# Time Series Analysis

UC Berkeley, School of Information: MIDS w271

2023-04-06

# Contents

# Problem Set 11

This is the final problem set that you will have to work on for this class. Congratulations! (Although there is still a group lab that will be the final assignment in the course.)

You will start with some guided work, and then proceed into less structured work that will let you stretch and demonstrate what you have learned to date.

Notice, in particular, that the last few questions are asking you essentially to "produce a model" using a method. At this point in the course, you should be familiar with many of the model forms that you *might* fit; and, you are familiar with methods that you can use to evaluate models' performances. In these questions, we are asking you to, essentially, fit a good model with a method and then to evaluate how a good model with "this" method is doing compared to another good model with "that" method.

In several of these questions, there isn't a correct answer, *per se*. Instead, there is the process that you will undertake and record as you are producing your argument for the model that you think is best meeting your objectives. This is a **very** applied task that we anticipate you will see many times in your work.

```
knitr::opts_chunk$set(echo=TRUE, dpi=1000, tidy=T, tidy.opts=list(width.cutoff=80))
```

We are providing you with an additional challenge, but one that is also very evocative of work that you're likely to come across. This is a well-built repository, that uses a well-documented framework to produce reports, namely `bookdown`.

Once you have done your work, you can render the entire book using the following call in your console:

```
> bookdown::render_book()
```

This will ingest each of the files `01-time_series_...`, `02-cross_validation.Rmd`, `03-ARIMA_model.Rmd` and so on ... and produce a PDF that is stored in `./_book/_main.pdf`. If you would like to read more about this framework, you can do so at the following website: https://bookdown.org/yihui/bookdown/ .

# 1 (10 points) ARIMA model

Consider `fma::sheep`, the sheep population of England and Wales from 1867–1939. :sheep:

```
# install.packages('fma')
library(fma)
library(fpp3)
head(fma::sheep)
```

```
## Time Series:
## Start = 1867
## End = 1872
## Frequency = 1
## [1] 2203 2360 2254 2165 2024 2078
```

## 1.1 Time series plot

Produce a time plot of the time series, and comment on what you observe.

```
plot(fma::sheep)
```


From the time series plot we can observe some piecewise trends, where the population from 1870 to the early 1920s was clearly declining as a whole, with a a lot of variation in the individual years, and it started to climb back up again from ~1922.

## 1.2 Fit a model

Assume you decide to fit the following model:

$$y_t = y_{t-1} + \phi_1(y_{t-1} - y_{t-2}) + \phi_2(y_{t-2} - y_{t-3}) + \phi_3(y_{t-3} - y_{t-4}) + \epsilon_t$$

where $\epsilon_t$ is a white noise series.

### 1.2.1 Model type

What sort of ARIMA model is this (i.e., what are $p$, $d$, and $q$)?

This is an ARIMA(3,1,0) model.

### 1.2.2 Back to the future

Express this ARIMA model using backshift operator notation.

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

Where $(1-B)$ represents the first differencing, and the multiplied polynomial is the AR(3) portion, with no MA portion.

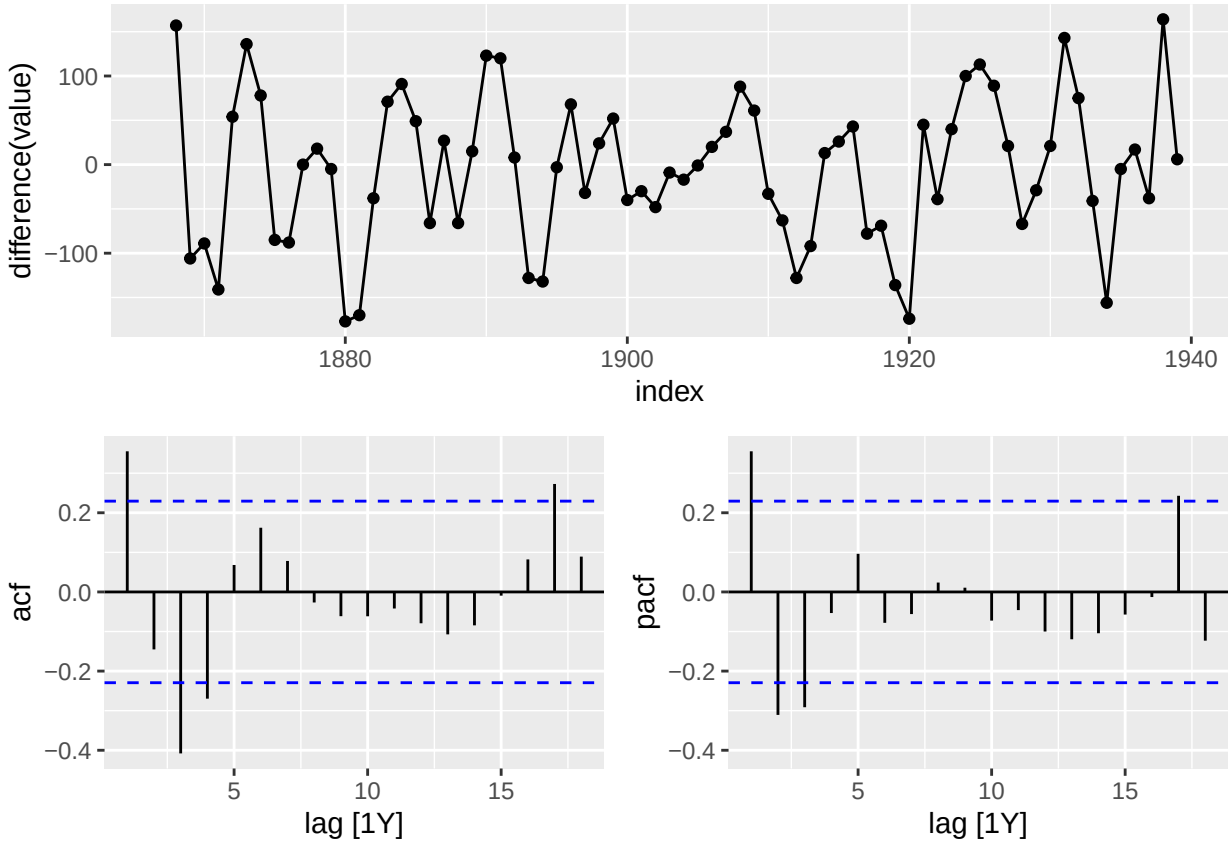
## 1.3 Is this model appropriate?

Examine the ACF and PACF of the differenced data. Evaluate whether this model is appropriate.

```
sheep <- as_tsibble(fma::sheep)
sheep %>%
    gg_tsdisplay(difference(value), plot_type = "partial")
```

```
## Warning: Removed 1 row containing missing values (`geom_line()`).
```

```
## Warning: Removed 1 rows containing missing values (`geom_point()`).
```




From the ACF plot, we can see a dampened sine wave pattern (with an uncharacteristic spike at lag 17 or 18). We further notice that the PACF plot has significant spikes at lags 1, 2, and 3, but not for any of the subsequent lags (except for one uncharacteristic spike at lag 19). These patterns are characteristic of AR(3) models, which provides evidence that this is a good model for the data. We do , however, underscore the uncharacteristic aforementioned spikes which might be an indication that there's an underlying signal that is not being adequately captured, so a more robust model might be chosen.

## 1.4 Forecasts, by hand!

The last five values of the series are given below:

| Year | 1935 | 1936 | 1937 | 1938 | 1939 |
|---|---|---|---|---|---|
| Millions of sheep | 1648 | 1665 | 1627 | 1791 | 1797 |

The estimated parameters are:

- $\phi_1 = 0.42$;
- $\phi_2 = -0.20$; and,
- $\phi_3 = -0.30$.

Without using the forecast function, calculate forecasts for the next three years (1940–1942).

```
## you can use R as a calculator here
phi_1 = 0.42
phi_2 = -0.2
phi_3 = -0.3
y.val <- function(h, base.data) {
    for (i in 1:h) {
```

```
        df <- slice_tail(base.data, n = 4)
        df <- as.ts(df)
        result <- nth(df, -1) + sum(phi_1 * c(nth(df, -1), -nth(df, -2))) + sum(phi_2 *
            c(nth(df, -2), -nth(df, -3))) + sum(phi_3 * c(nth(df, -3), -nth(df, -4)))
        base.data <- bind_rows(base.data, data.frame(index = (slice_tail(base.data,
            n = 1)$index) + 1, value = result))
    }
    return(slice_tail(base.data, n = h))
}
```

```
y.val(3, sheep)
```

```
## # A tsibble: 3 x 2 [1Y]
##   index value
##   <dbl> <dbl>
## 1  1940 1778.
## 2  1941 1720.
## 3  1942 1697.
```

## 1.5 Interpret roots

Find the roots of your model's characteristic equation. Is this process stationary?.

```
abs(polyroot(c(1, -phi_1, phi_2, phi_3)))
```

```
## [1] 1.045190 1.785837 1.785837
```

The absolute value of the roots of the 3rd order polynomial are above, and the other root is $B = 1$. Because this is a unit root, this process is *not stationary*.

# 2 (10 points) Seasonal ARIMA model

Download the series of E-Commerce Retail Sales as a Percent of Total Sales here.

(Feel free to explore the `fredr` package and API if interested.)

Our goal is to Build a Seasonal ARIMA model, following all appropriate steps for a univariate time series model.

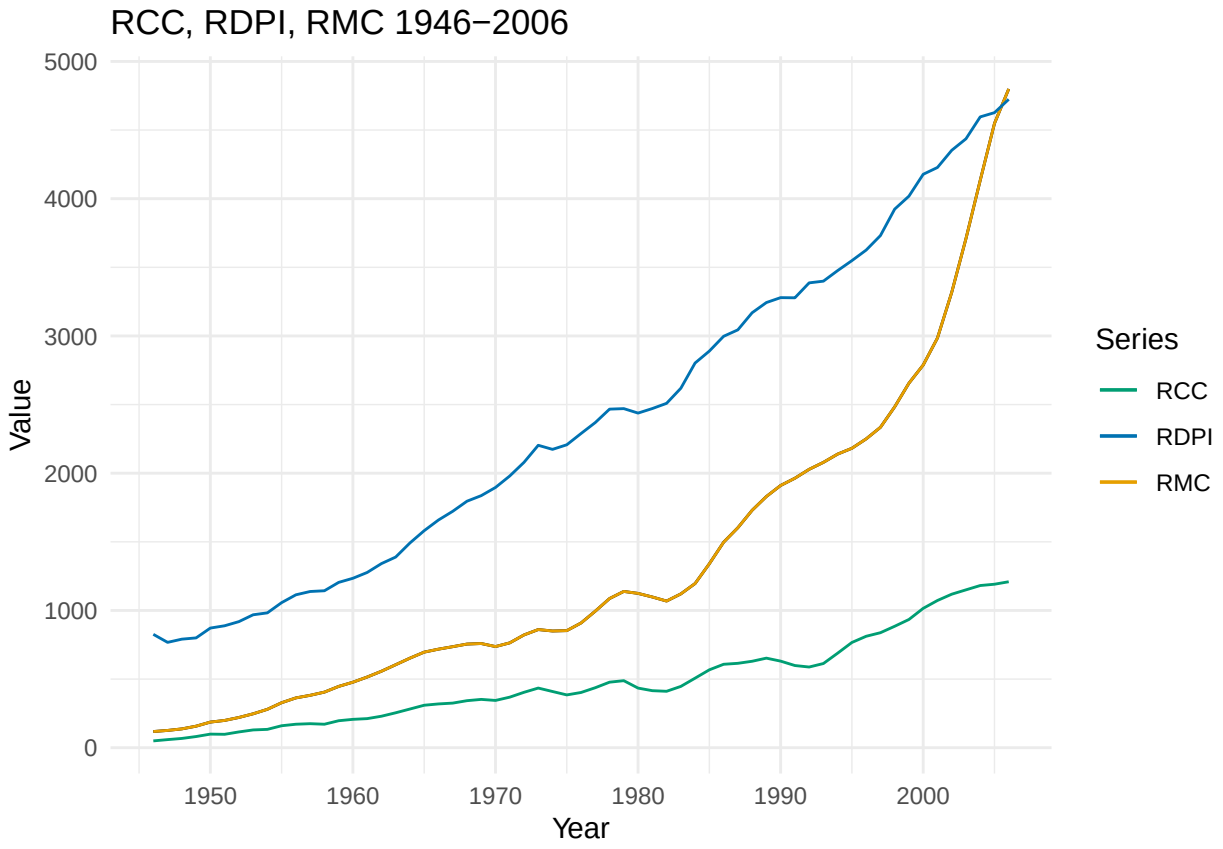
Separate the data set into training and test data. The training data is used to estimate model parameters, and it is for 10/1999-12/2020. The test data is used to evaluate its accuracy, and it is for 01/2021-01/22.

```
ects.orig <- ects
ects.test <- ects |>
    filter_index("2021 Q1" ~ "2022 Q1")
ects <- ects |>
    filter_index(~"2020 Q4")
```

## 2.1 Time series plot

Plot training data set of Retail Sales. What do you notice? Is there any transformation necessary?

```
# Fill this in
ects %>%
    autoplot(ecompctnsa) + labs(title = "E-commerce retail sales as a percent of total sales",
    subtitle = "Quarterly. 1999 Q4 - 2020 Q4", y = "Percent")
```



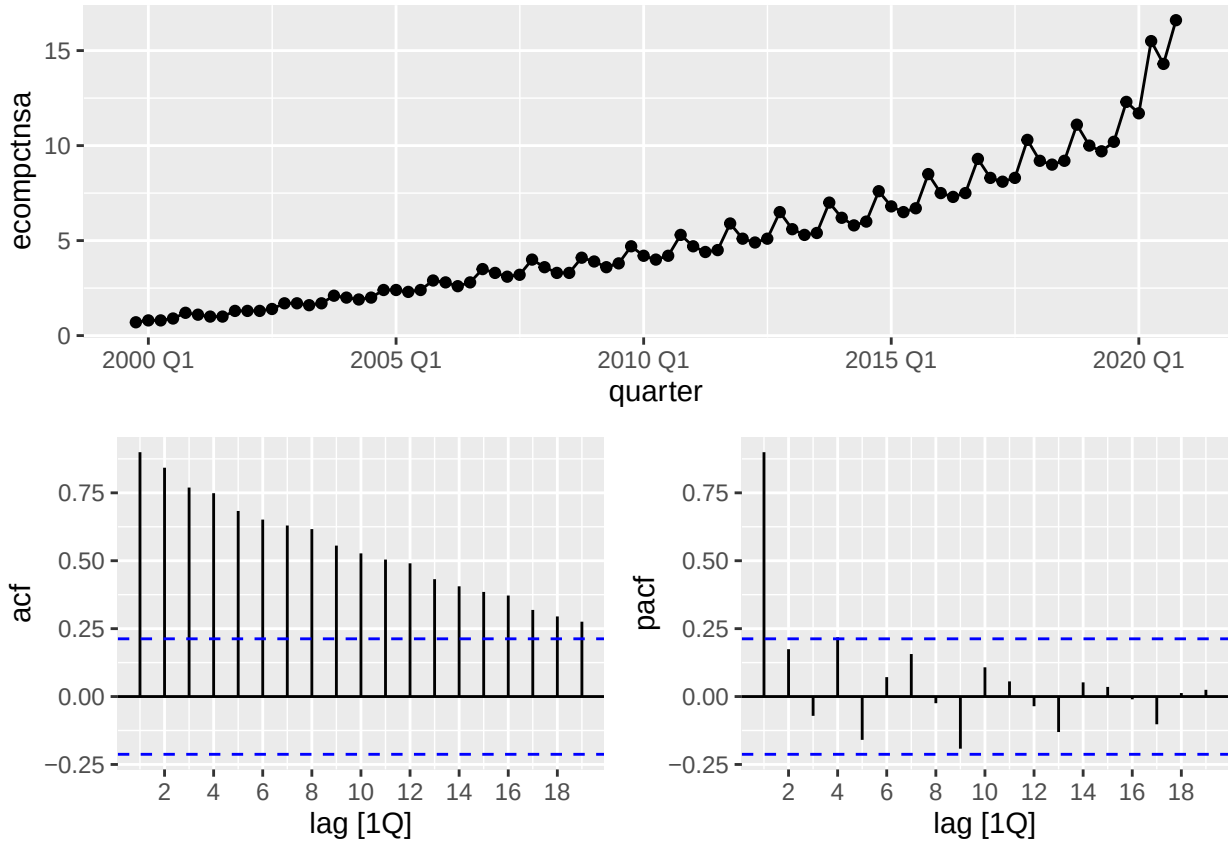

From the plot, we can observe a semi-linear trend with definite seasonality up until the second quarter of 2020. There is a big jump during that quarter which is attributable presumably to the COVID-19 lockdown, which precipitated

e-commerce sales in the face of the inability to attend physical stores and physical distancing measures. It is evident from the plot that the variance needs to be stabilized, so a log transform might be appropriate.

## 2.2 Check for Stationary

Use ACF/PACF and a unit root test to check if Retail Sales is stationary. If data is not stationary, difference the data, and apply the test again until it becomes stationary. How many differences are needed to make data stationary?

```
gg_tsdisplay(ects, ecompctnsa, plot_type = "partial")
```




From the time plot, it is evident that the data is not stationary in all possible ways it can be non-stationary: (1) there is a clear upwards trend, (2) the variance is heteroskedastic (non-stable), and (3) there is evident seasonality. The ACF and PACF give further evidence of this. The ACF plot shows a slowly decaying autocorrelation, consistent with non-stationary data. In the PACF, we can see lag 1 has a significant positive autocorrelation, which solidifies our conclusion.

```
tseries::adf.test(log(ects$ecompctnsa), k = 4)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  log(ects$ecompctnsa)
## Dickey-Fuller = -1.3331, Lag order = 4, p-value = 0.8496
## alternative hypothesis: stationary
```

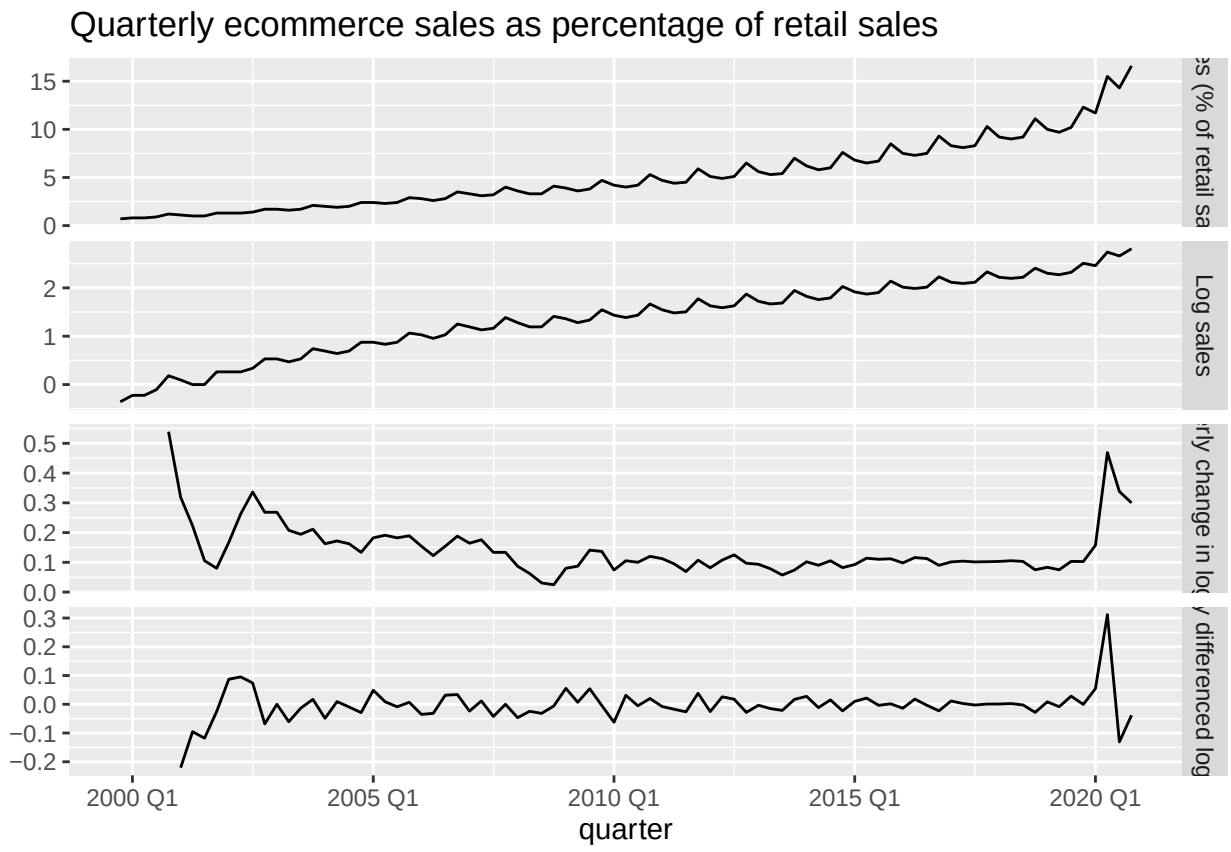
```
ects %>%
    features(log(ecompctnsa), list(unitroot_kpss, unitroot_nsdiffs, unitroot_ndiffs))
```

```
## # A tibble: 1 x 4
```

```
##   kpss_stat kpss_pvalue nsdiffs ndiffs
##       <dbl>       <dbl>   <int>  <int>
## 1      2.16        0.01       1      1
```

Running the ADF and KPSS tests, both indicate strong evidence for non-stationarity, and suggest that the data will require 1 seasonal difference and a first-order difference, which we can confirm through their plots:

```
# TODO: does it make sense to log percentage data?
ects %>%
    select(-date) %>%
    transmute(`Sales (% of retail sales)` = ecompctnsa, `Log sales` = log(ecompctnsa),
        `Quarterly change in log sales` = difference(log(ecompctnsa), 4), `Doubly differenced log sales` =
            4), 1)) |>
    pivot_longer(-quarter, names_to = "Type", values_to = "Sales") |>
    mutate(Type = factor(Type, levels = c("Sales (% of retail sales)", "Log sales",
        "Quarterly change in log sales", "Doubly differenced log sales"))) |>
    ggplot(aes(x = quarter, y = Sales)) + geom_line() + facet_grid(vars(Type), scales = "free_y") +
    labs(title = "Quarterly ecommerce sales as percentage of retail sales", y = NULL)
```




We have applied a single seasonal difference and a first-order difference. We have also log-transformed the original series to stabilize the variance. We now run the tests again to check for stationarity:

```
tseries::adf.test(ects$sales.trans[!is.na(ects$sales.trans)])
```

```
## Warning in tseries::adf.test(ects$sales.trans[!is.na(ects$sales.trans)]): p-value smaller than printed
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  ects$sales.trans[!is.na(ects$sales.trans)]
## Dickey-Fuller = -6.5565, Lag order = 4, p-value = 0.01
```

```
## alternative hypothesis: stationary
```

```
ects %>%
    features(sales.trans, list(unitroot_kpss, unitroot_nsdiffs, unitroot_ndiffs))
```

```
## # A tibble: 1 x 4
##   kpss_stat kpss_pvalue nsdiffs ndiffs
##       <dbl>       <dbl>   <int>  <int>
## 1     0.305         0.1       0      0
```

Both the ADF test and the KPSS test give strong statistical evidence that the transformed series is now stationary and we can proceed with modelling.

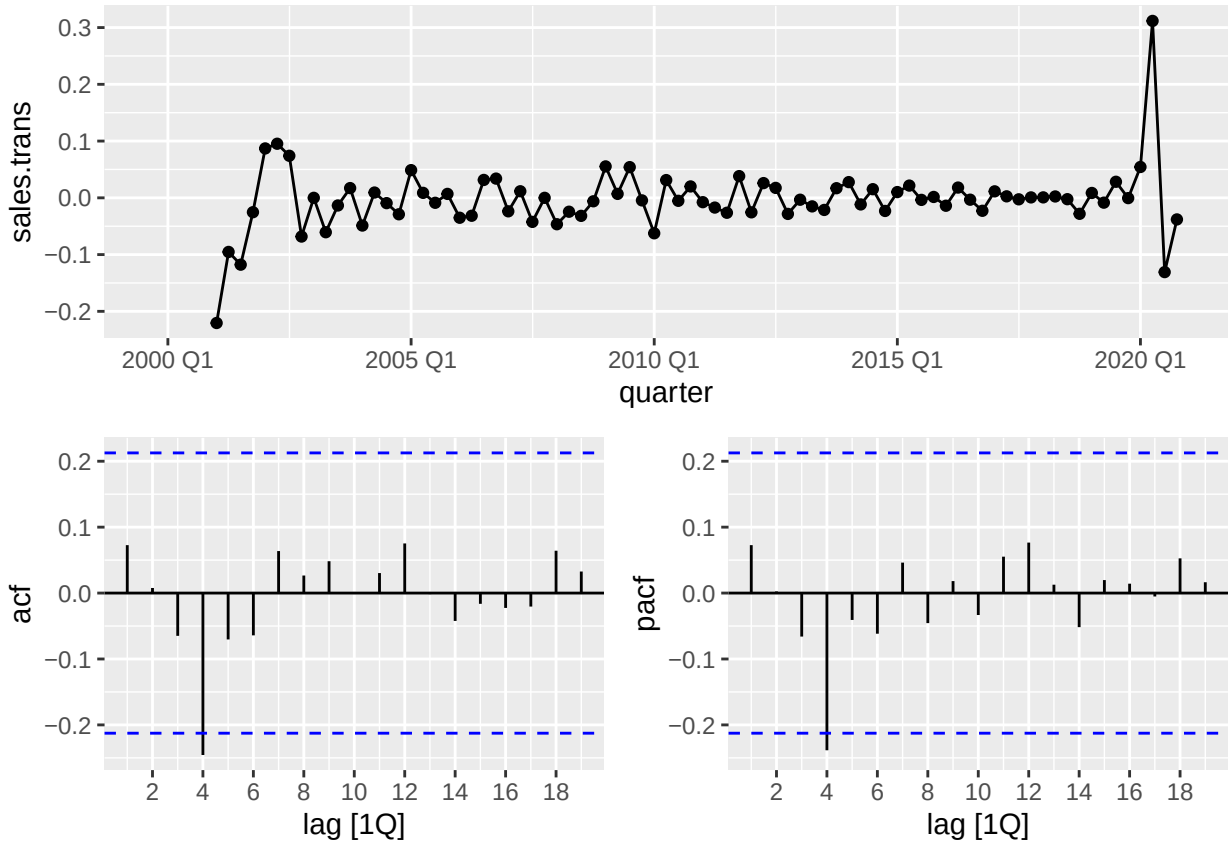
## 2.3 Model identification and estimation

Use ACF/PACF to identify an appropriate SARIMA model. Estimate both select model and model chosen by ARIMA()

```
# Fill this in
ects %>%
    gg_tsdisplay(sales.trans, plot_type = "partial")
```

```
## Warning: Removed 5 rows containing missing values (`geom_line()`).
```

```
## Warning: Removed 5 rows containing missing values (`geom_point()`).
```




From the ACF plot of the differenced - and transformed - series, we can observe a significant *negative* peak at lag 4, and no other significant peaks, which suggests a non-seasonal MA(4) component. At the same time, it suggests a seasonal MA(1) component. Hence, a good model to start with is ARIMA$(0,1,4)(0,1,1)_4$.

From the PACF plot, we have significant spikes at lags 4 and 8, which are multiples of the seasonality $m = 4$ (as this

is quarterly data). This might be indicative of a seasonal AR(1) component, but the PACF plot is not particularly well behaved, although lag 8 is just slightly over the significance threshold. This suggests that there might be a more robust model we can fit, which we will try to get to via the optimization search.

```
fit <- ects %>%
    model(arima014011 = ARIMA(log(ecompctnsa) ~ 0 + pdq(0, 1, 4) + PDQ(0, 1, 1)),
        arima014111 = ARIMA(log(ecompctnsa) ~ 0 + pdq(0, 1, 4) + PDQ(1, 1, 1)), auto = ARIMA(log(ecompctns
            stepwise = F, approx = F))
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 arima014011  <ARIMA(0,1,4)(0,1,1)[4]>
## 2 arima014111  <ARIMA(0,1,4)(1,1,1)[4]>
## 3 auto         <ARIMA(0,1,0)(0,1,2)[4]>
```

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

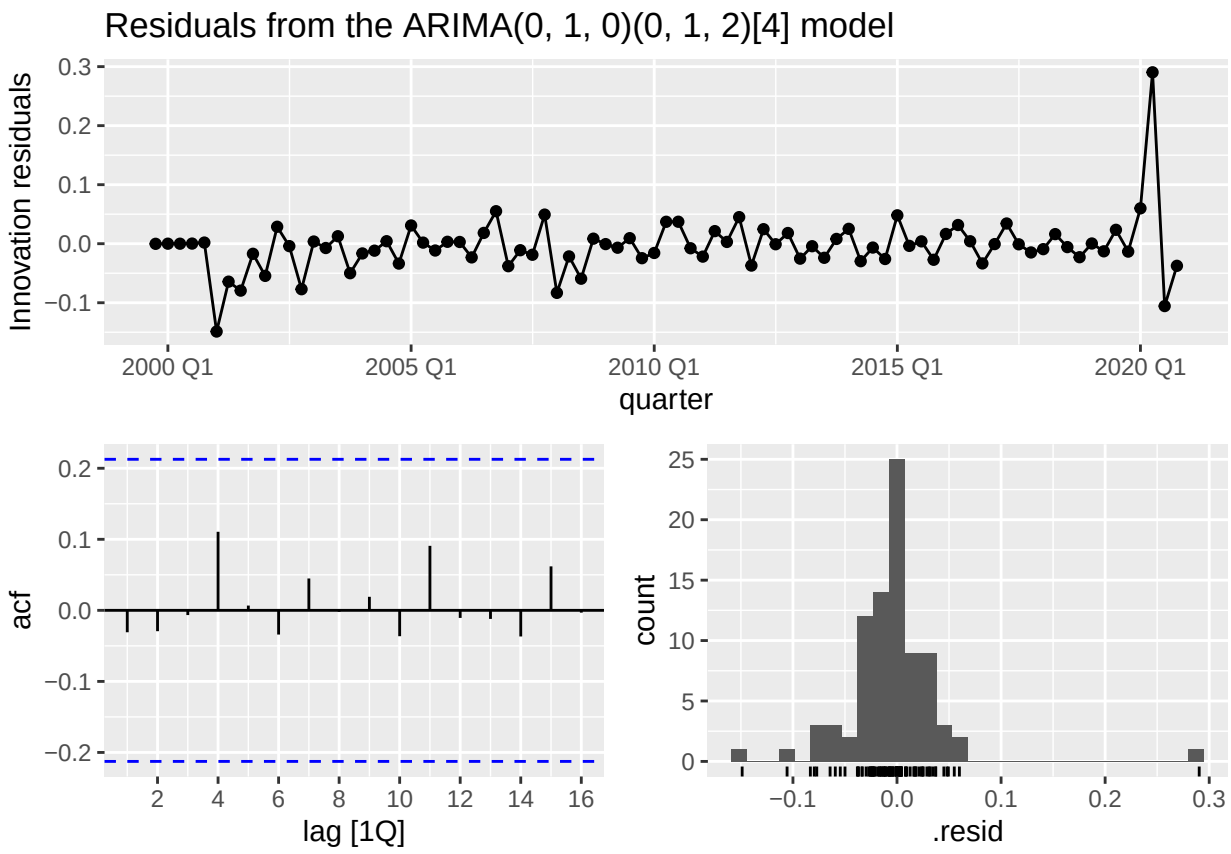
```
## # A tibble: 3 x 6
##   .model       sigma2 log_lik   AIC  AICc   BIC
##   <chr>         <dbl>   <dbl> <dbl> <dbl> <dbl>
## 1 auto        0.00242    126. -246. -246. -239.
## 2 arima014111 0.00267    125. -236. -234. -219.
## 3 arima014011 0.00279    123. -235. -234. -220.
```

Fitting the model results in an automatically selected ARIMA$(0,1,0)(0,1,2)_4$ model which means there is no non-seasonal autoregressive or MA component.

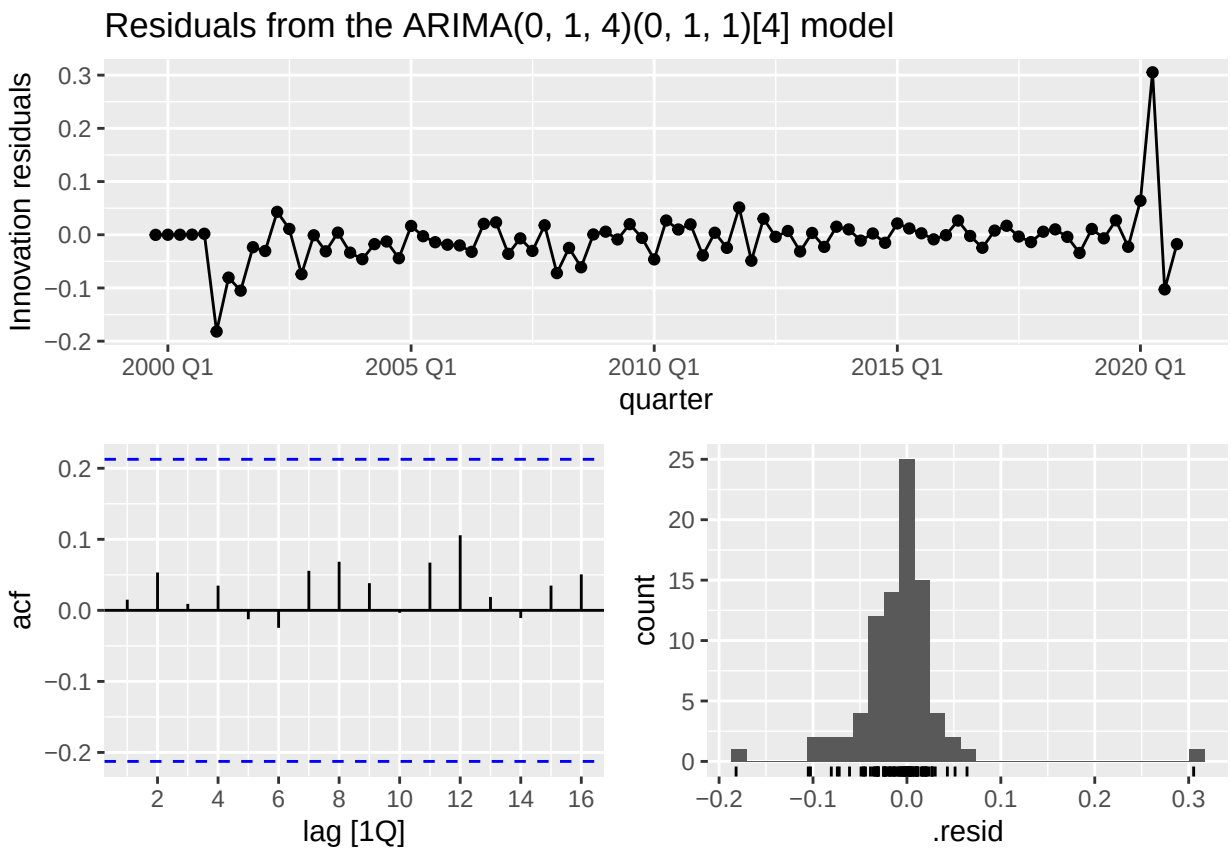
## 2.4 Model diagnostic

Do residual diagnostic checking of both models. Are the residuals white noise? Use the Ljung-box test to check if the residuals are white noise.

```
fit |>
    select(auto) |>
    gg_tsresiduals(lag = 16) + labs(title = paste("Residuals from the", expression(ARIMA(0,
    1, 0)(0, 1, 2)[4]), "model"))
```




```
fit |>
    select(arima014011) |>
    gg_tsresiduals(lag = 16) + labs(title = paste("Residuals from the", expression(ARIMA(0,
    1, 4)(0, 1, 1)[4]), "model"))
```




```
bind_rows(augment(select(fit, auto)) |>
    features(.innov, ljung_box, lag = 8, dof = 2), augment(select(fit, arima014011)) |>
    features(.innov, ljung_box, lag = 8, dof = 5))
```

```
## # A tibble: 2 x 3
##   .model      lb_stat lb_pvalue
##   <chr>         <dbl>     <dbl>
## 1 auto           1.59     0.954
## 2 arima014011    1.21     0.752
```

From the residual plots, the ACF plots for both models provide strong evidence that the residuals are white noise. Additionally, while the residuals are not exactly normal because of the outliers previously discussed, they look reasonably normal.The Box-Ljung test for both the models provides formal statistical evidence that these residual series are white noise.

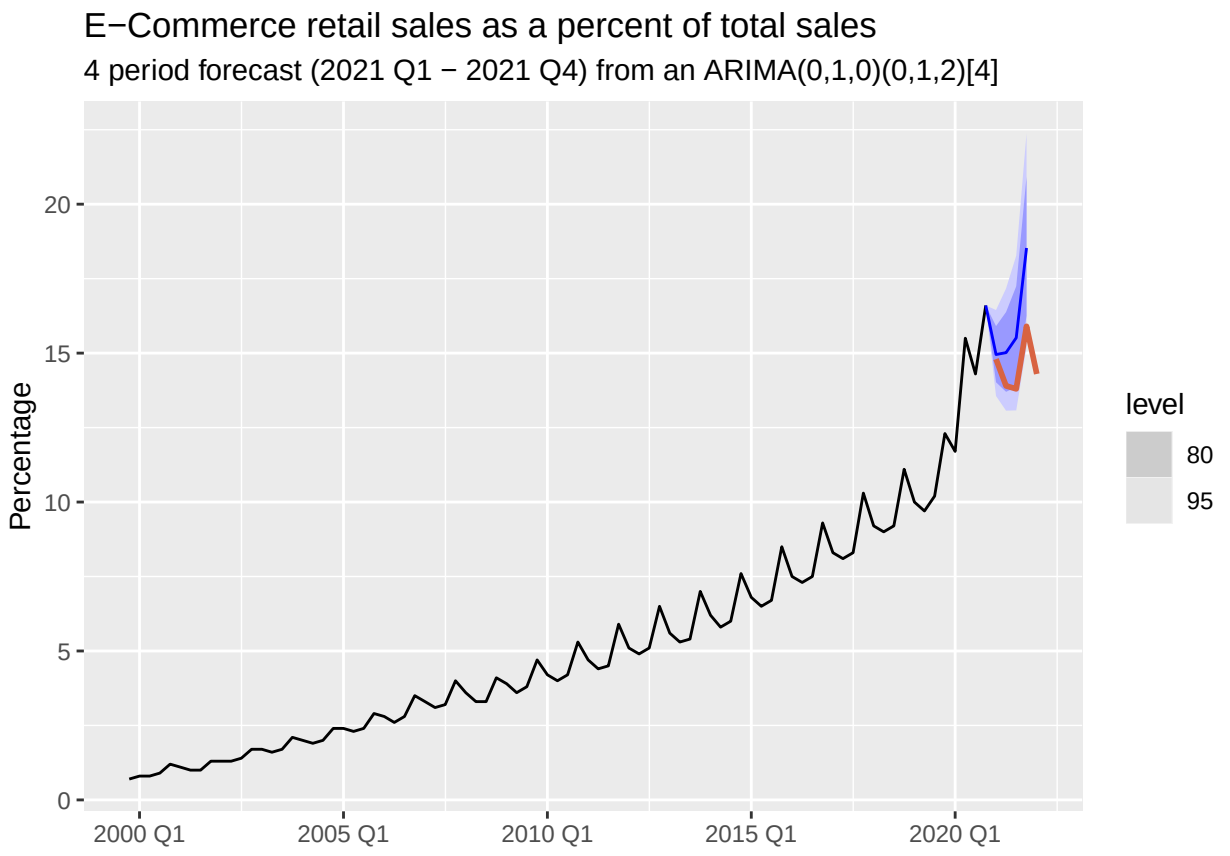
## 2.5 Forecasting

Use both models to forecast the next 12 months and evaluate the forecast accuracy of these models.

```
# Fill this in
fc <- fit |>
    fabletools::forecast(h = 4)
```

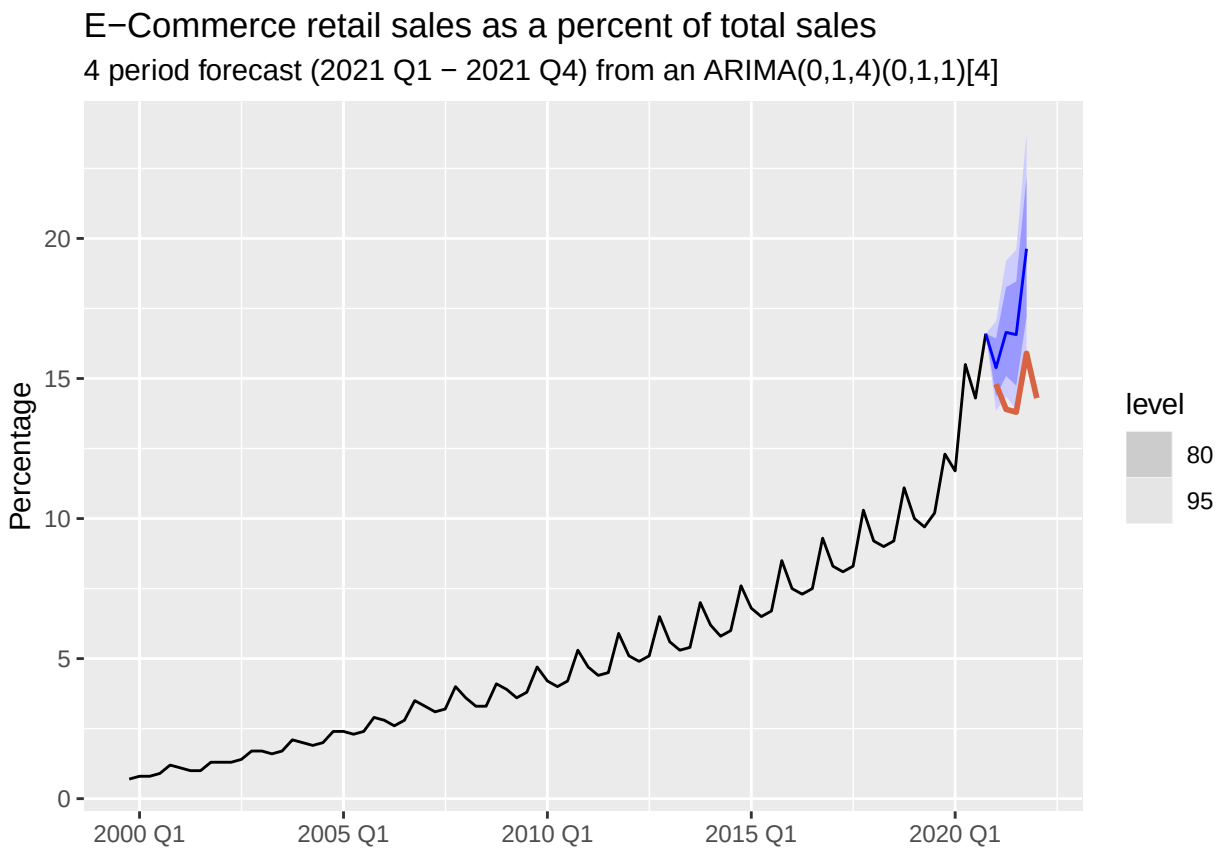
```
ects %>%
    autoplot(ecompctnsa) + autolayer(filter(fc, .model == "auto"), data = ects, show_gap = F) +
    autolayer(ects.test, colour = "#d86342", size = 1) + labs(title = "E-Commerce retail sales as a percen
    subtitle = "4 period forecast (2021 Q1 - 2021 Q4) from an ARIMA(0,1,0)(0,1,2)[4]",
    x = NULL, y = "Percentage")
```

```
## Plot variable not specified, automatically selected `.vars = ecompctnsa`
```




```
ects %>%
    autoplot(ecompctnsa) + autolayer(filter(fc, .model == "arima014011"), data = ects,
    show_gap = F) + autolayer(ects.test, colour = "#d86342", size = 1) + labs(title = "E-Commerce retail s
    subtitle = "4 period forecast (2021 Q1 - 2021 Q4) from an ARIMA(0,1,4)(0,1,1)[4]",
    x = NULL, y = "Percentage")
```

```
## Plot variable not specified, automatically selected `.vars = ecompctnsa`
```




In the figures we show the result of the forecast for both the automatically selected and the manually-input models. In orange is the actual realised values for the same period. We can see the actual values in the next table.

```
fc %>%
    hilo() %>%
    filter(.model != "arima014111") %>%
    select(-ecompctnsa)
```

```
## # A tsibble: 8 x 5 [1Q]
## # Key:       .model [2]
##   .model      quarter .mean                 `80%`                 `95%`
##   <chr>         <qtr> <dbl>                <hilo>                <hilo>
## 1 arima014011 2021 Q1  15.4 [14.35538, 16.43605]80 [13.85020, 17.03556]95
## 2 arima014011 2021 Q2  16.6 [15.09175, 18.25810]80 [14.34980, 19.20212]95
## 3 arima014011 2021 Q3  16.6 [14.74832, 18.46531]80 [13.89651, 19.59718]95
## 4 arima014011 2021 Q4  19.6 [17.19848, 22.19036]80 [16.07668, 23.73876]95
## 5 auto        2021 Q1  15.0 [14.02128, 15.90714]80 [13.56069, 16.44742]95
## 6 auto        2021 Q2  15.0 [13.70129, 16.37821]80 [13.06914, 17.17041]95
## 7 auto        2021 Q3  15.5 [13.85725, 17.24253]80 [13.07833, 18.26946]95
## 8 auto        2021 Q4  18.5 [16.25409, 20.92046]80 [15.20376, 22.36571]95
```

From the forecast plots, it is immediately obvious that neither of the forecasts are well suited to the actual values. This is in line with our expectation prior to the analysis, as our training horizon ended along the 1st quarter of 2020, which already was exhibiting anomalous behaviour. The model can only capture what it assumes is a polynomial trend in the data, but in actual fact, e-commerce sales stabilised during 2021 as the COVID-19 lockdown continued into its second year.

### 2.5.1 Evaluating accuracy

```
fabletools::accuracy(fc, ects.orig) |>
    filter(.model != "arima014111") |>
    arrange(.model) |>
    select(.model, .type, RMSE, MAE, MAPE, MASE, RMSSE)
```

```
## # A tibble: 2 x 7
##   .model      .type  RMSE   MAE  MAPE  MASE RMSSE
##   <chr>       <chr> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 arima014011 Test   2.71  2.46 16.8   3.62  2.49
## 2 auto        Test   1.67  1.40  9.50  2.07  1.53
```

In terms of accuracy measures, we can observe from the table that across the all measures included here the auto model performed better. It's interesting to note that the automatically-selected model could not really capture that seasonal aspect of the original series, while the manually-selected one could. Nonetheless, it's clear that while none of the models performed particularly well, he auto model does slightly better.

# 3 (10 points) Time Series Linear Model and Cointegration

Daily electricity demand and temperature (in degrees Celsius) is recorded in `./data/temperature_demand.csv`. Please work through the following questions to build a time series linear model against this data.

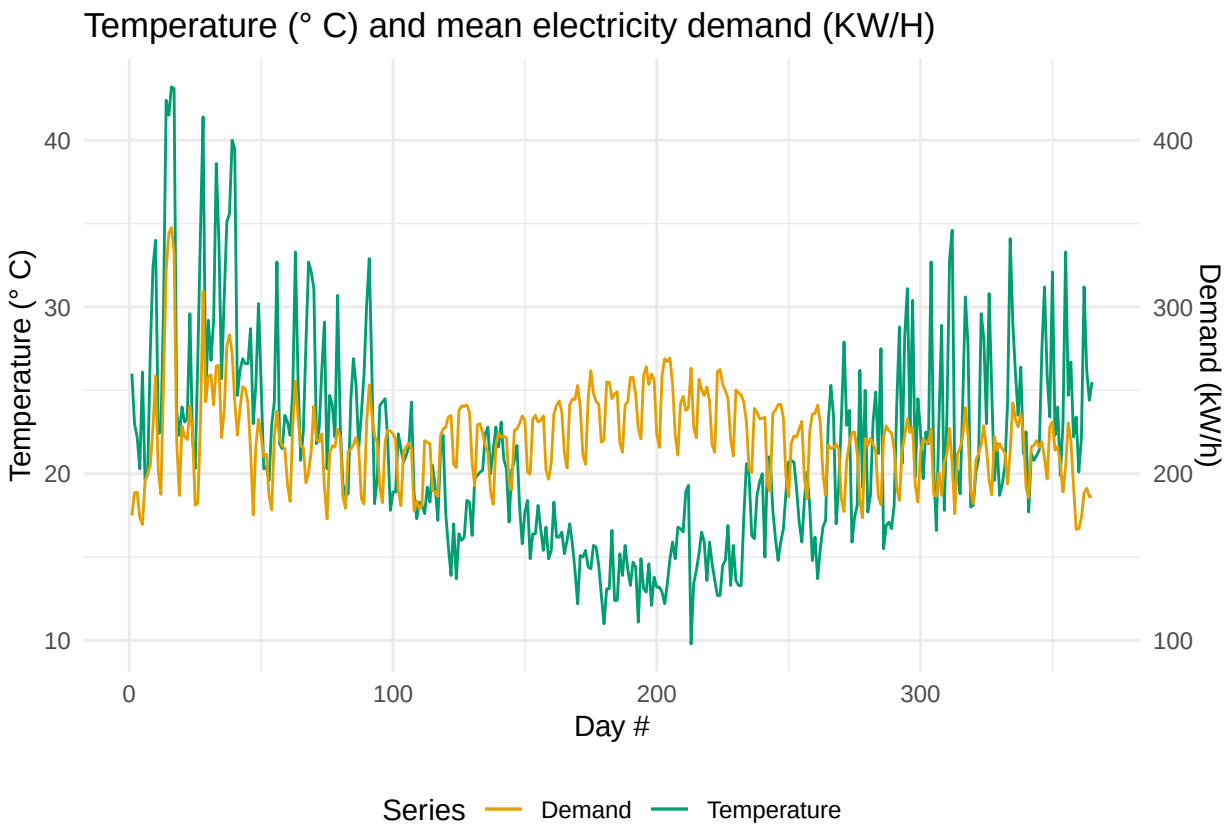
```
library(tidyverse)
# library(fpp3)
```

```
temperature <- read_csv("./data/temperature_demand.csv") %>%
    rename(index = "...1", demand = "Demand", work_day = "WorkDay", temperature = "Temperature")
temp <- as_tsibble(temperature, index = index, regular = T)
# glimpse(temperature)
```

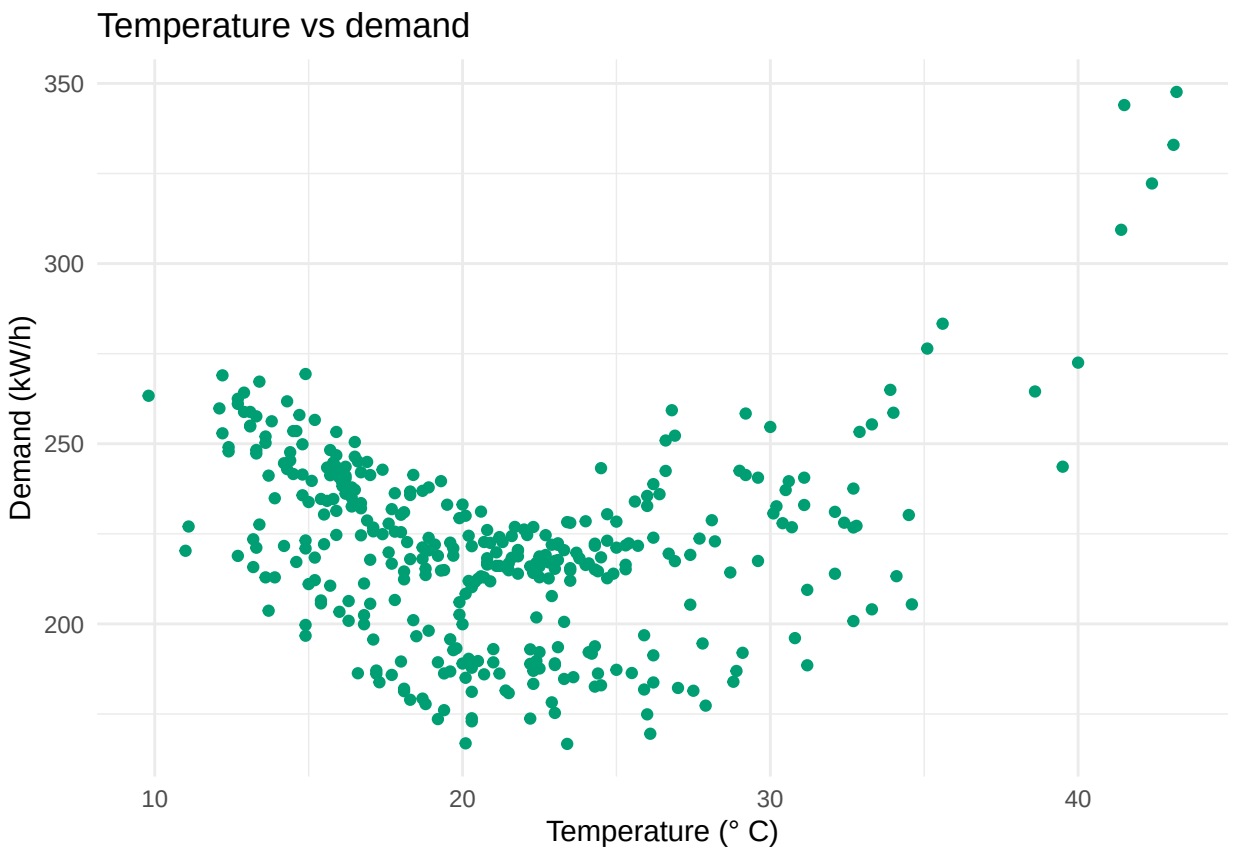
## 3.1 Plot electricity

Plot electricity demand and temperature as time series. Is there any correlation between these to variables? If yes, Do you think is it a spurious correlation?

```
ggplot(temp, aes(x = index)) + geom_line(aes(y = temperature, colour = "Temperature")) +
    geom_line(aes(y = demand/10, colour = "Demand")) + scale_y_continuous(name = "Temperature (° C)",
    sec.axis = sec_axis(~. * 10, name = "Demand (kW/h)")) + labs(x = "Day #", title = "Temperature (° C) a
    colour = "Series") + theme_minimal() + scale_colour_manual(values = c(Demand = "#e69f00",
    Temperature = "#009e73")) + theme(legend.position = "bottom")
```




```
ggplot(temp, aes(temperature, demand)) + geom_point(colour = "#009e73") + theme_minimal() +
    labs(title = "Temperature vs demand", x = "Temperature (° C)", y = "Demand (kW/h)")
```




From the plots we can see that there is a clear correlation between demand and temperature. The relationship is not exactly linear, as it has a clear curve, but a polynomial of degree 2 might reasonably approximate it. From experience, we can probably assert that this correlation is not spurious, and is actually a result of the use of climate-adjusting technology in households and businesses, which consume a great amount of energy. Naturally, the hotter it gets, the more people will use their ACs, which consume a significant amount of energy, which is why there's a rapid climb in demand as temperatures increase. An interesting feature of the correlation is the left half of the scatter plot and the center portion of the time plots. The data show that the same is true on the other extreme: as temperatures start to go down, people will also use more electricity, by means of the usage of heaters, which also consume high electricity.

## 3.2 Cointegration test

Use the Engle-Granger test to check for cointegration. What do you conclude?

```
coint_reg <- lm(log(demand) ~ log(temperature), data = temp)
coef(summary(coint_reg))
```

```
##                     Estimate Std. Error   t value      Pr(>|t|)
## (Intercept)       5.46179708 0.06751336 80.899496 2.612444e-234
## log(temperature) -0.02298349 0.02227778 -1.031678  3.029099e-01
```

```
coint_res <- ts(coint_reg$res)
tseries::pp.test(coint_res)
```

```
## Warning in tseries::pp.test(coint_res): p-value smaller than printed p-value
```

```
##
##  Phillips-Perron Unit Root Test
##
## data:  coint_res
```

```
## Dickey-Fuller Z(alpha) = -106.82, Truncation lag parameter = 5, p-value = 0.01
## alternative hypothesis: stationary
```

```
# Phillips-Ouliaris cointegration test
tseries::po.test(mutate(temp, log.temp = log(temperature), log.dem = log(demand)) %>%
    select(log.temp, log.dem))
```

```
## Warning in tseries::po.test(mutate(temp, log.temp = log(temperature), log.dem = log(demand)) %>% : p-va
```

```
##
##  Phillips-Ouliaris Cointegration Test
##
## data:  mutate(temp, log.temp = log(temperature), log.dem = log(demand)) %>%     select(log.temp, log.de
## Phillips-Ouliaris demeaned = -73.443, Truncation lag parameter = 3, p-value = 0.01
```

Both the Engle-Granger and the Phillips Ouliaris test reject the null hypothesis that there is no cointegration, and so we can assert there *is* cointegration.

## 3.3 Fit Model

Based on cointegration test, fit a regression model for demand with temperature as an explanatory variable (or their first difference).

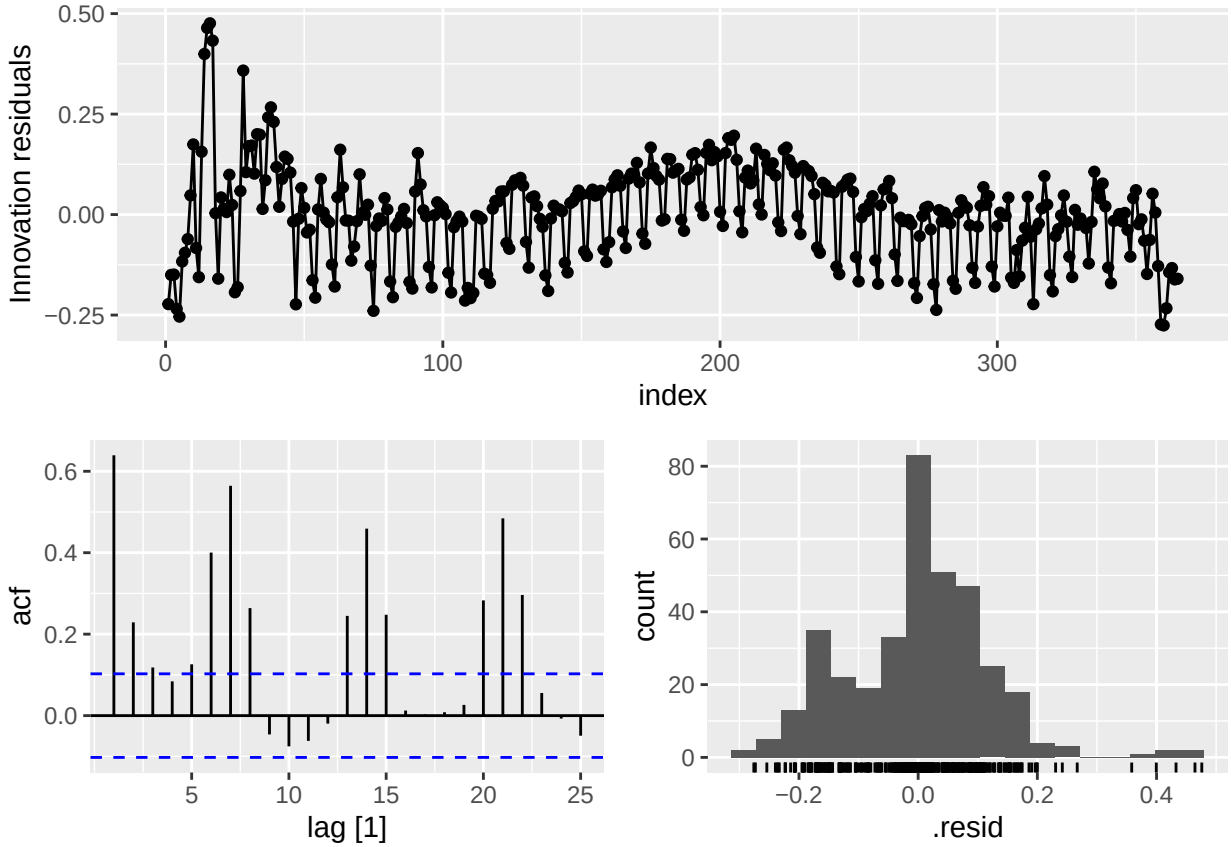
```
m1 <- temp %>%
    model(auto = TSLM(log(demand) ~ log(temperature)))
report(m1)
```

```
## Series: demand
## Model: TSLM
## Transformation: log(demand)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.275695 -0.068629  0.004688  0.073285  0.475915
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)       5.46180    0.06751  80.899   <2e-16 ***
## log(temperature) -0.02298    0.02228  -1.032    0.303
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1177 on 363 degrees of freedom
## Multiple R-squared: 0.002924,    Adjusted R-squared: 0.0001768
## F-statistic: 1.064 on 1 and 363 DF, p-value: 0.30291
```

## 3.4 Residuals Plot

Produce a residual plot of the estimated model in previous part. Is the model adequate? Describe any outliers or influential observations, and discuss how the model could be improved.

```
# Fill this in
m1 %>%
    gg_tsresiduals()
```




The residuals are not white noise, and there is clearly a trend that's not being captured by the linear model. This is evident from the original plot, since the data behave different at the extremes (very high or very low temperatures) for demand than at the middle temperatures. We can see that behaviour clearly in the residuals plot because for more tepid temperatures, the residuals are small and clustered around 0, whereas at the extremes (the first few data points, when temperature was high in the original series), the residuals are significantly bigger. A quasdratic term could better capture the underlying behaviour for this model.

## 3.5 Forecasting model

Use a model to forecast the electricity demand (with **prediction** intervals) that you would expect for the next day if the maximum temperature was 15°. Compare this with the forecast if the maximum temperature was 35°. Do you believe these forecasts? Why or why not?

```
temp.nd <- new_data(temp, n = 2)
temp.nd$temperature <- c(15, 35)
fabletools::forecast(m1, temp.nd) %>%
    hilo()
```

```
## # A tsibble: 2 x 7 [1]
## # Key:       .model [1]
##   .model index           demand .mean temperature                  `80%`                  `95%`
##   <chr>  <dbl>           <dist> <dbl>       <dbl>                 <hilo>                 <hilo>
## 1 auto     366 t(N(5.4, 0.014))  223.          15 [190.2400, 257.4503]80 [175.6001, 278.9142]95
## 2 auto     367 t(N(5.4, 0.014))  219.          35 [186.4744, 252.6162]80 [172.0770, 273.7523]95
```

- For the 15°, the model predicts a demand of 222.85 with a 95% CI of $[175.6, 278.91]$.
- For the 35°, the model predicts a demand of 218.56 with a 95% CI of $[172.07, 273.75]$.

Overall, we do not believe these forecasts, because as we have mentioned, the model performs poorly at large or small temperature values. In this case, the assertion being made is that at 15 degrees, the mean consumption would

be *bigger* than at 35 degrees, which is significantly hotter. This does not hold up to our intuition that people use more energy as temperature increases. It's possible the 15 degree prediction, being more tepid temperature, might be more believable, however, the given CI is quite large.

# 4 (12 points) Vector autoregression

```
library(tidyverse)
library(vars)
library(patchwork)
```

Annual values for real mortgage credit (RMC), real consumer credit (RCC) and real disposable personal income (RDPI) for the period 1946-2006 are recorded in `./data/mortgage_credit.csv`.

All of the observations are measured in billions of dollars, after adjustment by the Consumer Price Index (CPI).

Our goal is to develop a VAR model for these data for the period 1946-2003, and then forecast the last three years, 2004-2006.

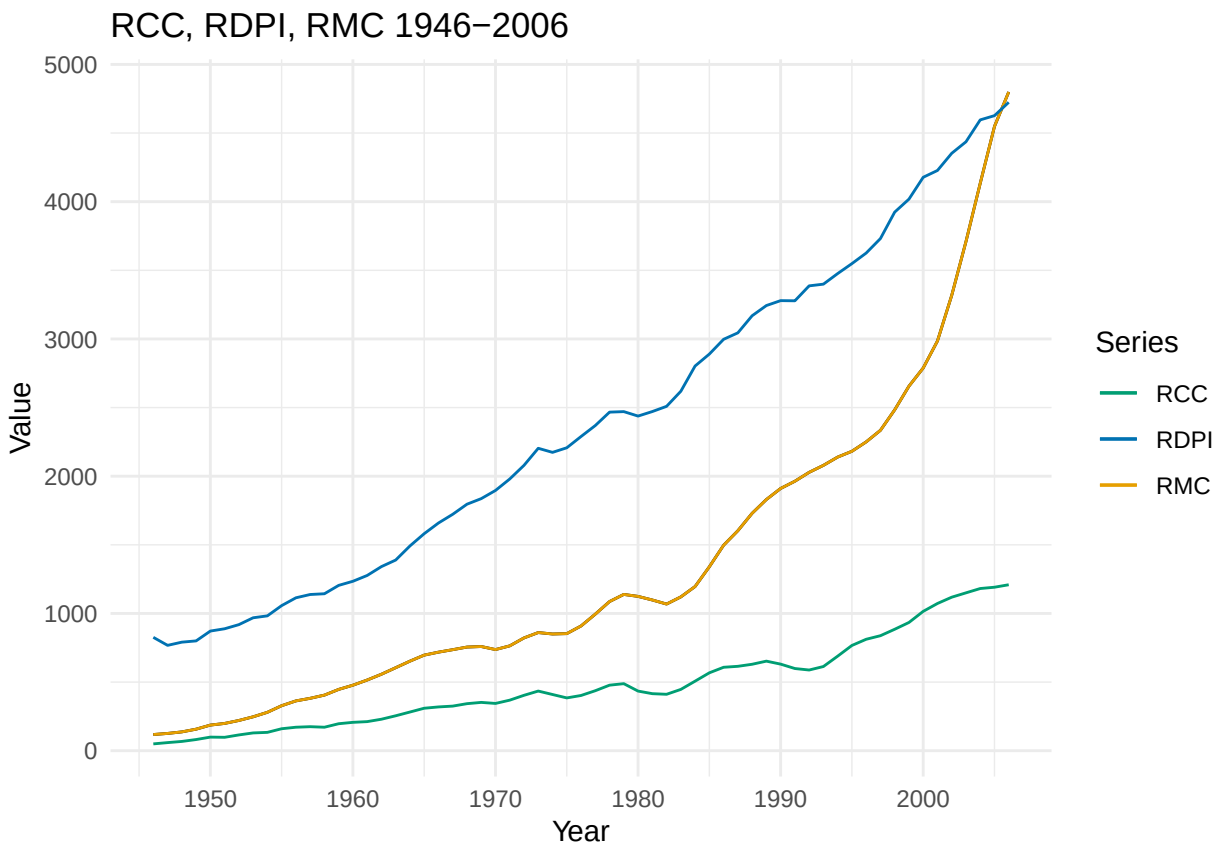
```
credit <- read_csv("./data/mortgage_credit.csv")
names(credit) <- tolower(names(credit))
# glimpse(credit)
credit <- as_tsibble(credit, index = year, regular = T)
```

## 4.1 Time series plot

Plot the time-series of real mortgage credit (RMC), real consumer credit (RCC) and real disposable personal income (RDPI)? Do they look stationary?

```
series.colors <- thematic::okabe_ito(3)
credit %>%
    autoplot() + geom_line(aes(y = rmc, color = "RMC")) + geom_line(aes(y = rcc,
    color = "RCC")) + geom_line(aes(y = rdpi, color = "RDPI")) + labs(x = "Year",
    y = "Value", color = "Series", title = "RCC, RDPI, RMC 1946-2006") + theme_minimal() +
    scale_color_manual(values = c(RMC = series.colors[1], RCC = series.colors[2],
        RDPI = series.colors[3])) + scale_x_continuous(n.breaks = 10)
```

```
## Plot variable not specified, automatically selected `.vars = rmc`
```



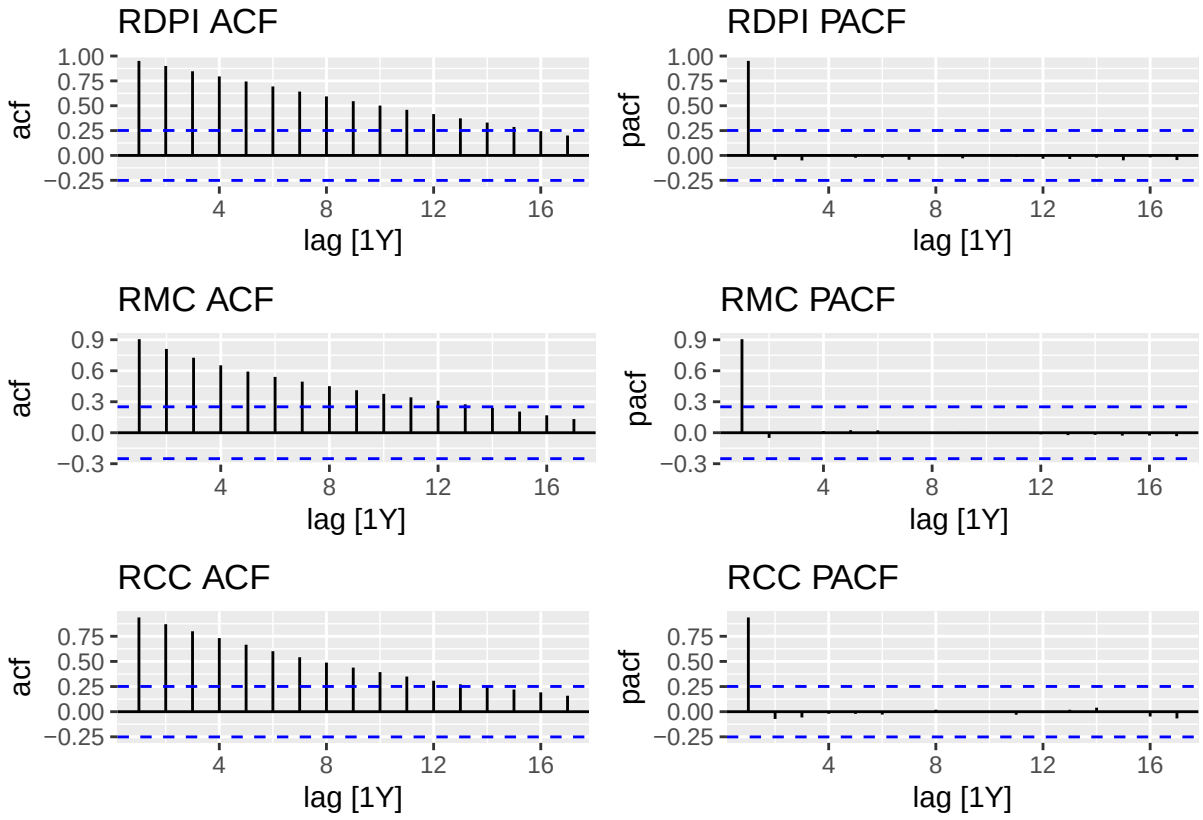

These do not look stationary given they have a clear trend.

## 4.2 Check for the unit root

Plot ACF/PACF and Perform the unit root test on these variables and report the results. Do you reject the null of unit root for them? Is the first differencing necessary?

```
rdpi.acf <- credit %>%
    ACF(rdpi) %>%
    autoplot() + labs(title = "RDPI ACF")
rdpi.pacf <- credit %>%
    PACF(rdpi) %>%
    autoplot() + labs(title = "RDPI PACF")
rmc.acf <- credit %>%
    ACF(rmc) %>%
    autoplot() + labs(title = "RMC ACF")
rmc.pacf <- credit %>%
    PACF(rmc) %>%
    autoplot() + labs(title = "RMC PACF")
rcc.acf <- credit %>%
    ACF(rcc) %>%
    autoplot() + labs(title = "RCC ACF")
rcc.pacf <- credit %>%
    PACF(rcc) %>%
    autoplot() + labs(title = "RCC PACF")

(rdpi.acf + rdpi.pacf)/(rmc.acf + rmc.pacf)/(rcc.acf + rcc.pacf)
```




```
bind_rows(credit %>%
    features(rdpi, list(unitroot_kpss, unitroot_nsdiffs, unitroot_ndiffs)) %>%
    mutate(series = "RDPI", .before = kpss_stat), credit %>%
    features(rcc, list(unitroot_kpss, unitroot_nsdiffs, unitroot_ndiffs)) %>%
    mutate(series = "RCC", .before = kpss_stat), credit %>%
    features(rmc, list(unitroot_kpss, unitroot_nsdiffs, unitroot_ndiffs)) %>%
    mutate(series = "RMC", .before = kpss_stat))
```

```
## # A tibble: 3 x 5
##   series kpss_stat kpss_pvalue nsdiffs ndiffs
##   <chr>      <dbl>       <dbl>   <int>  <int>
## 1 RDPI        1.60        0.01       0      2
## 2 RCC         1.49        0.01       0      1
## 3 RMC         1.42        0.01       0      2
```

From the ACF and PACF plots, all of series exhibit the same behaviour: slowly decaying ACF and a single spike in the PACF. It's unsurprising then, that the KPSS test rejects the null of stationarity and tell us that we will need second-order differencing for RDPI and RMC, and first-order differencing for RCC (which is expected to remove the trend).

## 4.3 Determine VAR model

Based on the unit root results transform the variables and determine the lag length of the VAR using the information criteria.

```
cdiff <- credit %>%
    mutate(rcc.diff = difference(rcc), rmc.diff = difference(rmc, differences = 2),
        rdpi.diff = difference(rdpi, differences = 2)) %>%
    data.frame() %>%
```

```
    dplyr::select(ends_with("diff")) %>%
    slice_tail(n = -3)  # remove the NA values
```

```
VARselect(cdiff, lag.max = 4, type = "none")
```

```
## $selection
## AIC(n)  HQ(n)  SC(n) FPE(n)
##      1      1      1      1
##
## $criteria
##                   1            2            3            4
## AIC(n) 2.080289e+01 2.090098e+01 2.096298e+01 2.089217e+01
## HQ(n)  2.093074e+01 2.115667e+01 2.134652e+01 2.140356e+01
## SC(n)  2.113439e+01 2.156397e+01 2.195747e+01 2.221816e+01
## FPE(n) 1.083257e+09 1.197787e+09 1.282911e+09 1.211102e+09
```

All the information criteria suggest the adequate lag length is 1.

## 4.4 Estimation

Estimate the selected VAR in previous part and comment on the results.

```
var.diff <- VAR(cdiff, p = 1, type = "none")
summary(var.diff)
```

```
##
## VAR Estimation Results:
## ========================
## Endogenous variables: rcc.diff, rmc.diff, rdpi.diff
## Deterministic variables: none
## Sample size: 57
## Log Likelihood: -824.256
## Roots of the characteristic polynomial:
## 0.6628 0.6416 0.3568
## Call:
## VAR(y = cdiff, p = 1, type = "none")
##
##
## Estimation results for equation rcc.diff:
## =========================================
## rcc.diff = rcc.diff.l1 + rmc.diff.l1 + rdpi.diff.l1
##
##               Estimate Std. Error t value Pr(>|t|)
## rcc.diff.l1  0.658501   0.100246   6.569 2.06e-08 ***
## rmc.diff.l1  0.174111   0.098613   1.766   0.0831 .
## rdpi.diff.l1 0.003597   0.052574   0.068   0.9457
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 21.84 on 54 degrees of freedom
## Multiple R-Squared: 0.5901,  Adjusted R-squared: 0.5673
## F-statistic: 25.91 on 3 and 54 DF,  p-value: 1.609e-10
##
##
## Estimation results for equation rmc.diff:
## =========================================
```

```
## rmc.diff = rcc.diff.l1 + rmc.diff.l1 + rdpi.diff.l1
##
##              Estimate Std. Error t value Pr(>|t|)
## rcc.diff.l1   0.03629    0.18501   0.196    0.845
## rmc.diff.l1   0.32473    0.18200   1.784    0.080 .
## rdpi.diff.l1  0.06121    0.09703   0.631    0.531
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 40.3 on 54 degrees of freedom
## Multiple R-Squared: 0.1102,  Adjusted R-squared: 0.06073
## F-statistic: 2.228 on 3 and 54 DF,  p-value: 0.09534
##
##
## Estimation results for equation rdpi.diff:
## ==========================================
## rdpi.diff = rcc.diff.l1 + rmc.diff.l1 + rdpi.diff.l1
##
##              Estimate Std. Error t value Pr(>|t|)
## rcc.diff.l1   -0.5382     0.2199  -2.448   0.0176 *
## rmc.diff.l1    0.5222     0.2163   2.414   0.0192 *
## rdpi.diff.l1  -0.6053     0.1153  -5.250 2.63e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 47.89 on 54 degrees of freedom
## Multiple R-Squared: 0.3804,  Adjusted R-squared: 0.346
## F-statistic: 11.05 on 3 and 54 DF,  p-value: 9.177e-06
##
##
##
## Covariance matrix of residuals:
##           rcc.diff rmc.diff rdpi.diff
## rcc.diff     444.1    289.3     634.7
## rmc.diff     289.3   1623.3     715.3
## rdpi.diff    634.7    715.3    2217.0
##
## Correlation matrix of residuals:
##           rcc.diff rmc.diff rdpi.diff
## rcc.diff    1.0000   0.3407    0.6396
## rmc.diff    0.3407   1.0000    0.3771
## rdpi.diff   0.6396   0.3771    1.0000
```

From eq (1), we get that the lagged RCC values are predictive of the current RCC values (significant a the .05 alpha), and a small predictiveness of the RMC at the 10% level.From eq (2), we get that lagged values of RMC are predictive of it at the 10% significance level, but not much else. From eq (3), we get that the lagged values of RDPI are strongly predictive of it (significant at the .001 alpha level), as well as RMC and RCC being predictive of it (at the .05 alpha significance level).

Therefore, we can assert that RMC, RCC, and RDPI lags could be predictive of the RDPI value, but the same is not true for RCC or RMC insofar as their relationship to the other two series. If anything, lagged RCC values could be predictive of current RCC values.

## 4.5 Model diagnostic

Do diagnostic checking of the VAR model.

```
roots(var.diff)
```

```
## [1] 0.6627614 0.6415831 0.3567547
```

```
serial.test(var.diff, lags.pt = 12)
```

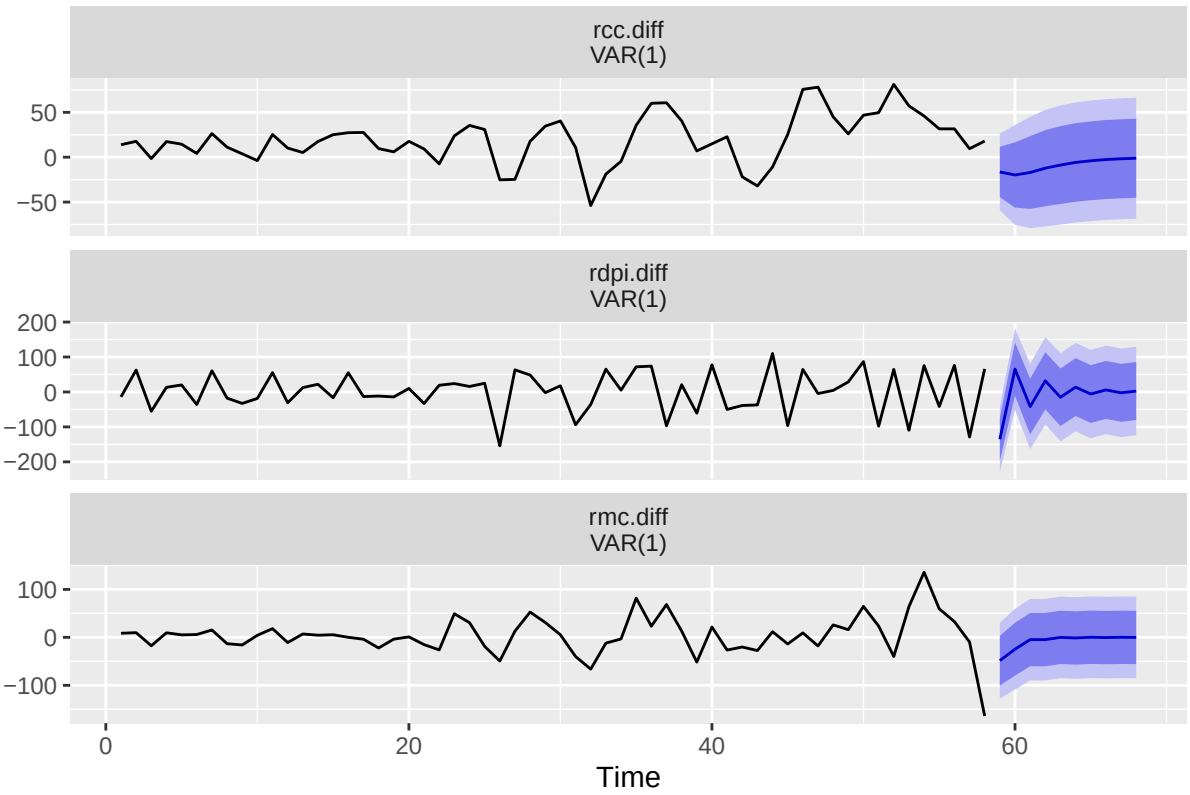
```
##
##  Portmanteau Test (asymptotic)
##
## data:  Residuals of VAR object var.diff
## Chi-squared = 130.26, df = 99, p-value = 0.01925
```

Given that all the eigenvalues of the companion matrix are less than 1, we can assert that VAR(1) is a stable process. Running the Portmanteau test, we get a p-value of .01, which is significant at an alpha of 0.05, hence we reject the null of no autocorrelation in the residuals.

## 4.6 Forecasting

forecast the last three years, 2004-2006.

```
var.diff <- VAR(as.ts(cdiff), p = 1, type = "none")
forecast(var.diff) %>%
    autoplot()
```




We can see that the VAR(1) model is not able to capture the variance really well except maybe for RDPI, which is expected, because the other series are not predictive amongst themselves of any of the other series.