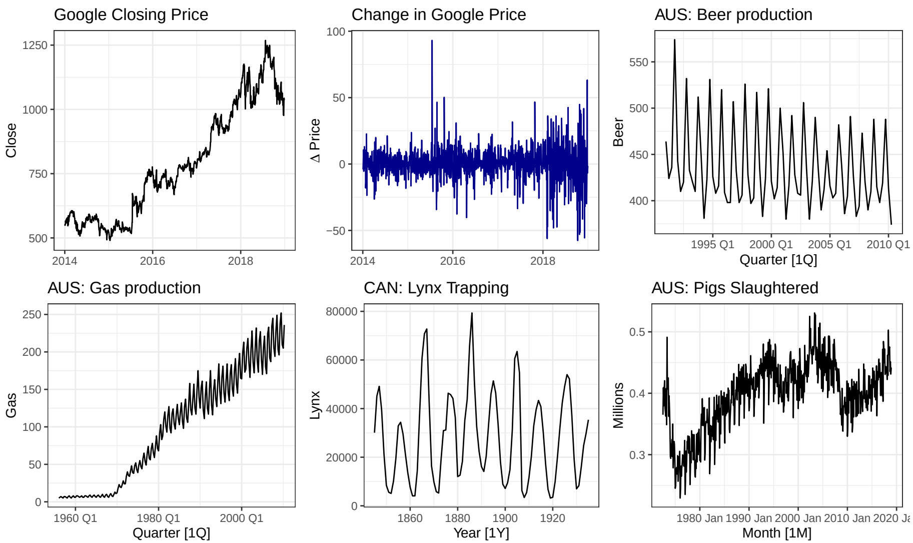
A **stationary series** is:

- constant mean
- constant variance
- no patterns predictable in the long-term
- No Trend
- No Seasonality

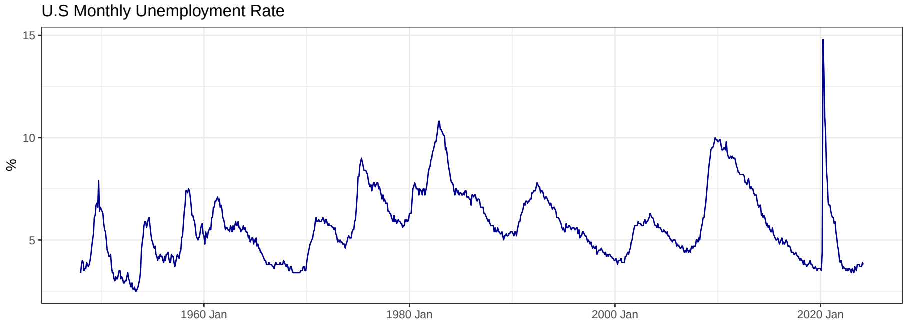
A Note on Cyclic Series

A time series with cyclic behavior (but with no trend or seasonality) is stationary. This is because the cycles are not of a fixed length, so we are unable to predict where the peaks and troughs of the cycles will be.

Stationary?


Google Closing Price
Close
1250
1000
750
500
2014
2016
2018
Change in Google Price
Δ Price
100
50
0
−50
2014
2016
2018
AUS: Beer production
Beer
550
500
450
400
1995 Q1
2000 Q1
2005 Q1
2010 Q1
Quarter [1Q]
AUS: Gas production
Gas
250
200
150
100
50
0
1960 Q1
1980 Q1
2000 Q1
Quarter [1Q]
CAN: Lynx Trapping
Lynx
80000
60000
40000
20000
0
1860
1880
1900
1920
Year [1Y]
AUS: Pigs Slaughtered
Millions
0.5
0.4
0.3
1980 Jan
1990 Jan
2000 Jan
2010 Jan
2020 Ja
Month [1M]

Stationary?



Source: FRED

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.

- The transformations from earlier modules help to **stabilize the variance**.
- For ARIMA modeling, 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.
  - Think back to the ACF of a series with a trend.
- For non-stationary data, the value of r_1 (the first lag of the ACF) is often large and positive.

Non-stationarity in the mean: Differencing

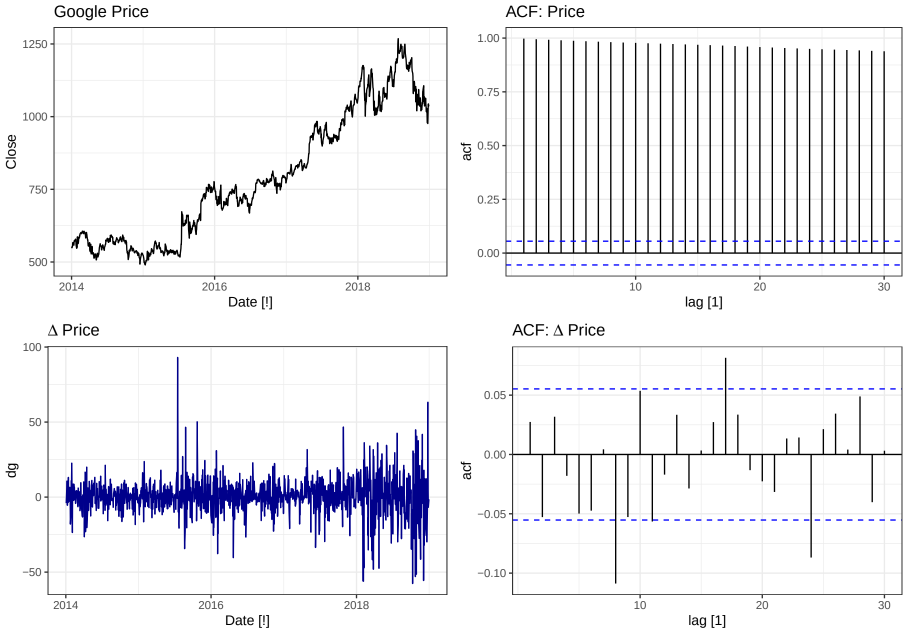
- Differencing helps to **stabilize the mean**.
- The differenced series is the *change* between each observation in the original series:

$$\Delta y_t = y_t - y_{t-1}.$$

- The differenced series will have only $T - 1$ values since it is not possible to calculate a difference, Δy_1, for the first observation.

Non-stationarity in the mean: Differencing

```r
goog <- gafa_stock %>% filter(Symbol == "GOOG") %>%
  #First Diff
  mutate(dg = difference(Close))

p1 <- goog %>% autoplot(Close) + ggtitle("Google Price")
p2 <- goog %>% autoplot(dg, col = "blue4") +
  ggtitle(bquote(Delta~"Price"))

## ACFs
p3 <- goog %>% ACF(Close) %>% autoplot() +
  ggtitle("ACF: Price")
p4 <- goog %>% ACF(dg) %>% autoplot() +
  ggtitle(bquote("ACF:"~Delta~"Price"))

gridExtra::grid.arrange(p1,p3,p2,p4, ncol = 2)
```



Google Price
Close
1250
1000
750
500
2014
2016
2018
Date [!]
ACF: Price
acf
1.00
0.75
0.50
0.25
0.00
10
20
30
lag [1]
Δ Price
dg
100
50
0
−50
2014
2016
2018
Date [!]
ACF: Δ Price
acf
0.05
0.00
−0.05
−0.10
10
20
30
lag [1]

Non-stationarity in the mean: Differencing

Random Walk Model

When the differenced series is white noise, the model for the original series can be written as:

$$y_t - y_{t-1} = \varepsilon_t$$

where ε_t denotes white noise. Rearranging this leads to the “random walk” model:

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

Random walk models are widely used for non-stationary data, particularly financial and economic data. Random walks typically have:

- long periods of apparent trends up or down
- sudden and unpredictable changes in direction.

Non-stationarity in the mean: Differencing

Random Walk w. Drift

A closely related model allows the differences to have a non-zero mean. Then

$$y_t = c + y_{t-1} + \varepsilon_t \quad \text{OR} \quad y_t - y_{t-1} = c + \varepsilon_t$$

c represents the average of the changes between consecutive observations.

- c is positive, then the average change is an increase in the value of y_t. Thus, y_t will tend to drift upwards.
- If c is negative, y_t will tend to drift downwards.

Non-stationarity in the mean: Differencing

Second-order differencing

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

$$\begin{aligned}\Delta^2 y_t &= \Delta y_t - \Delta y_{t-1}\\ &= (y_t - y_{t-1}) - (y_{t-1} - y_{t-2})\\ &= y_t - 2y_{t-1} + y_{t-2}.\end{aligned}$$

- $\Delta^2 y_t$ 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 that season in the previous year**.

$$\Delta_m y_t = y_t - y_{t-m}$$

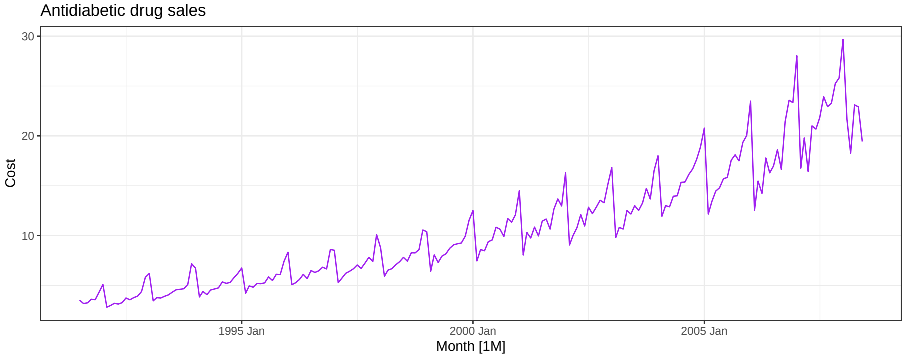
where $m =$ number of seasons.

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

Seasonal differencing

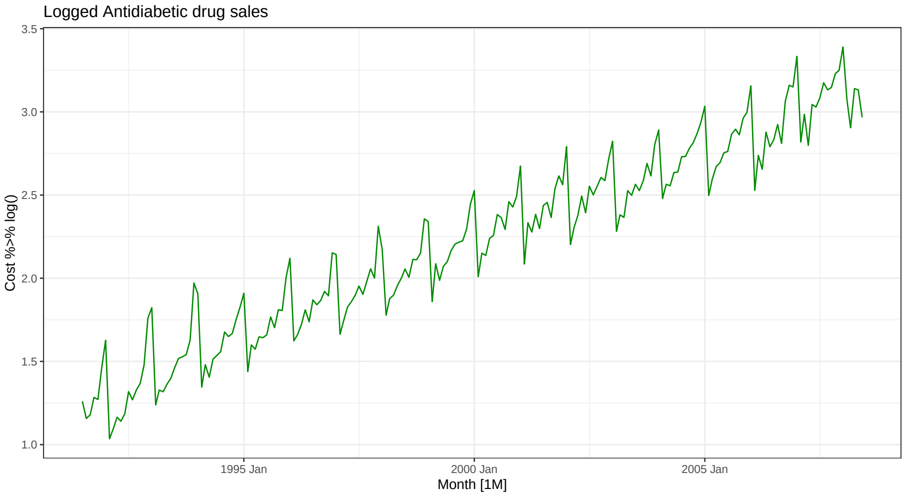
Stable Variance?

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

a10 %>% autoplot(Cost, col = "purple") + ggtitle("Antidiabetic drug sales")
```



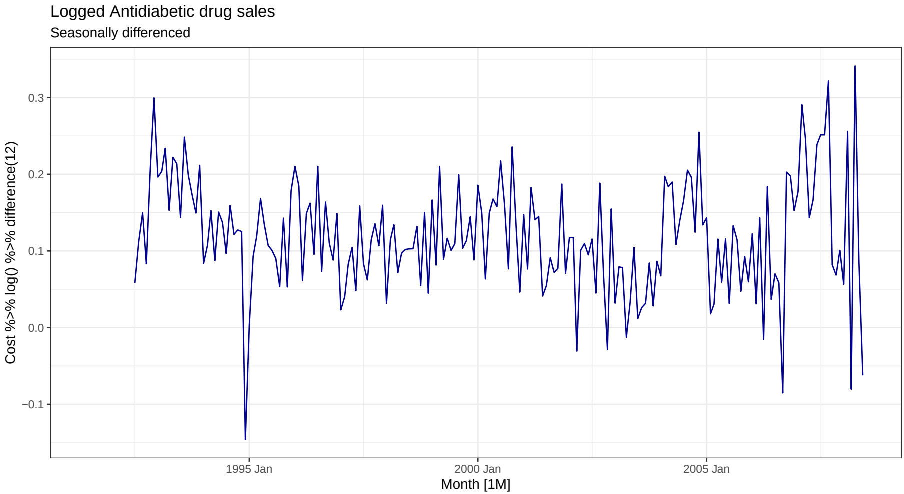
Seasonal differencing

```
a10 %>% autoplot(Cost %>% log(), col = "green4") +
  ggtitle("Logged Antidiabetic drug sales")
```



Seasonal differencing

```r
a10 %>% autoplot(Cost %>% log() %>% difference(12),
                 col = "darkblue") +
  labs(title = "Logged Antidiabetic drug sales",
       subtitle = "Seasonally differenced")
```



Seasonal differencing

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

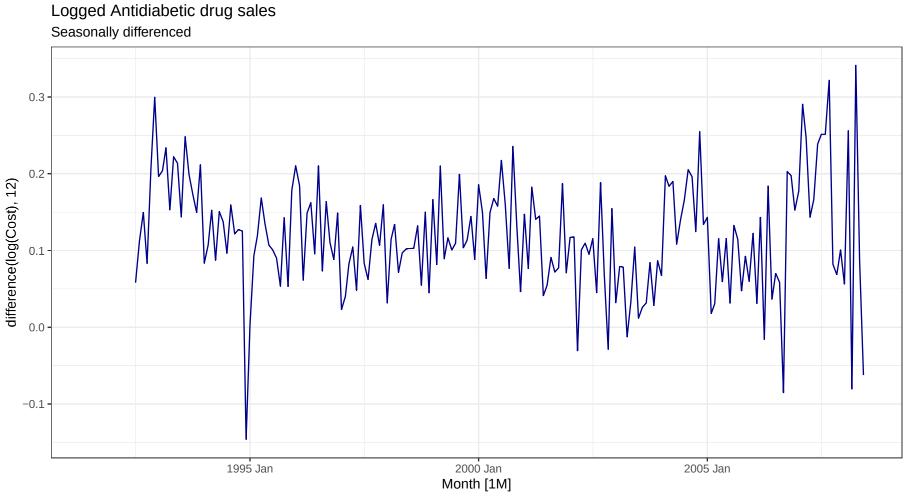
If $\Delta_m y_t = y_t - y_{t-12}$ denotes seasonally differenced series, then a twice-differenced series takes the form:

$$\begin{aligned} y_t^* &= \Delta_m y_t - \Delta_m^2 y_{t-1} \\ &= (y_t - y_{t-12}) - (y_{t-1} - y_{t-13}) \\ &= y_t - y_{t-1} - y_{t-12} + y_{t-13}\,. \end{aligned}$$

$$\text{Generic Form } y_t^* = y_t - y_{t-1} - y_{t-m} + y_{t-m-1}$$

Seasonal differencing

```
a10 %>% autoplot(difference(log(Cost), 12), col = "darkblue") +
  labs(title = "Logged Antidiabetic drug sales",
       subtitle = "Seasonally differenced")
```



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, it is recommended 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**.

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 (ADF) 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.

ADF test

KPSS test

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

```
## # A tibble: 1 x 3
##   Symbol kpss_stat kpss_pvalue
##   <chr>      <dbl>       <dbl>
## 1 GOOG        14.8        0.01
```

Takeaway: The p-value of 0.01 suggest that we can reject the null (of stationarity) at the 1% significance level. This implies that the variable is actually nonstationary and will need to be differenced.

Note

- By default, p-values smaller than 0.01 will be reported as 0.01 exactly.
- Those p-values larger than 0.1 will be reported as 0.1 exactly.

KPSS test

```r
goog %>% features(Close %>% difference(), unitroot_kpss)
```

```
## # A tibble: 1 x 3
##   Symbol kpss_stat kpss_pvalue
##   <chr>      <dbl>       <dbl>
## 1 GOOG      0.0529         0.1
```

Takeaways: The pvalue of the KPSS test is rather large. Since it is greater than 10%, we fail to reject the null of stationarity and conclude that the differenced series does appear to be stationary.

KPSS test

Determine the optimal number of times to difference using package.

Non-Seasonal Differencing:

```
goog %>% features(Close, unitroot_ndiffs)
```

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

Seasonal Differencing:

```
a10 %>% features(Cost %>% log(), unitroot_nsdiffs)
```

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

KPSS test

Was it necessary to doubly difference the a10 series earlier?

```
a10 <- a10 %>% mutate(l.cost = log(Cost))
# Seasonal Difference
a10 %>% features(l.cost, unitroot_nsdiffs)
```

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

```
# First Difference of the seasonally differenced series
a10 %>% features(l.cost %>% difference(12), unitroot_ndiffs)
```

```
## # A tibble: 1 x 1
##   ndiffs
##    <int>
## 1      0
```

Your turn!

Using the codes below, determine the appropriate differencing, **after variance stabilization, if necessary**, that must be done to the Turnover series to render it stationary.

```
aus_retail <- aus_retail %>%
  summarise(Turnover = sum(Turnover))
```

Section 2

AR Models

AR Models

In an autoregression model, we forecast our variable of interest (y_t) using a linear combination of **past values** of that variable.

Therefore, the term autoregression indicates that it is a regression of the variable against itself.

An autoregressive model of lag order p can be expressed as:

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

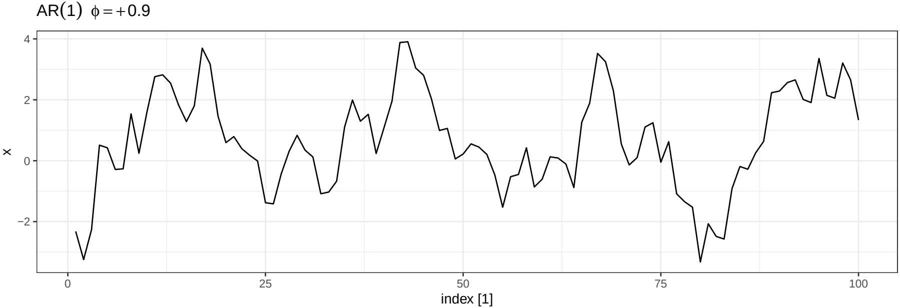
where ε_t is white noise.

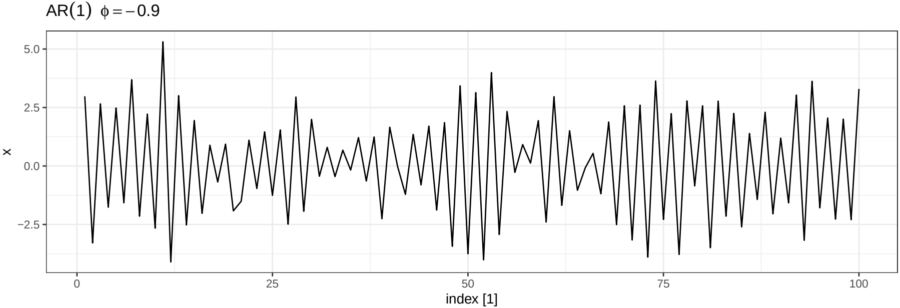
AR(1) model

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

- 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**.

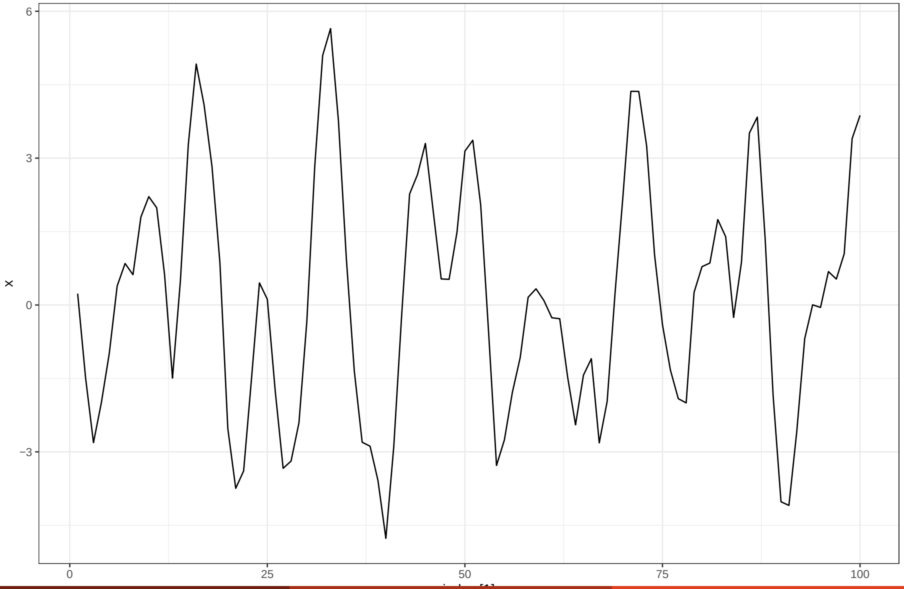
Sample of an AR(1) Process





Sample of an AR(2) Process

AR(2) $\phi_1 = -1.3$ $\phi_2 = -0.7$



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.

- For $p = 1$: $-1 < \phi_1 < 1$.
- For $p = 2$:

$$-1 < \phi_2 < 1, \quad \phi_2 + \phi_1 < 1, \quad \phi_2 - \phi_1 < 1.$$

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

Section 3

MA Models

Moving Averages (MA)

Moving Average (MA) models:

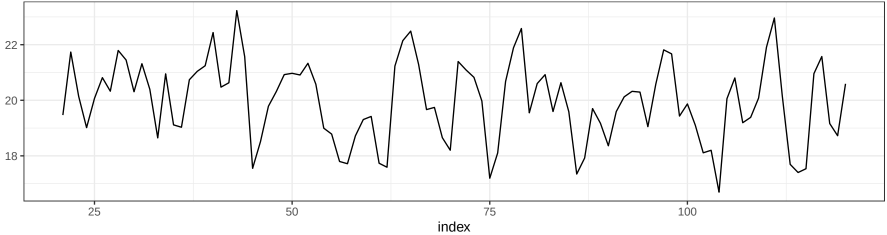
$$y_t = c + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \cdots + \theta_q \varepsilon_{t-q},$$

where ε_t is white noise.
This is a multiple regression with ***past errors*** as predictors.

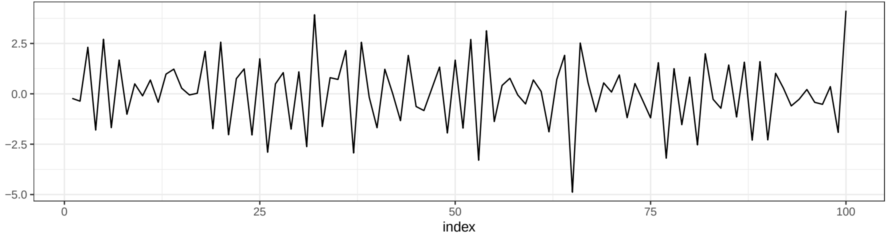
Do not confuse this with moving average smoothing!

Sample MA Models

MA(1): $y_t = 20 + \varepsilon_t + 0.8\varepsilon_{t-1}$



MA(2): $y_t = \varepsilon_t - \varepsilon_{t-1} + 0.8\varepsilon_{t-2}$



MA(∞) models

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

Example: AR(1)

$$\begin{aligned} y_t &= \phi_1 y_{t-1} + \varepsilon_t \\ &= \phi_1(\phi_1 y_{t-2} + \varepsilon_{t-1}) + \varepsilon_t \\ &= \phi_1^2 y_{t-2} + \phi_1 \varepsilon_{t-1} + \varepsilon_t \\ &= \phi_1^3 y_{t-3} + \phi_1^2 \varepsilon_{t-2} + \phi_1 \varepsilon_{t-1} + \varepsilon_t \\ &\dots \\ y_t &= \phi^k y_{t-k} + \sum_{k=0}^{\infty} \phi^k \varepsilon_{t-k}. \end{aligned}$$

Provided $-1 < \phi_1 < 1$:

$$y_t = \varepsilon_t + \phi_1 \varepsilon_{t-1} + \phi_1^2 \varepsilon_{t-2} + \phi_1^3 \varepsilon_{t-3} + \cdots$$

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.

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.

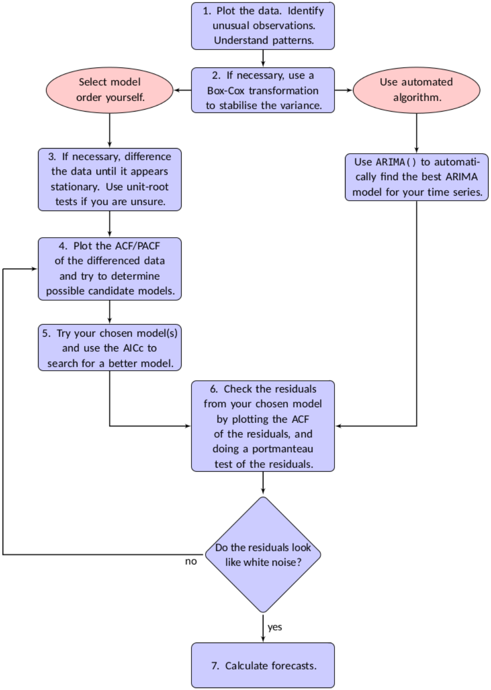
- For $q = 1$: $-1 < \theta_1 < 1$.
- For $q = 2$:
  $-1 < \theta_2 < 1 \qquad \theta_2 + \theta_1 > -1 \qquad \theta_1 - \theta_2 < 1$.
- More complicated conditions hold for $q \geq 3$.
- Estimation software takes care of this.

Section 4

ARIMA Modeling

Box-Jenkins Methodology

The Box-Jenkins methodology refers to a set of procedures for identifying, fitting, and checking ARIMA models with time series data. Forecasts follow directly from the form of the fitted model.


1. Plot the data. Identify unusual observations. Understand patterns.
2. If necessary, use a Box-Cox transformation to stabilise the variance.
Select model order yourself.
Use automated algorithm.
3. If necessary, difference the data until it appears stationary. Use unit-root tests if you are unsure.
Use ARIMA() to automatically find the best ARIMA model for your time series.
4. Plot the ACF/PACF of the differenced data and try to determine possible candidate models.
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.
Do the residuals look like white noise?
no
yes
7. Calculate forecasts.

Box-Jenkins Methodology

- Autoregressive integrated moving average (ARIMA) models
  - a class of linear models that are capable of representing ***stationary*** as well as ***nonstationary*** time series.
  - it generates forecasts based on a description of ***historical patterns in the data itself,***

$$Y_t = \beta_0 + \underbrace{\beta_1 Y_{t-1} + \beta_2 Y_{t-2} + \cdots + \beta_k Y_{t-p}}_{\text{autoregressive terms}}$$

$$+ \varepsilon_t \underbrace{-\omega_1 \varepsilon_{t-1} - \omega_2 \varepsilon_{t-2} - \cdots - \omega_q \varepsilon_{t-q}}_{\text{moving average terms}}$$

Box-Jenkins Methodology

Nonstationary time series data

1. The mean of the data is function of time

$$E(Y_t) = a + b \cdot t$$

so-called first moment nonstationary.

- A unit root (random walk)

$$Y_t = Y_{t-1} + \varepsilon_t$$

Stationary time series data

- The mean of the data is independent of time (i.e., constant),

$$E\left(Y_t\right) = a$$

so-called first moment stationary.

Box-Jenkins Methodology

- The initial selection of an ARIMA model is based on

  1. Autocorrelation function (ACF) ==> **Will help us to determine the moving average (MA) component.**

$$r_k = \frac{cov\left(Y_t, Y_{t-k}\right)}{\sqrt{var\left(Y_t\right)}\sqrt{var\left(Y_{t-k}\right)}}$$

2. Partial autocorrelation function (PACF) ==> **Will help us to determine the autoregressive (AR) component.**

$$Y_t = \beta_0 + \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + \cdots + \beta_k Y_{t-k} + \varepsilon_t$$

such that

$$r_k = \beta_k$$

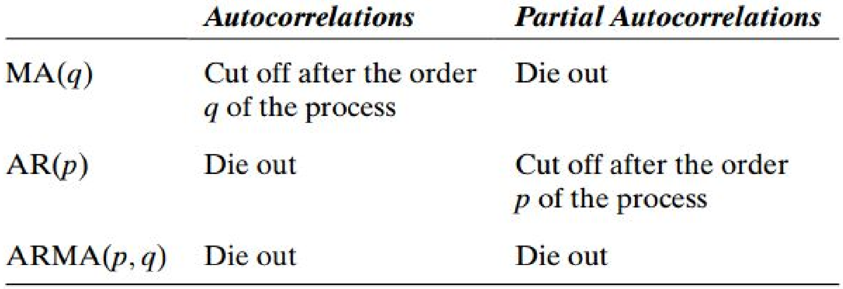
Box-Jenkins Methodology

Distance k	Regression	Partial Autocorrelation coefficient r_k
1	$Y_t = \beta_0 + \beta_1 Y_{t-1} + \varepsilon_t$	β_1
2	$Y_t = \beta_0 + \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + \varepsilon_t$	β_2
3	$Y_t = \beta_0 + \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + \beta_3 Y_{t-3} + \varepsilon_t$	β_3
…	…	…
k	$Y_t = \beta_0 + \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + \cdots + \beta_k Y_{t-k} + \varepsilon_t$	β_k

Section 5

Nonseasonal ARIMA Models

Identifying Nonseasonal AR and MA Models



	Autocorrelations	*Partial Autocorrelations*
MA(q)	Cut off after the order q of the process	Die out
AR(p)	Die out	Cut off after the order p of the process
ARMA(p, q)	Die out	Die out

Let us explore some simulations and test our theory!

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.

Autoregressive Integrated Moving Average models

- Combine ARMA model with **differencing**.

Nonseasonal 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.

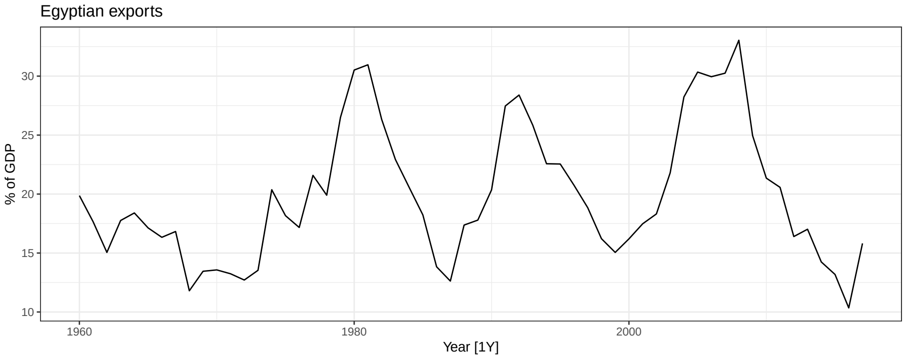
Special cases of Nonseasonal ARIMA Models

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

ARIMA in Motion

Step (1): Plot Data

```
egy <- global_economy |>
  filter(Code == "EGY")

egy |>  autoplot(Exports) +
  labs(y = "% of GDP", title = "Egyptian exports")
```



ARIMA in Motion

Step 2: Stabilize Variance?

ARIMA in Motion

Step 3: Check if Stationary

```
egy |> features(Exports, unitroot_kpss)
```

```
## # A tibble: 1 x 3
##   Country              kpss_stat kpss_pvalue
##   <fct>                    <dbl>       <dbl>
## 1 Egypt, Arab Rep.         0.192         0.1
```

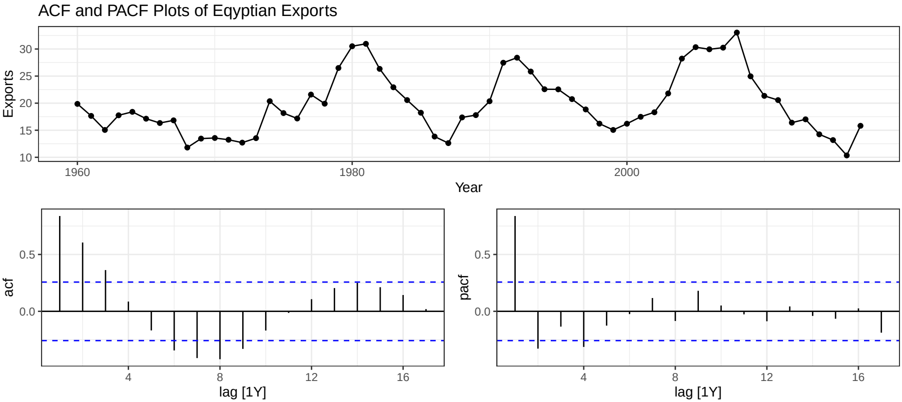
Key Takeaways:

- The large p-value of the `KPSS` test suggests that we **fail to reject** the null of stationarity at all conventional levels of significance.
- Our unit root test therefore, suggests that differencing is not necessary.
  - If you were not able to see this at the outset, look again and notice that the series seems to represent somewhat of a cycle in Exports. **Recall from earlier, that cycles are stationary!**

ARIMA in Motion

Step 4: Plot ACF and PACFs to determine potential model candidates.

```
egy |> gg_tsdisplay(Exports, plot_type = c("partial")) +
  labs(title = "ACF and PACF Plots of Eqyptian Exports")
```



ARIMA in Motion

Takeaways

The ACF decays in a ‘sinusoidal‘ pattern while the last significant lag in the PACF is lag 4. We could explore the possibility of an AR(4) model. For this exercise, we will assume that there is only 1 candidate model.

```
mod.fit <- egy |> model(ARIMA(Exports ~ pdq(4,0,0)))
mod.fit |> report()
```

```
## Series: Exports
## Model: ARIMA(4,0,0) w/ mean
##
## Coefficients:
##          ar1      ar2     ar3      ar4  constant
##       0.9861  -0.1715  0.1807  -0.3283    6.6922
## s.e.  0.1247   0.1865  0.1865   0.1273    0.3562
##
## sigma^2 estimated as 7.885:  log likelihood=-140.5
## AIC=293.1   AICc=294.7   BIC=305.4
```

ARIMA in Motion

What about other potential models? How about the best ARIMA models with $p \in \{1,2,4\}$ and $q \in \{0,1,2,3\}$

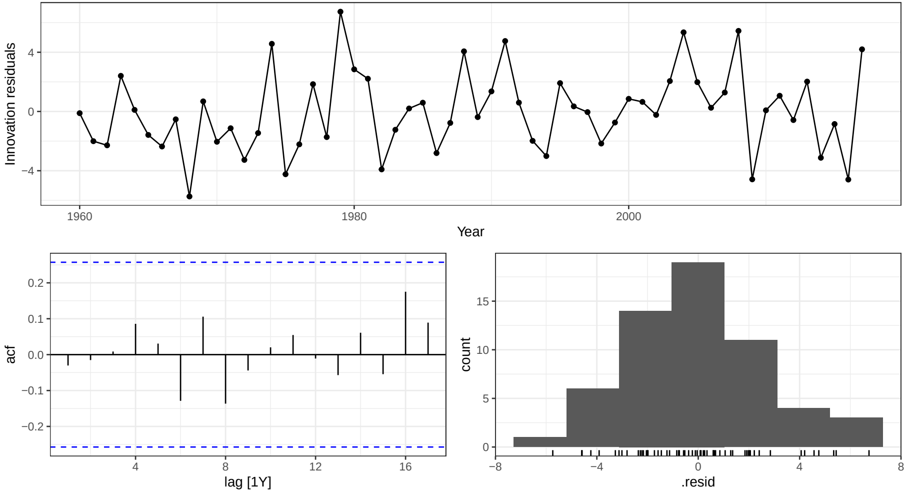
```r
egy |> model(ARIMA(Exports ~ pdq(p = c(1,2,4),
                                q = 0:3))) |>
  report()
```

```
## Series: Exports
## Model: ARIMA(2,0,1) w/ mean
##
## Coefficients:
##          ar1      ar2      ma1  constant
##       1.6764  -0.8034  -0.6896    2.5623
## s.e.  0.1111   0.0928   0.1492    0.1161
##
## sigma^2 estimated as 8.046:  log likelihood=-141.6
## AIC=293.1   AICc=294.3   BIC=303.4
```

ARIMA in Motion

Step 6: Check residuals

```
mod.fit |> gg_tsresiduals()
```



ARIMA in Motion

Step 6: Check residuals

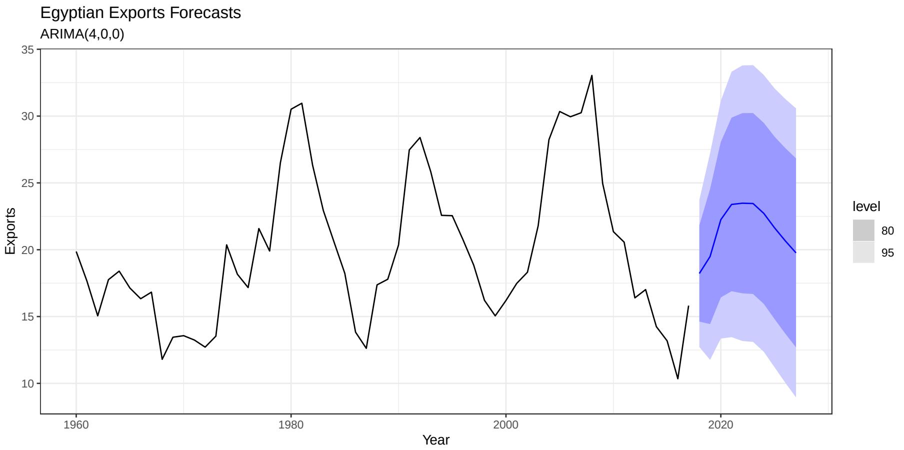
```
augment(mod.fit) |>
  features(.innov, ljung_box, lag = 10, dof = 4)
#where dof (K) = p+q, is the number of AR & MA parameters in model
```

```
## # A tibble: 1 x 4
##   Country          .model                             lb_stat lb_pvalue
##   <fct>            <chr>                                <dbl>     <dbl>
## 1 Egypt, Arab Rep. ARIMA(Exports ~ pdq(4, 0, 0))         3.95     0.684
```

ARIMA in Motion

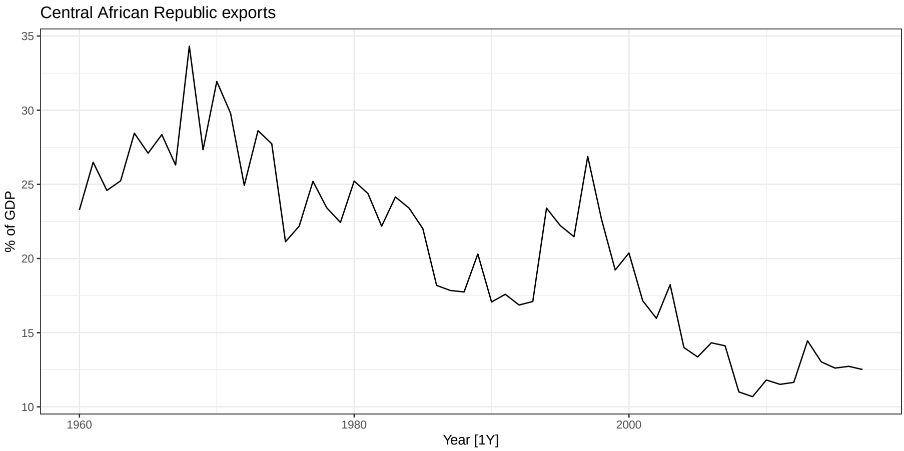
Step 7: Forecast using the preferred Model

```
mod.fit |> forecast(h = 10) |>
  autoplot(egy) +
  labs(title = "Egyptian Exports Forecasts",
       subtitle = "ARIMA(4,0,0)")
```



Example #2:

```
caf <- global_economy |> filter(Code == "CAF")

caf |> autoplot(Exports) +
  labs(title="Central African Republic exports", y="% of GDP")
```



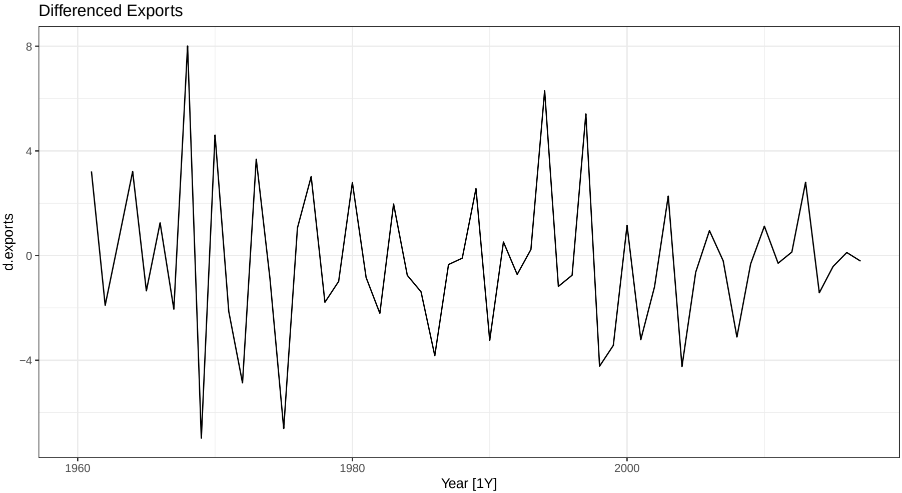
```
caf |> features(Exports, unitroot_kpss)
```

```
## # A tibble: 1 x 3
##   Country                  kpss_stat kpss_pvalue
##   <fct>                        <dbl>       <dbl>
## 1 Central African Republic      1.28        0.01
```

```
caf |> features(Exports, unitroot_ndiffs)
```

```
## # A tibble: 1 x 2
##   Country                  ndiffs
##   <fct>                     <int>
## 1 Central African Republic      1
```

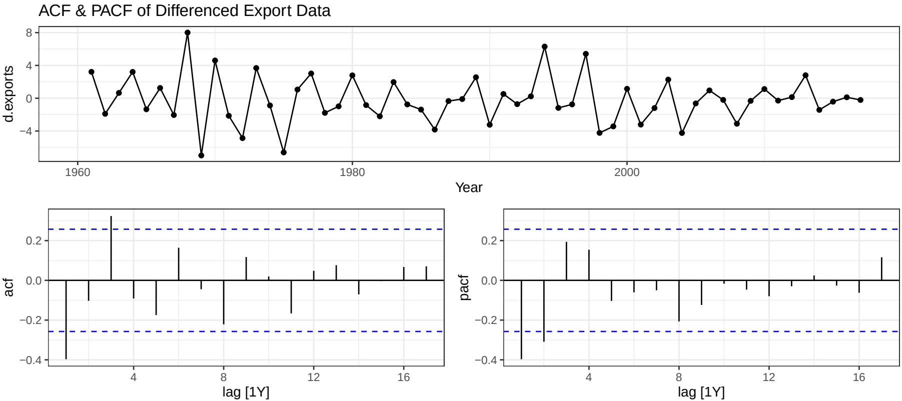
```
caf <- caf |> mutate(d.exports = difference(Exports))

caf |> autoplot(d.exports) + ggtitle("Differenced Exports")
```



```
caf |> features(d.exports, unitroot_kpss)
```

```
## # A tibble: 1 x 3
##   Country                  kpss_stat kpss_pvalue
##   <fct>                        <dbl>       <dbl>
## 1 Central African Republic    0.0922         0.1
```

```
caf |> gg_tsdisplay(d.exports,plot_type = "partial") +
  labs(title = "ACF & PACF of Differenced Export Data")
```



Candidate Models:

- ARIMA(1,1,0)
- ARIMA(2,1,0)
- ARIMA(0,1,1)
- ARIMA(0,1,3)
- ARIMA(1,1,1)
- ARIMA(1,1,3)
- ARIMA(2,1,1)
- ARIMA(2,1,3)

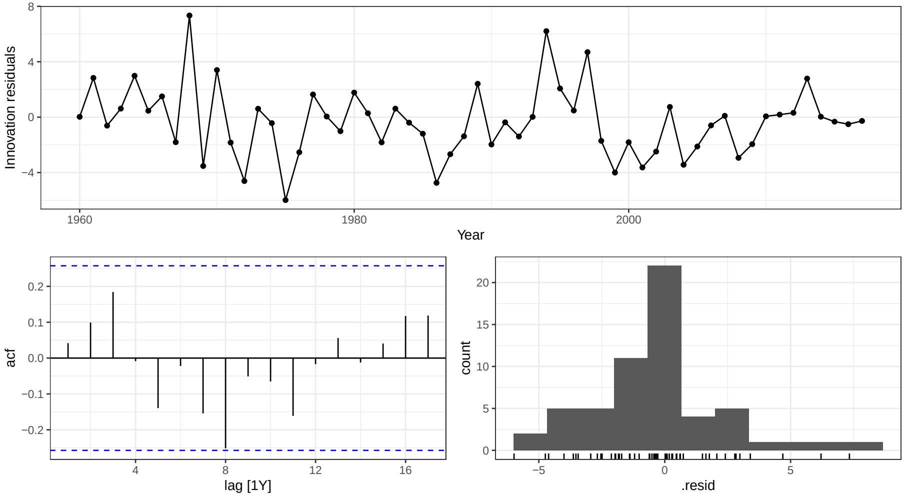
```r
caf.fits <- caf |> model(
  arima110 = ARIMA(Exports ~ pdq(1,1,0)),
  arima210 = ARIMA(Exports ~ pdq(2,1,0)),
  arima011 = ARIMA(Exports ~ pdq(0,1,1)),
  arima013 = ARIMA(Exports ~ pdq(0,1,3)),
  arima111 = ARIMA(Exports ~ pdq(1,1,1)),
  arima113 = ARIMA(Exports ~ pdq(1,1,3)),
  arima211 = ARIMA(Exports ~ pdq(2,1,1)),
  arima213 = ARIMA(Exports ~ pdq(2,1,3))
)
```

Select “Best” Model

```
caf.fits |> glance() |> select(.model:BIC) |>
  arrange(AICc)
```

```
## # A tibble: 7 x 6
##   .model   sigma2 log_lik   AIC  AICc   BIC
##   <chr>     <dbl>   <dbl> <dbl> <dbl> <dbl>
## 1 arima210   6.71   -134.  275.  275.  281.
## 2 arima013   6.54   -133.  274.  275.  282.
## 3 arima011   7.01   -136.  276.  276.  280.
## 4 arima211   6.69   -134.  275.  276.  284.
## 5 arima113   6.60   -133.  276.  277.  286.
## 6 arima110   7.22   -137.  278.  278.  282.
## 7 arima111   7.07   -136.  277.  278.  284.
```

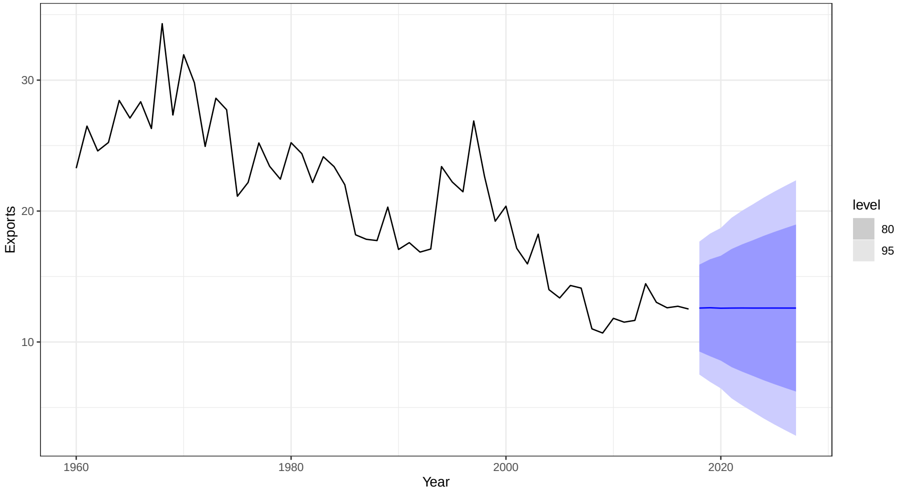
```r
caf.fits |>
  select(arima210) |>
  gg_tsresiduals()
```



```r
caf.fits |> augment() |>
  filter(.model == "arima210") |>
  features(.innov, ljung_box, lag = 10, df = 2)
```

```
## # A tibble: 1 x 4
##   Country                  .model   lb_stat lb_pvalue
##   <fct>                    <chr>      <dbl>     <dbl>
## 1 Central African Republic arima210    10.7     0.382
```

```
caf.fits |>
  forecast(h = 10) |>
  filter(.model == "arima210") |>
  autoplot(caf)
```



Section 6

Seasonal ARIMA Models

Seasonal ARIMA Models

- Seasonal pattern in a year
  - Seasonal pattern year after year
  - Seasonal differences for nonstationary data
- The notation for a seasonal ARIMA model:

$$\text{ARIMA} \quad \underbrace{(p, d, q)}_{\text{Nonseasonal part}} \quad \underbrace{(P, D, Q)_m}_{\text{Seasonal part}}$$

where

- p is regular AR terms
- d is regular differences
- q is regular MA terms
- m is the seasonal period (e.g., number of observations per year)
- P is seasonal autoregressive terms at lag m
- D is seasonal difference at lag m
- Q is seasonal MA terms at lag m.

ACF/PACF Pattern for Seasonal lags

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

For example, an $\text{ARIMA}(0,0,0)(0,0,1)_{12}$ model will show:

- a spike at lag 12 in the ACF but no other significant spikes;
- exponential decay in the seasonal lags of the PACF (i.e., at lags 12, 24, 36, . . .).

ACF/PACF Pattern for Seasonal lags

Similarly, an $\text{ARIMA}(0,0,0)(1,0,0)_{12}$ model will show:

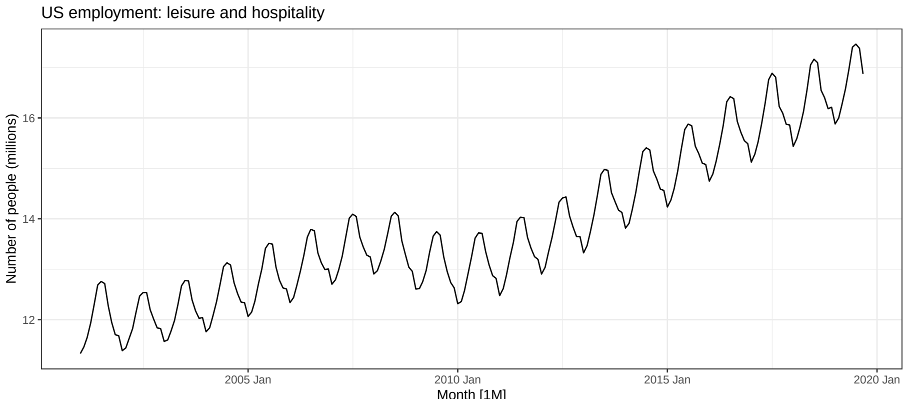
- exponential decay in the seasonal lags of the ACF;
- a single significant spike at lag 12 in the PACF.

When considering the appropriate seasonal orders for a seasonal ARIMA model, we restrict attention to the **seasonal lags**.

The modelling procedure is almost the same as for non-seasonal data, except that we need to select seasonal AR and MA terms as well as the non-seasonal components of the model.

Seasonal ARIMA model in Motion

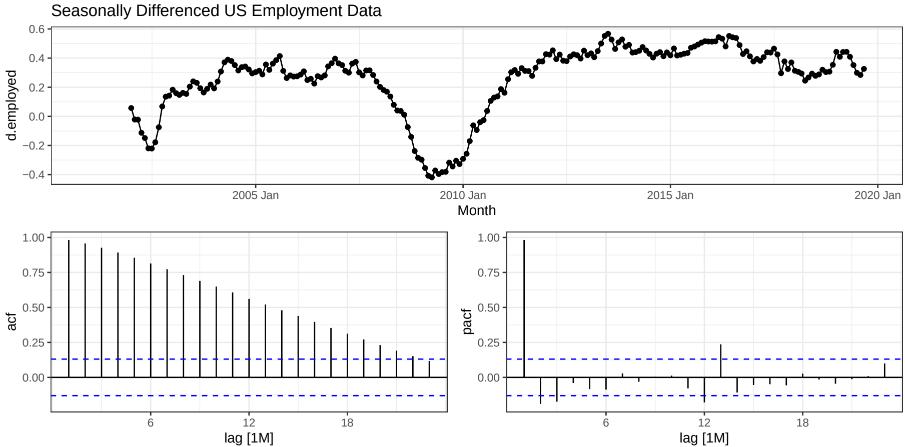
```r
leisure <- us_employment |>
  filter(Title == "Leisure and Hospitality", year(Month) > 2000) |>
  mutate(Employed = Employed/1000) |>
  select(Month, Employed)

autoplot(leisure, Employed) +
  labs(title = "US employment: leisure and hospitality",
       y="Number of people (millions)")
```



The data are non-stationary with trend and seasonality.

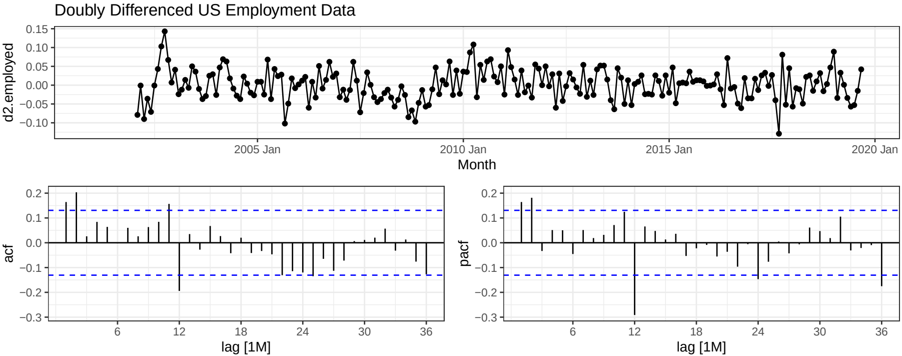
Let us first seasonally difference the data

```
leisure |> mutate(d.employed = difference(Employed, 12)) |>
  gg_tsdisplay(d.employed, plot_type = "partial") +
  labs(title = "Seasonally Differenced US Employment Data")
```



The series still appears to be nonstationary with a non-linear trend. Take further first difference

```
leisure |> mutate(d2.employed = difference(Employed, 12) |>
                    difference()) |>
  gg_tsdisplay(d2.employed, plot_type = "partial", lag_max = 36) +
  labs(title = "Doubly Differenced US Employment Data")
```



Candidate Models

1. $\text{ARIMA}(0,1,2)(1,1,0)_{12}$
2. $\text{ARIMA}(0,1,2)(0,1,1)_{12}$
3. $\text{ARIMA}(0,1,2)(1,1,1)_{12}$
4. $\text{ARIMA}(2,1,0)(1,1,0)_{12}$
5. $\text{ARIMA}(2,1,0)(0,1,1)_{12}$
6. $\text{ARIMA}(2,1,0)(1,1,1)_{12}$
7. $\text{ARIMA}(2,1,2)(1,1,0)_{12}$
8. $\text{ARIMA}(2,1,2)(0,1,1)_{12}$
9. $\text{ARIMA}(2,1,2)(1,1,1)_{12}$

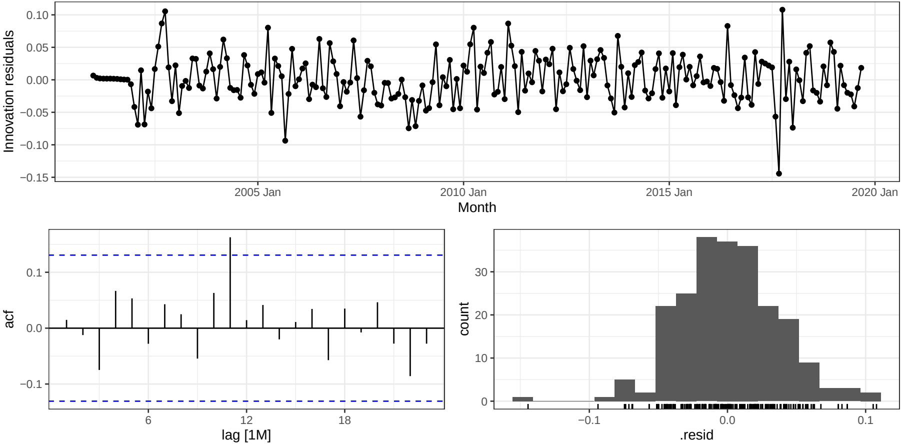
```r
s.arima.fit <- leisure |> model(
  arima012110 = ARIMA(Employed ~ pdq(0,1,2) + PDQ(1,1,0)),
  arima012011 = ARIMA(Employed ~ pdq(0,1,2) + PDQ(0,1,1)),
  arima012111 = ARIMA(Employed ~ pdq(0,1,2) + PDQ(1,1,1)),
  arima210110 = ARIMA(Employed ~ pdq(2,1,0) + PDQ(1,1,0)),
  arima210011 = ARIMA(Employed ~ pdq(2,1,0) + PDQ(0,1,1)),
  arima210111 = ARIMA(Employed ~ pdq(2,1,0) + PDQ(1,1,1)),
  arima212110 = ARIMA(Employed ~ pdq(2,1,2) + PDQ(1,1,0)),
  arima212011 = ARIMA(Employed ~ pdq(2,1,2) + PDQ(0,1,1)),
  arima212111 = ARIMA(Employed ~ pdq(2,1,2) + PDQ(1,1,1))
)
```

```
s.arima.fit |> glance() |>
  arrange(AICc) |> select(.model:BIC)
```

```
## # A tibble: 9 x 6
##   .model       sigma2 log_lik   AIC  AICc   BIC
##   <chr>         <dbl>   <dbl> <dbl> <dbl> <dbl>
## 1 arima210111 0.00142    395. -780. -780. -763.
## 2 arima012111 0.00142    395. -779. -779. -763.
## 3 arima210011 0.00145    392. -776. -776. -763.
## 4 arima212111 0.00143    395. -776. -775. -752.
## 5 arima012011 0.00146    391. -775. -775. -761.
## 6 arima212011 0.00146    393. -773. -773. -753.
## 7 arima210110 0.00154    387. -766. -765. -752.
## 8 arima012110 0.00156    386. -764. -764. -751.
## 9 arima212110 0.00156    387. -762. -762. -742.
```

Check Model Residuals

```
s.arima.fit |> select(arima210111) |> gg_tsresiduals()
```



Lag 11 might be concerning. Conduct a Ljung-Box test.

Check Model Residuals

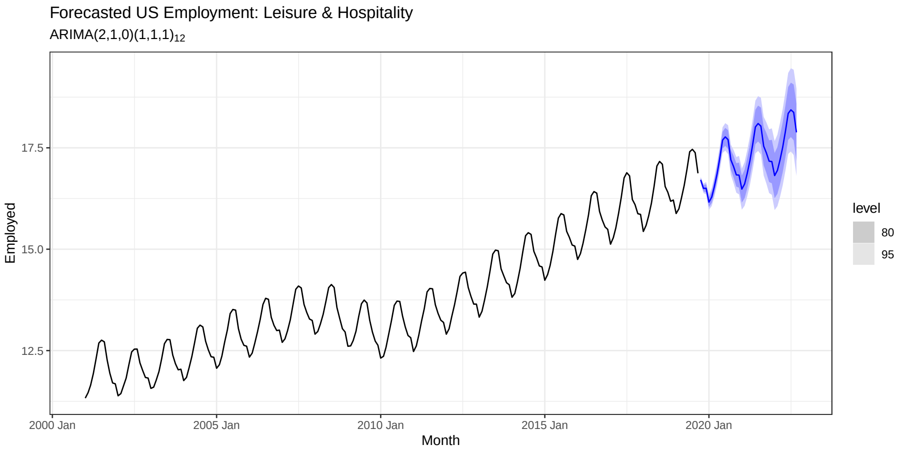
```
s.arima.fit |> select(arima210111) |>
  augment() |> features(.innov, ljung_box, lag = 24,
                           dof = 4)
```

```
## # A tibble: 1 x 3
##   .model      lb_stat lb_pvalue
##   <chr>         <dbl>     <dbl>
## 1 arima210111    16.6     0.680
```

False Alarm: We actually fail to reject null of no autocorrelation. Our residuals are indeed white noise.

Forecasting with Preferred Model

```r
s.arima.fit |> forecast(h = "3years") |>
  filter(.model == "arima210111") |>
  autoplot(leisure) +
  labs(title = "Forecasted US Employment: Leisure & Hospitality",
       subtitle = bquote("ARIMA(2,1,0)(1,1,1)"[12]))
```



ARIMA Models with Exogenous Variables

- ARIMA models can be extended to include exogenous variables.
- The model is then called ARIMAX.
- These are also referred to as Regression with ARIMA errors.
  - The `ARIMA()` function, from earlier, will fit a regression model with `ARIMA` errors if exogenous regressors are included in the formula.
  - If differencing is specified, then the **differencing is applied to all variables in the regression model before the model is estimated**.

ARIMA Models with Exogenous Variables

Here is an example:

```
ARIMA(y ~ x + pdq(1,1,0))
```

This will fit the model of the form:

$$y_t = \beta_0 + \beta_1 x_t + \eta_t$$

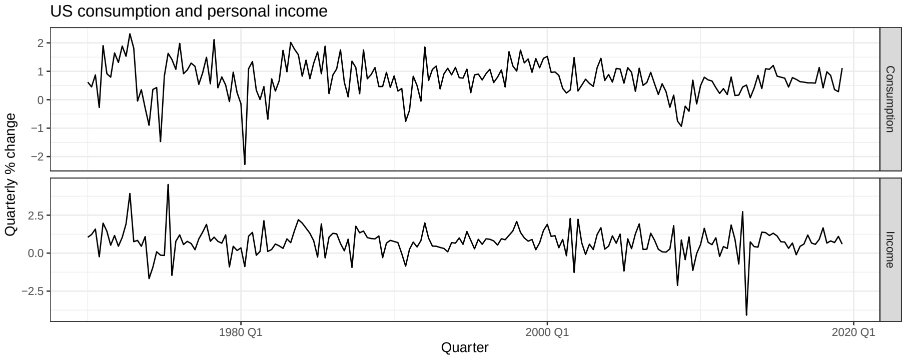
where η_t is the ARIMA(1,1,0) error term and $\Delta\eta_t = \phi_1 \Delta\eta_{t-1} + \varepsilon_t$.

ARIMA Models with Exogenous Variables

- Optimal lag orders of the errors can be determined by the `ARIMA` function.
    - `R` will select these automatically if the `pdq()` arguments are not specified.
    - Whether differencing is required is determined by applying a `KPSS` test to the residuals from the OLS regression model.
    - If differencing is required, then all variables are differenced and the model re-estimated using maximum likelihood estimation.
    - Final model will be expressed in terms of the **original variables**, even if it has been estimated using differenced variables.

ARIMA Models with Exogenous Variables

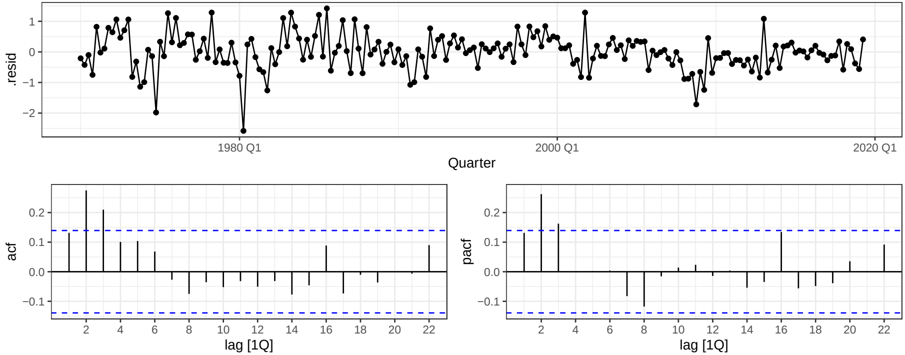
```
us_change |> pivot_longer(c(Consumption, Income),
                          names_to = "var", values_to = "value") |>
  ggplot(aes(x = Quarter, y = value)) +  geom_line() +
  facet_grid(vars(var), scales = "free_y") +
  labs(title = "US consumption and personal income",
       y = "Quarterly % change")
```



```r
fit <- us_change |> model(TSLM(Consumption ~ Income))
report(fit)
```

```
## Series: Consumption
## Model: TSLM
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.5824 -0.2778  0.0186  0.3233  1.4223
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.5445     0.0540   10.08  < 2e-16 ***
## Income        0.2718     0.0467    5.82  2.4e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.591 on 196 degrees of freedom
## Multiple R-squared: 0.147,   Adjusted R-squared: 0.143
## F-statistic: 33.8 on 1 and 196 DF, p-value: 2.4e-08
```

```
fit.resid <- fit |> augment() %>% select(.resid)

fit.resid %>% gg_tsdisplay(plot_type = "partial")
```



```r
fit.resid %>% model(
  ARIMA(.resid ~ pdq(
    p = c(1:3), d = 0, q = c(1:3)))
) %>% report()
```

```
## Series: .resid
## Model: ARIMA(1,0,2)
##
## Coefficients:
##          ar1      ma1     ma2
##       0.6920  -0.6456  0.2187
## s.e.  0.1188   0.1315  0.0703
##
## sigma^2 estimated as 0.3121:  log likelihood=-164.3
## AIC=336.6   AICc=336.8   BIC=349.7
```

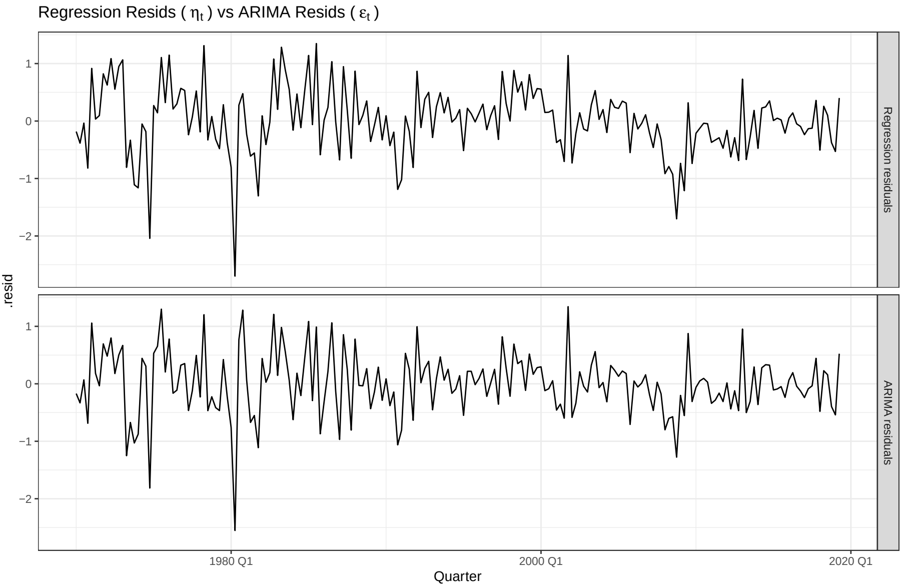
$$\begin{aligned}
y_t &= 0.595 + 0.198x_t + \eta_t, \\
\eta_t &= 0.707\eta_{t-1} + \varepsilon_t - 0.617\varepsilon_{t-1} + 0.207\varepsilon_{t-2}, \\
\varepsilon_t &\sim \text{NID}(0, 0.311).
\end{aligned}$$

```
fit2 <- us_change |>
  model(ARIMA(Consumption ~ Income))
report(fit2)
```

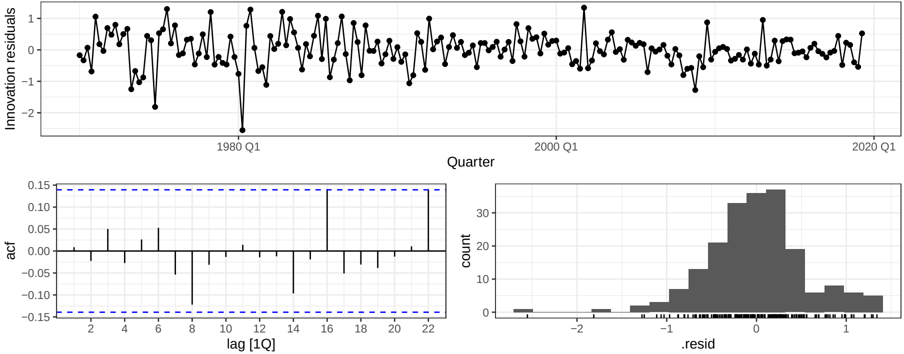
```
## Series: Consumption
## Model: LM w/ ARIMA(1,0,2) errors
##
## Coefficients:
##          ar1      ma1     ma2  Income  intercept
##       0.7070  -0.6172  0.2066  0.1976     0.5949
## s.e.  0.1068   0.1218  0.0741  0.0462     0.0850
##
## sigma^2 estimated as 0.3113:  log likelihood=-163
## AIC=338.1   AICc=338.5   BIC=357.8
```

Regression vs ARIMA Residuals

```
bind_rows(
    `Regression residuals` =
        as_tibble(residuals(fit2, type = "regression")),
    `ARIMA residuals` =
        as_tibble(residuals(fit2, type = "innovation")),
    .id = "type"
  ) %>%
  mutate(
    type = factor(type, levels=c(
      "Regression residuals", "ARIMA residuals"))
  ) %>%
  ggplot(aes(x = Quarter, y = .resid)) + geom_line() +
  facet_grid(vars(type)) +
  labs(title =
  bquote("Regression ("~eta[t]~") vs ARIMA ("~epsilon[t]~") Resids"))
```


Regression Resids (η_t) vs ARIMA Resids (ε_t)
.resid
1
0
−1
−2
Regression residuals
1
0
−1
−2
ARIMA residuals
1980 Q1
2000 Q1
2020 Q1
Quarter

```
fit2 |> gg_tsresiduals()
```



```
augment(fit) |>
  features(.innov, ljung_box, dof = 3, lag = 8)
```

```
## # A tibble: 1 x 3
##   .model                          lb_stat  lb_pvalue
##   <chr>                             <dbl>      <dbl>
## 1 TSLM(Consumption ~ Income)         34.2 0.00000213
```