Forecasting & Causal Analysis of Bicycle Rentals

Lachezar Popov

ADS, 2021-2022

Contents

0. Prepare

▶ Load the R-packages you will use.

```
library(tidyverse)
library(fpp3)
library(tseries)
library(expsmooth)
```

▶ Include R-code you used to load (and prepare) the data.

```
# reading the dataset
bike_rentals = read_csv('data/bike_rentals_merged.csv')

#transforming variables to their appropriate type
bike_rentals$timestamp = as.POSIXct(bike_rentals$timestamp, format = "%d/%m/%Y %H:%M")
bike_rentals$weather_code = as.factor(bike_rentals$weather_code)
bike_rentals$is_weekend = as.factor(bike_rentals$is_weekend)

# transforming dataset to tsibble
bike_rentals = as_tsibble(bike_rentals, index=timestamp)
```

1. General

▶ To be able to use fpp3, the data have to be a tsibble object. If they aren't already, transform them. Describe the structure of this object.

The dataframe was transformed to a tsibble object in the previous section of this notebook. A tsibble object is an extension of the tibble object from the tidyverse package with added structure for analyzing temporal data. A tsibble object has the timestamp as it's index and (optionally) a key consisting of one or more columns for identifying the cases/persons/dyads. As the bike rentals dataset has only 1 case (the city of London), a key is not required. The bike_rentals tsibble object which was created in the previous section has 720 observations of 7 variables (excl. the index).

The tsibble contains hourly bicycle rental data for London's public bike sharing scheme - Santander Cycles, provided by Transport for London (a local governmental body). The dataset also contains weather data such as temperature, humidity and wind speed from freemeteo.com.

Excluding the timestamp, the tsibble object consists of the following variables: * "cnt" - the count of new bike rentals * "t1" - real temperature in °C * "t2" - "feels like" temperature in °C * "hum" - humidity in percentage * "wind_speed" - wind speed in km/h * "weather_code" - weather category * 1: clear / mostly clear * 2: scattered clouds / few clouds * 3: broken clouds * 4: cloudy * 7: rain / light rain * 10: rain with thunderstorm * 26: snowfall * 94: freezing fog * "is_weekend" - boolean array - 1 for Saturday and Sunday / 0 for Monday:Friday

1.1. Describe your data

Start with answering the following questions:

▶ What is your outcome variable; how was it measured (how many times, how frequently, etc.)?

The outcome variable for my analysis is the number of bike rentals - the “cnt” column of the tsibble.

The number of bike rentals were measured on an hourly basis spanning a period of one month - from 08 April 2016 00:00 to 07 May 2017 23:00. In total there are 720 observations of the “cnt” variable.

▶ What are the predictor variable(s) you will consider? Why would this make sense as a predictor?

- **Humidity** (hum): A continuous variable containing air humidity in percentage. Humidity makes sense as a predictor because people might cycle less when the humidity is higher as they may feel hotter and become tired more easily.
- **Wind speed** (wind_speed): A continuous variable containing wind speed in km/h. Wind speed makes sense as a predictor as it is harder to cycle when it is windy.
- **Weather code** (weather_code): A categorical variable for the type of weather (clear, cloudy, rain, snowfall, etc.). It is assumed that bike rentals would be lower when the weather is bad as cyclists would be exposed to the bad weather.
- **Is weekend** (is_weekend): A binary variable showing weather it is a weekend day. The number of bike rentals may be different in weekend days, as opposed to working days, as most people do not have routine commutes during the weekend. Moreover, they would be during the times when they would otherwise be working which may equate to an increase in bike rentals in the 9:00-17:00 hours.
- **Temperature** (t1): A continuous variable for temperature in °C. Temperature may have an effect on bike rentals as people might cycle less when it is too hot or too cold. As we have two variables for temperature in our dataset (real temperature and “feeks like” temperature) we would have to decide which one to include, as using both variables would introduce multicollinearity in our model. I have chosen to include real temperature (t1) and exclude “feels like” temperature (t2) for three main reasons:
  - *Measurement error*: it is expected that real temperature would have lower measurement error than the feels like temperate, as the “feels like” temperature is a function of the real temperature and other variables carrying their own measurement errors. Hence, “feels like” temperature may exhibit compounding of measurement errors.
  - *Independence of other covariates*: it is likely that “feels like” temperature is a function of humidity and wind speed. As we already have these variables in our dataset (and can account for them), using feels like temp becomes unnecessary. Moreover, if we were to use “feels like” temperature instead of real temperature - this would again introduce multicollinearity in our model.
  - *Correlation with bike rentals*: real temperature has a higher correlation with bike rentals than “feels like” temperature as can be seen in the output of the below code chunk.

```
cor(bike_rentals$t1, bike_rentals$cnt)
```

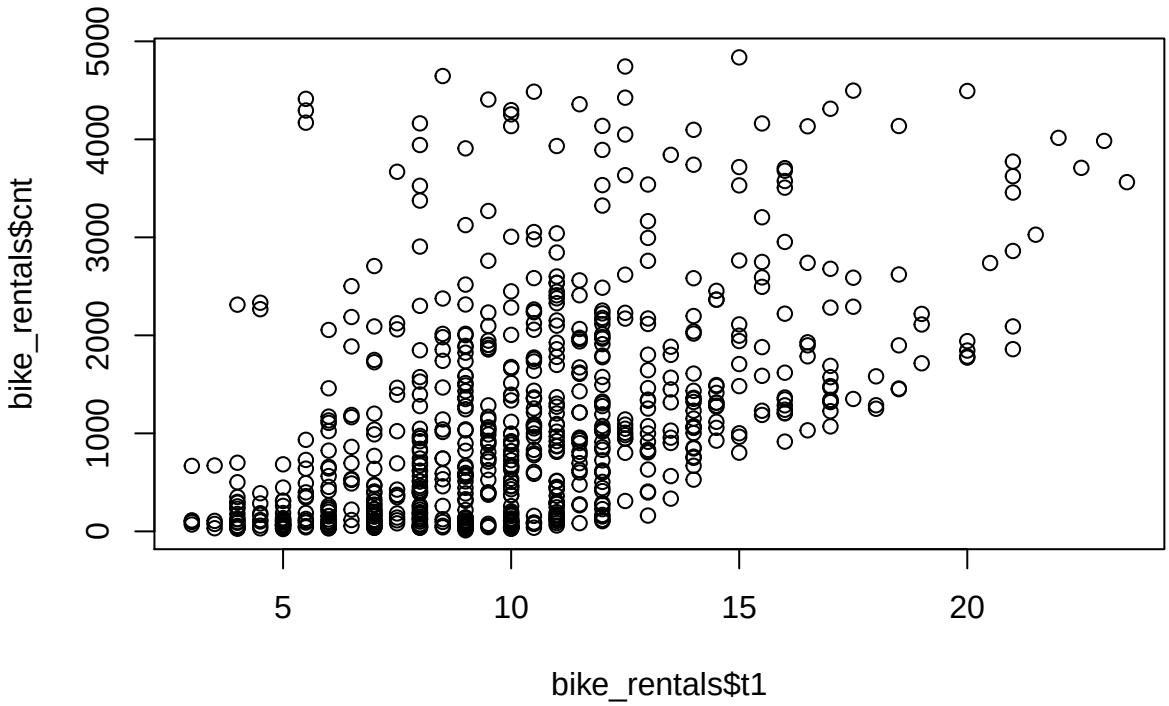
```
## [1] 0.4908311
```

```
cor(bike_rentals$t2, bike_rentals$cnt)
```

```
## [1] 0.4574468
```

Another important aspect to consider is the form of the relationship between temperature and bike rentals. As mentioned previous, bike rentals may decrease when the temperature is too hot **OR** too cold. This suggests that there might be a quadratic relationship with a negative coefficient between bike rentals and temperature centered at the optimally pleasant temperature for cycling. If this were true, we would expect to see something resembling a downwards open parabola on the scatterplot of t1 and cnt, with the vertex lying on the optimally pleasant temperature for cycling. We can plot this and check whether this is indeed the case.

```
plot(bike_rentals$t1, bike_rentals$cnt)
```



We do not see evidence of such a non-linear relationship on the scatterplot. This may be due to the fact that it doesn't really become "too hot" in the time period that comprises our dataset (April 8th to May 7th). Hence there is no need to transform the t1 variable - we can use it as is.

▶ What are the cause(s) you will consider? Why would this make sense as a cause?

I will consider **temperature** as a cause of bike rentals. The reason this makes sense is the same reason stated in the preceding section - too cold or too hot temperature may have a negative effect on bike rentals.

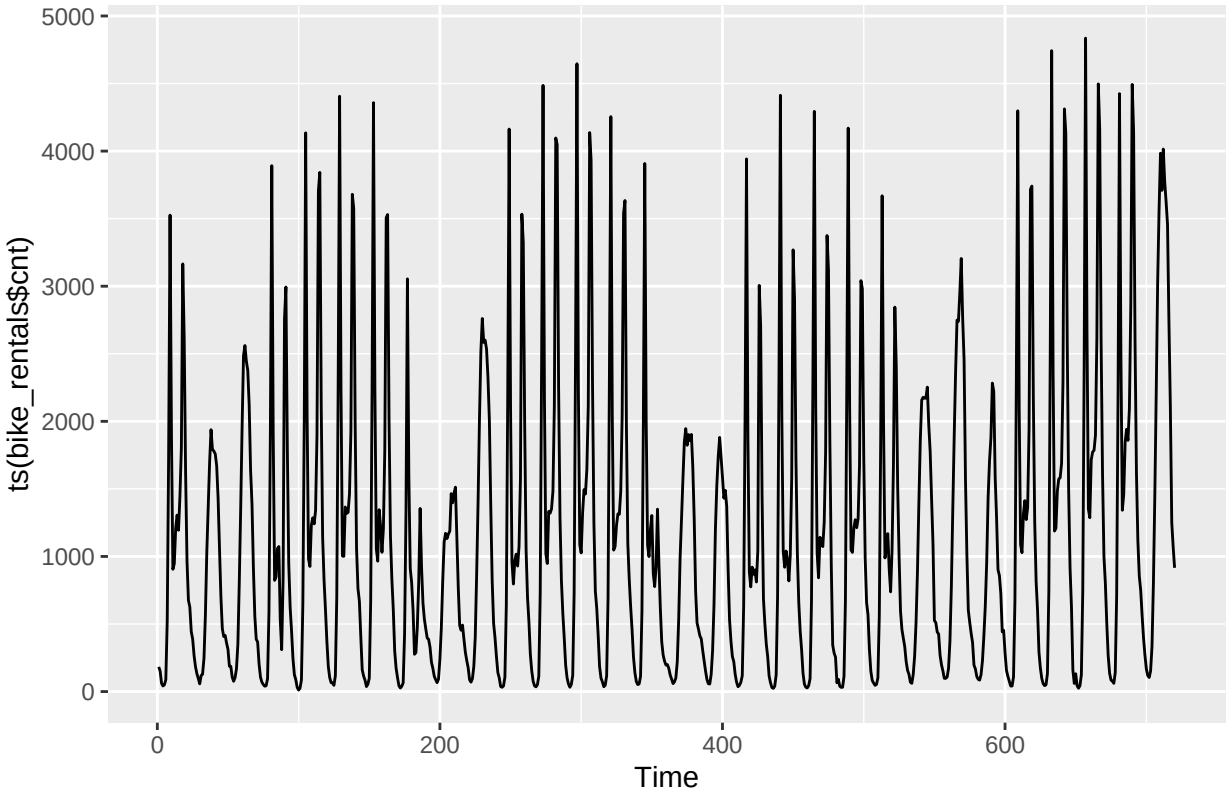
1.2. Visualize your data

▶ Create a sequence plot of the data with the function autoplot(). Interpret the results.

sequence plot of bike rentals

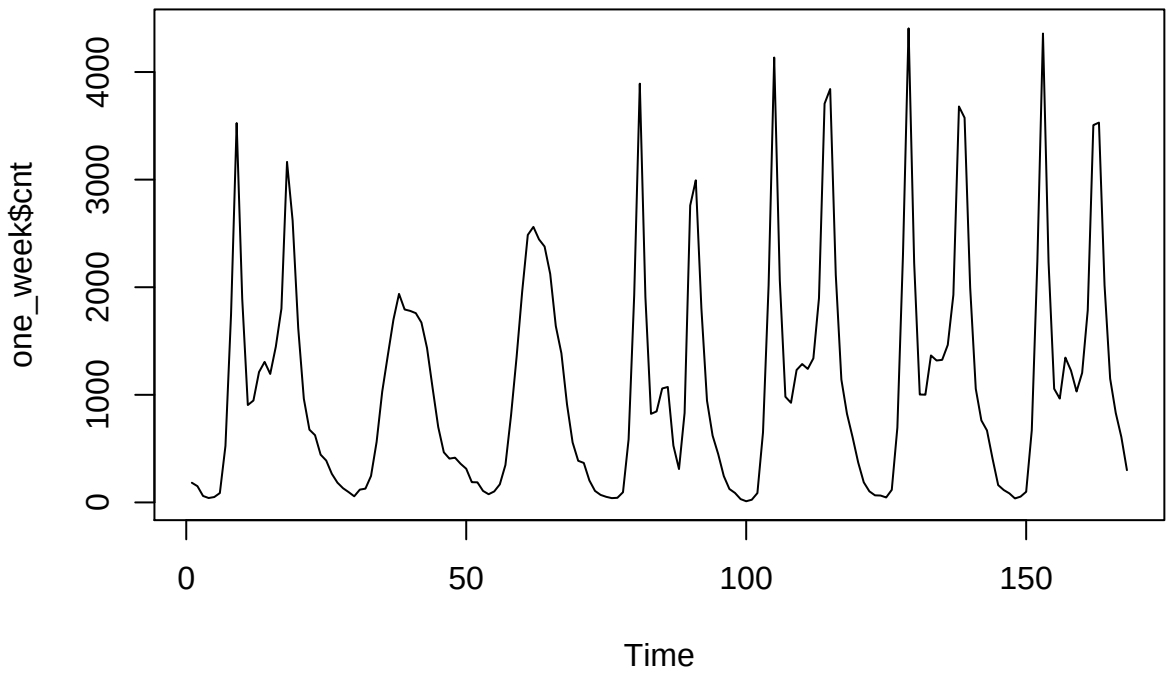
First let us examine a sequence plot of all observations of the “cnt” variable.

```
autoplot(ts(bike_rentals$cnt))
```



We can also zoom in on the first week of the data.

```
one_week = bike_rentals[bike_rentals$timestamp < as.Date('2016-04-15'),]
plot.ts(one_week$cnt)
```



The first day of the one_week plot is a Friday, followed by a Saturday and Sunday. We can see two peaks during each working day (Mon-Fri) and a single peak during weekend days (Sat-Sun). This pattern also appears in the sequence plot of the whole data. The only exception beeing the Monday of the last week (the single peak just before the 600 mark) - perhaps because it was a holiday. Hence, in both plots we can see a clear seasonal pattern. Perhaps even multiple seasonal patterns - one with a period of 24 (daily) and one with a period of 168 (weekly).

Moreover, the peaks during Saturdays and Sundays appear to be consistently lower than the peaks observed on working days. This suggests that the variance in the time series would be lower on weekend days than working days. We can verify this by computing the two variances.

```
cat('Variance during working days: ', as.character(var(bike_rentals[bike_rentals$is_week
```

```
## Variance during working days:  1321764.21059358
```

```
cat('Variance during weekends: ', as.character(var(bike_rentals[bike_rentals$is_weekend
```

```
## Variance during weekends:  922774.110055986
```

Lastly, while the data appear to vary across a somewhat stable mean on working days - the mean appears to drop in the weekend. We can verify this by computing these two means:

```
cat('Mean during working days: ', as.character(mean(bike_rentals[bike_rentals$is_weekend
```

```
## Mean during working days:  1205.4880952381
```

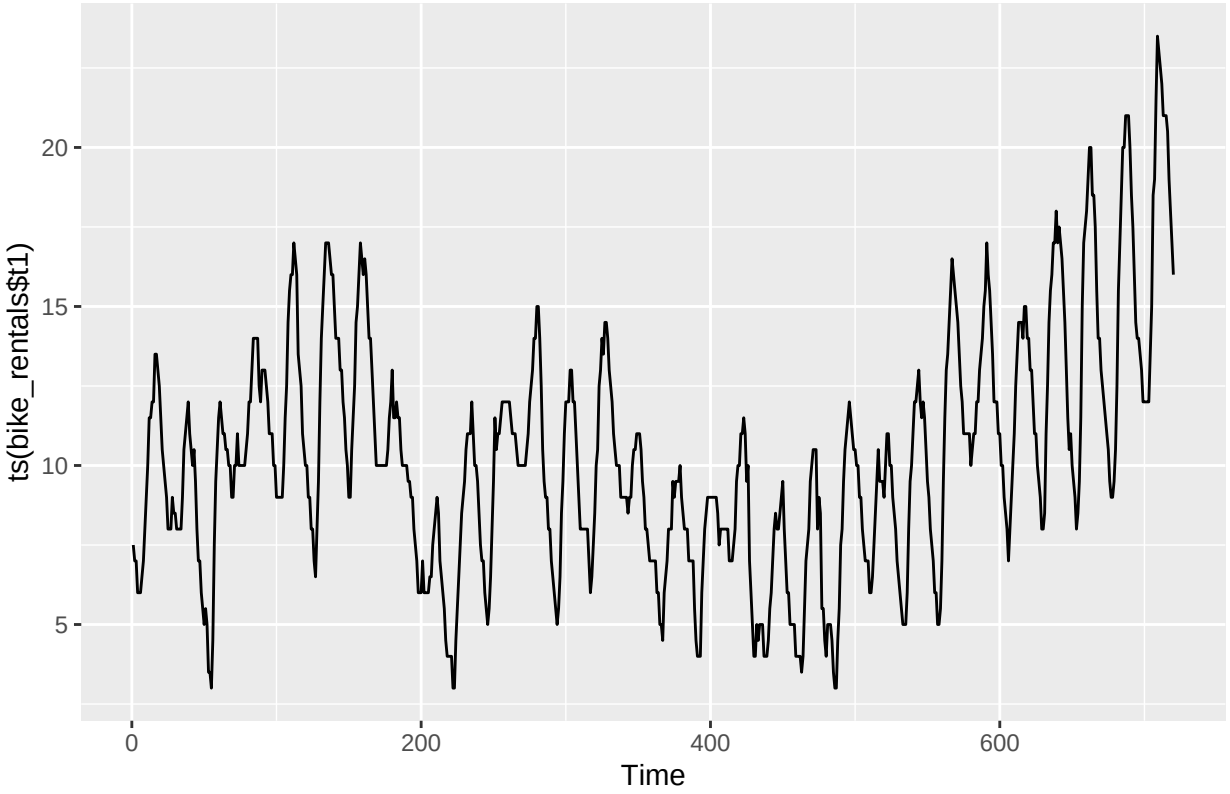
```
cat('Mean during weekends: ', as.character(mean(bike_rentals[bike_rentals$is_weekend ==
```

```
## Mean during weekends:  1044.9212962963
```

The season pattern and change in statistical properties (mean and variance) during the weekend suggest that the data is not stationary.

sequence plot of temperature

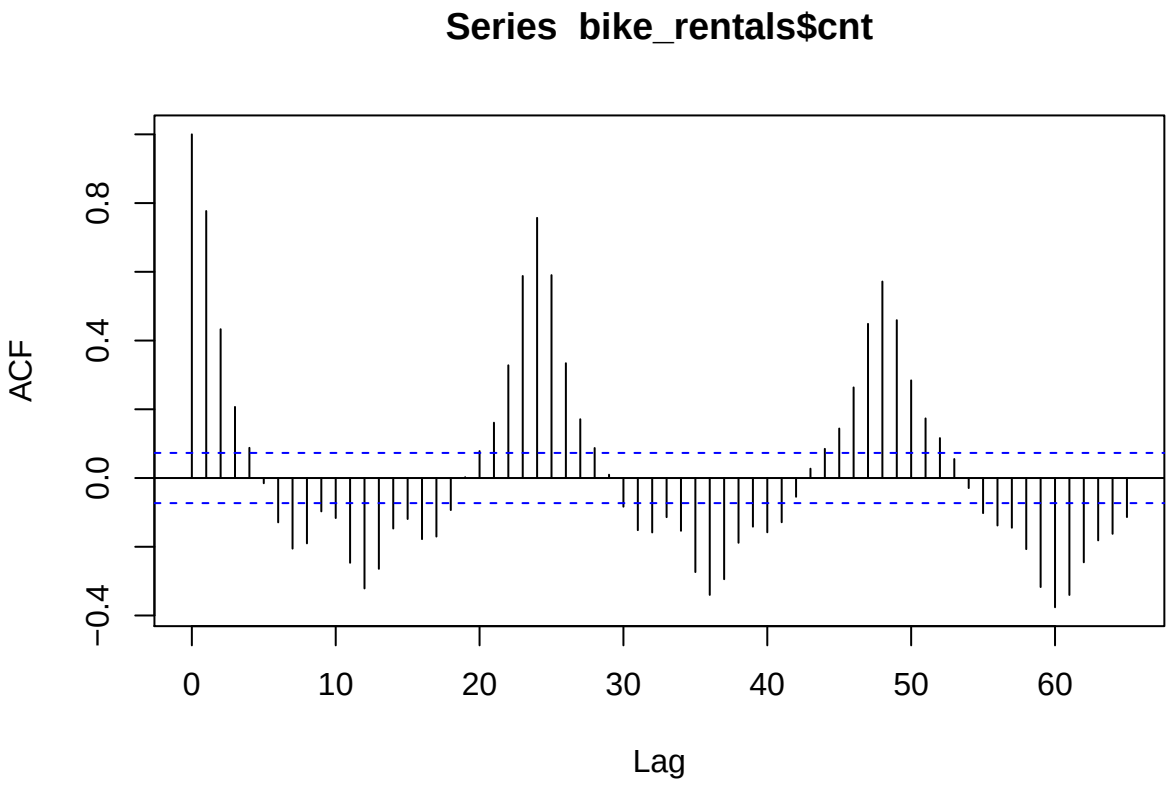
```
autoplot(ts(bike_rentals$t1))
```



The time series of temperature also appears to follow a seasonal pattern. Moreover, it shows a clear upward trend from the 500th observation onward.

▶ Plot the autocorrelation function with the function acf(). Interpret the results.

```
acf(bike_rentals$cnt, lag.max = 65)
```



We can see a sinusoidal pattern with peaks at the 24th and 48th lags and troughs at the 12th, 36th and 60th lags. This is clear evidence of a seasonal pattern with a period of 24. The presence of such a 24-hour (daily) season is not surprising for hourly measurements of human activity (renting bicycles / cycling).

Thus, the data are clearly not stationary and must be seasonally differenced.

▶ Based on (basic) content knowledge about the variable, and these visualizations, is there reason to assume the data are non-stationary and/or that there is a seasonal component?

Yes, both the sequence plots and the ACF plot show clear evidence of non-stationarity in the form of a seasonal a seasonal component with a period of 24.

This is not surprising given that the data concerns bicycle rentals. It is likely the case that people cycle in the morning and evenings (possibly to and from work/school/etc.) and the least during the afternoon and night.

Moreover the sequence plot shows further evidence of non-stationarity in the form of changes to the mean and variance during weekends. This may be due to the fact that people have regular commutes during weekdays but not during the weekend.

2. Forecasting

2.1. SARIMA modeling

▶ Perform the Dickey-Fuller test. What is your conclusion?

```
adf.test(bike_rentals$cnt)
```

```
## Warning in adf.test(bike_rentals$cnt): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  bike_rentals$cnt
## Dickey-Fuller = -8.1371, Lag order = 8, p-value = 0.01
## alternative hypothesis: stationary
```

The null hypothesis for the Dickey-Fuller test is that there is a unit root process present in the time series while the alternative hypothesis is that the data is stationary. Since the p-value is smaller than 0.01 we can reject the null hypothesis at a significance threshold of 0.05. Hence, according to the test the data does not follow a unit root process.

The Dickey-Fuller test suggest that the data is stationary, however, it is important to note that the test only checks for a certain type of non-stationarity (a unit root process). Thus, while there appears to be no unit root component to the time series, the seasonal component evident from the sequence and ACF plots, and the changes in the mean and variance during weekends evident from the sequence plot suggest that the data are not stationary.

▶ Fit an (S)ARIMA model to the data; what is the order of the model that was selected?

```
#bike_rentals = fill_gaps(bike_rentals)
fit_cnt <- bike_rentals %>% model(ARIMA(cnt))
report(fit_cnt)
```

```
## Series: cnt
## Model: ARIMA(2,0,2)(2,1,0)[24]
##
## Coefficients:
##          ar1      ar2      ma1      ma2    sar1     sar2
##       1.2534  -0.4959  -0.1083  -0.2473  0.0113  -0.2493
## s.e.  0.1103   0.0631   0.1166   0.0763  0.0406   0.0392
##
## sigma^2 estimated as 158021:  log likelihood=-5152.52
## AIC=10319.04   AICc=10319.21   BIC=10350.86
```

The model that was selected is an ARIMA(2,0,2)(2,1,0)[24] model.

The coefficients for non-seasonal AR components are somewhat large relative to their standard error, while the MA1 and MA2 components have much smaller coefficients rela-

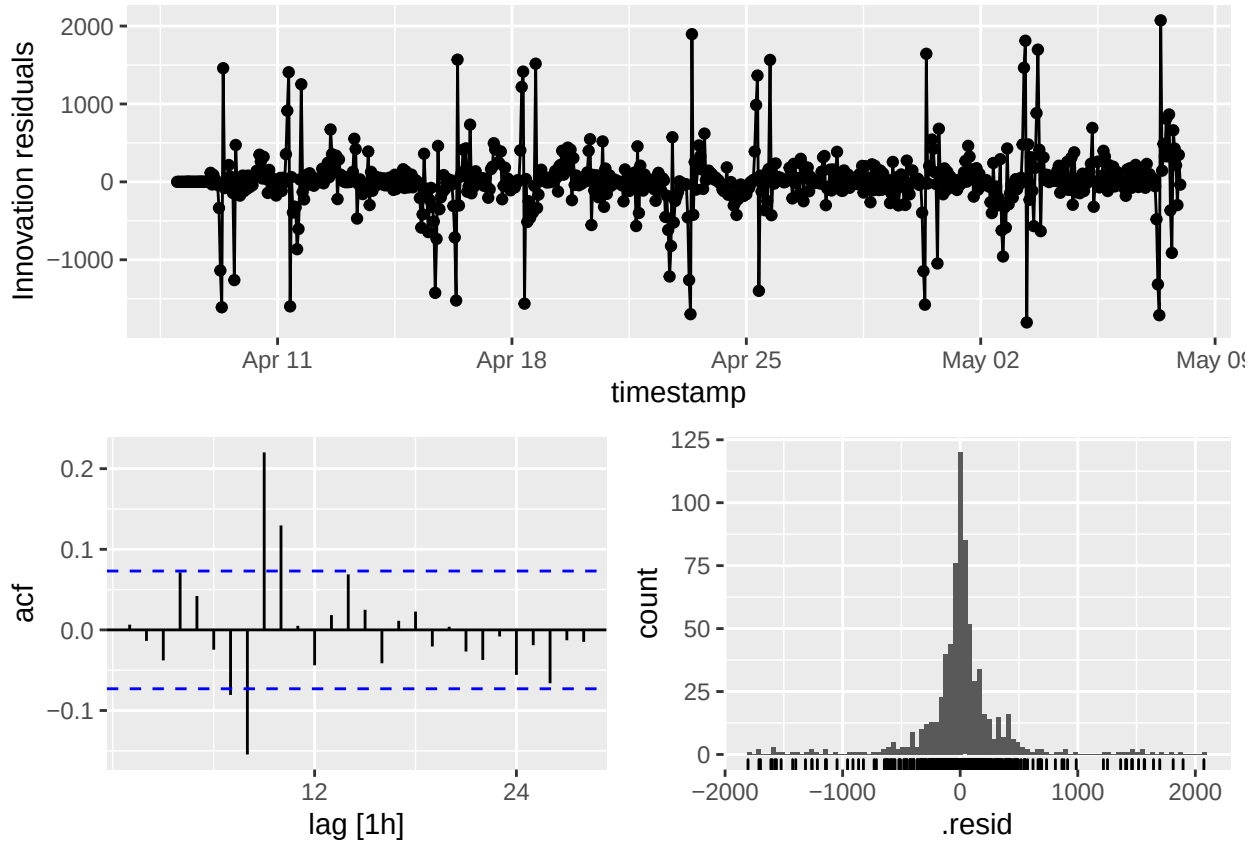
tive to their standard error (especially MA1).

The coefficient for the seasonal AR1 component is also relatively small, while the seasonal AR2 component has a larger coefficient both in absolute terms and relative to its standard error.

Here the seasonal period is 24 as was suggested by the ACF and sequence plots.

▶ Check the residuals of the model using the function gg_tsresiduals(). What is your conclusion?

```
gg_tsresiduals(fit_cnt)
```



The histogram suggests that the residuals follow a normal distribution with a mean of 0.

The residuals on the sequence plot appear to be fairly stable although there are occasional large spikes suggesting some areas might have higher variance than others. It is not entirely clear whether the residuals are stationary, so we can test the residuals for stationarity.

```
adf.test(residuals(fit_cnt)$.resid)
```

```
## Warning in adf.test(residuals(fit_cnt)$.resid): p-value smaller than printed p-
## value

##
```

```
##  Augmented Dickey-Fuller Test
##
## data:  residuals(fit_cnt)$.resid
## Dickey-Fuller = -8.4221, Lag order = 8, p-value = 0.01
## alternative hypothesis: stationary
```

```
kpss.test(residuals(fit_cnt)$.resid)
```

```
## Warning in kpss.test(residuals(fit_cnt)$.resid): p-value greater than printed p-
## value
```

```
##
##  KPSS Test for Level Stationarity
##
## data:  residuals(fit_cnt)$.resid
## KPSS Level = 0.16858, Truncation lag parameter = 6, p-value = 0.1
```

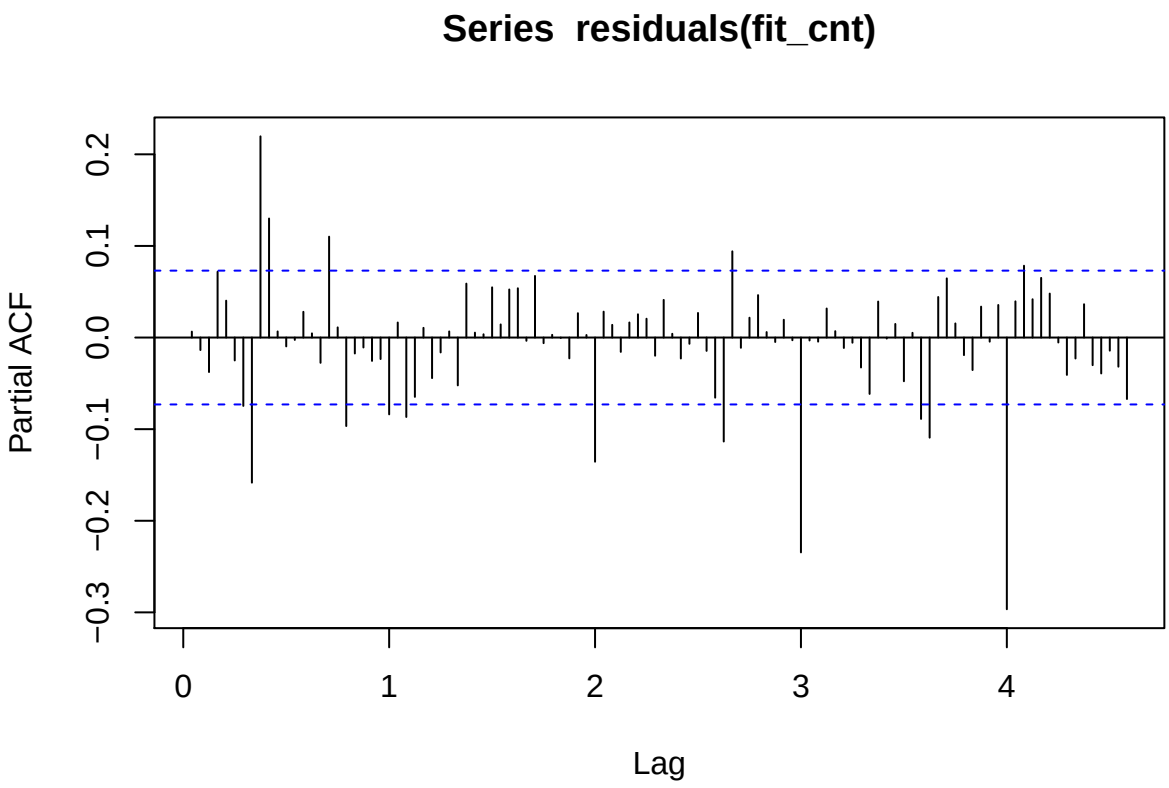
Both the augmented Dickey-Fuller test and the KPSS test show that the residuals are stationary.

While the residuals are stationary, the do not appear to be white noise.

The ACF plot shows that the autocorrelations between the residuals at lags 8, 9 and 10 are significantly larger than 0. The autocorrelations at lag 7 is also significantly larger than 0, however, it sits much closer to the significance threshold represented by the blue dotten lines. Thus, the ACF plot of the residuals suggest that there is still structure in the data that was not accounted for by the ARIMA(2,0,2)(2,1,0)[24] and which can be used to improve predictions.

Let us also check the PACF plot of the residuals.

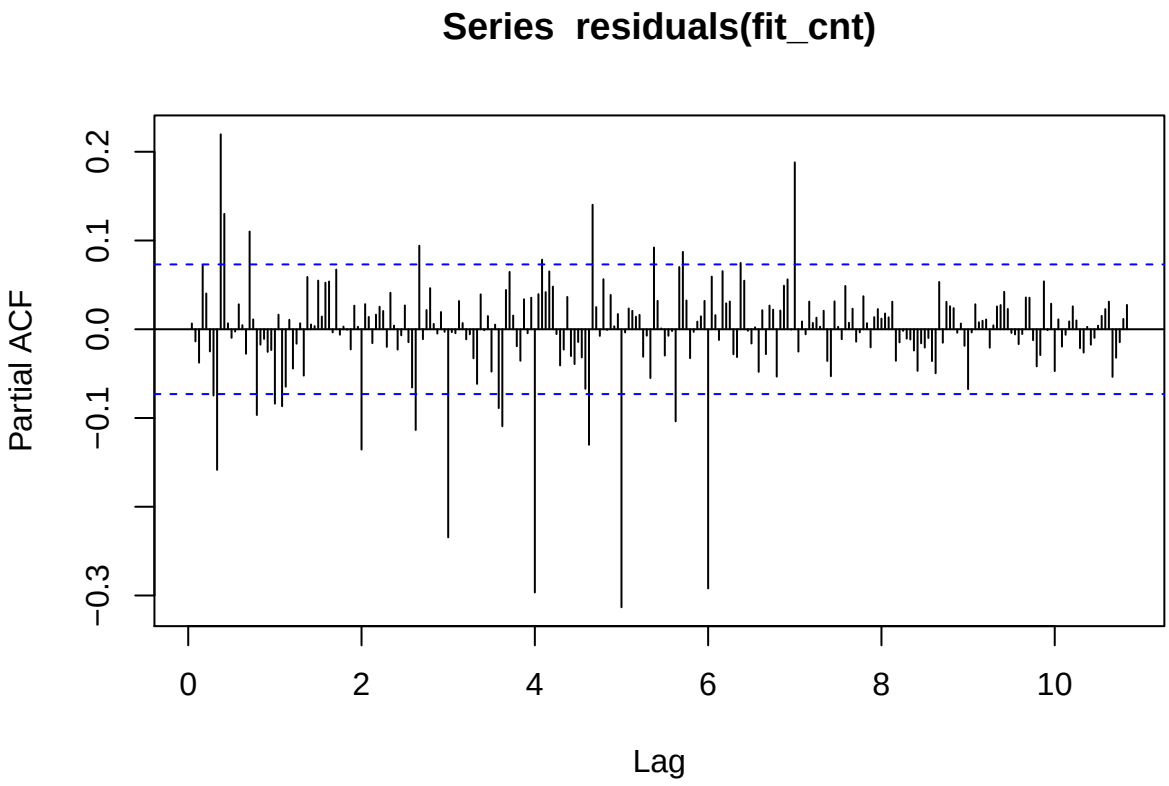
```
pacf(residuals(fit_cnt), lag.max = 110)
```



The scale of the X axis appears to be in multiples of the seasonal period (24). The PACF suggests that that there are seasonal AR3 and AR4 component as there are highly significant spikes at lags 72 and 96. Moreover, the seasonal AR2 component present in the model does not fully account for the autocorrelation at lag 48.

Let us try to extend the PACF graph to see where this pattern stops.

```
pacf(residuals(fit_cnt), lag.max = 260)
```

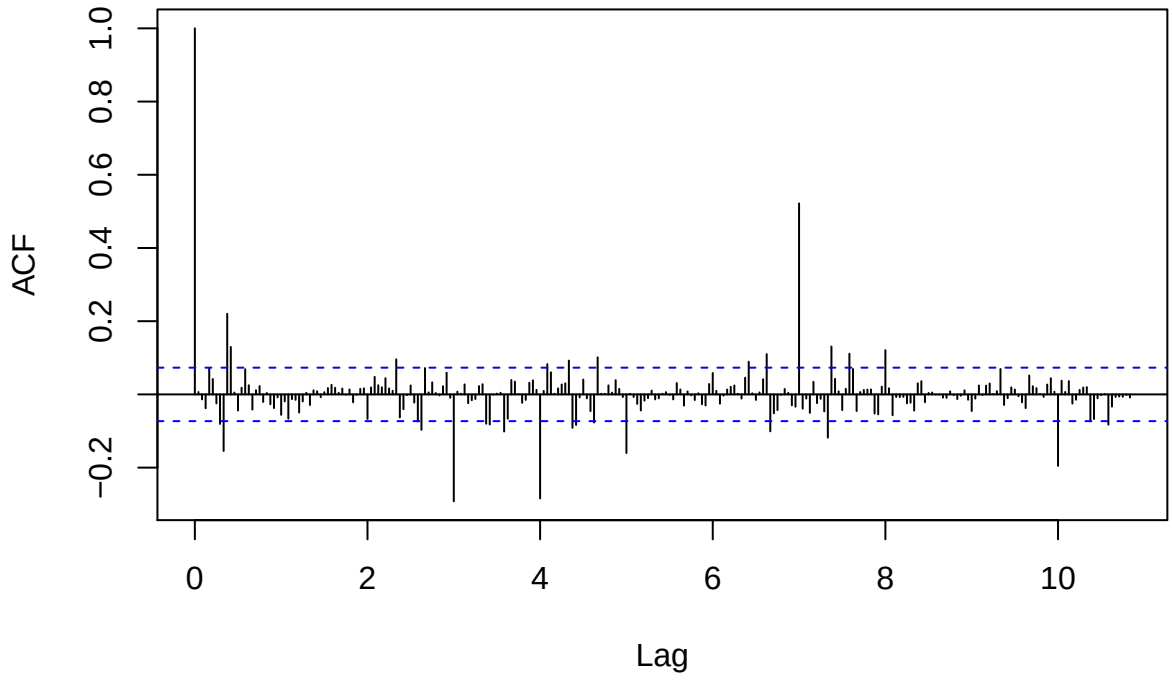


It seems that there are seasonal autocorrelations up to 7th seasonal lag, after which the seasonal spikes stop. This pattern on the PACF plot suggests that there might be a seasonal AR7 component to the data.

It is likely not a coincidence that the seasonal autocorrelations stop at the 7th seasonal lag. As the seasonal period is 24 (daily), the 7th seasonal lag represents the same day from last week. Hence showing evidence of the weekly pattern in the data.

Let us also extend the ACF to the same number of lags.

```
acf(residuals(fit_cnt), lag.max = 260)
```



Here we also see a large spike at lag 168 (the lag at the 7th multiple of the seasonal period).

The pattern in the PACF suggests that there might be a seasonal AR7 component to the time series. Moreover, the ACF suggests that there might be a non-seasonal MA10 component to the data as the autocorrelations were significantly different from 0 at lags 8, 9 and 10. Hence, perhaps we can try to improve our ARIMA model by setting q to 10 and P to 7 - namely, a SARIMA(2,0,10)(7,1,0)[24] model.

Unfortunately, trying to manually fit such a model to the data with the ARIMA() function produces various errors. Various other models with manually specified parameters were run in an attempt to achieve the AICc and reduce the residuals to white noise. For the sake of cleanliness, the code is not and output (often various errors) is not included in this notebook. The model with the best AICc that was successfully fitted is an SARIMA(2,0,2)(4,1,0)[24] model which is shown below:

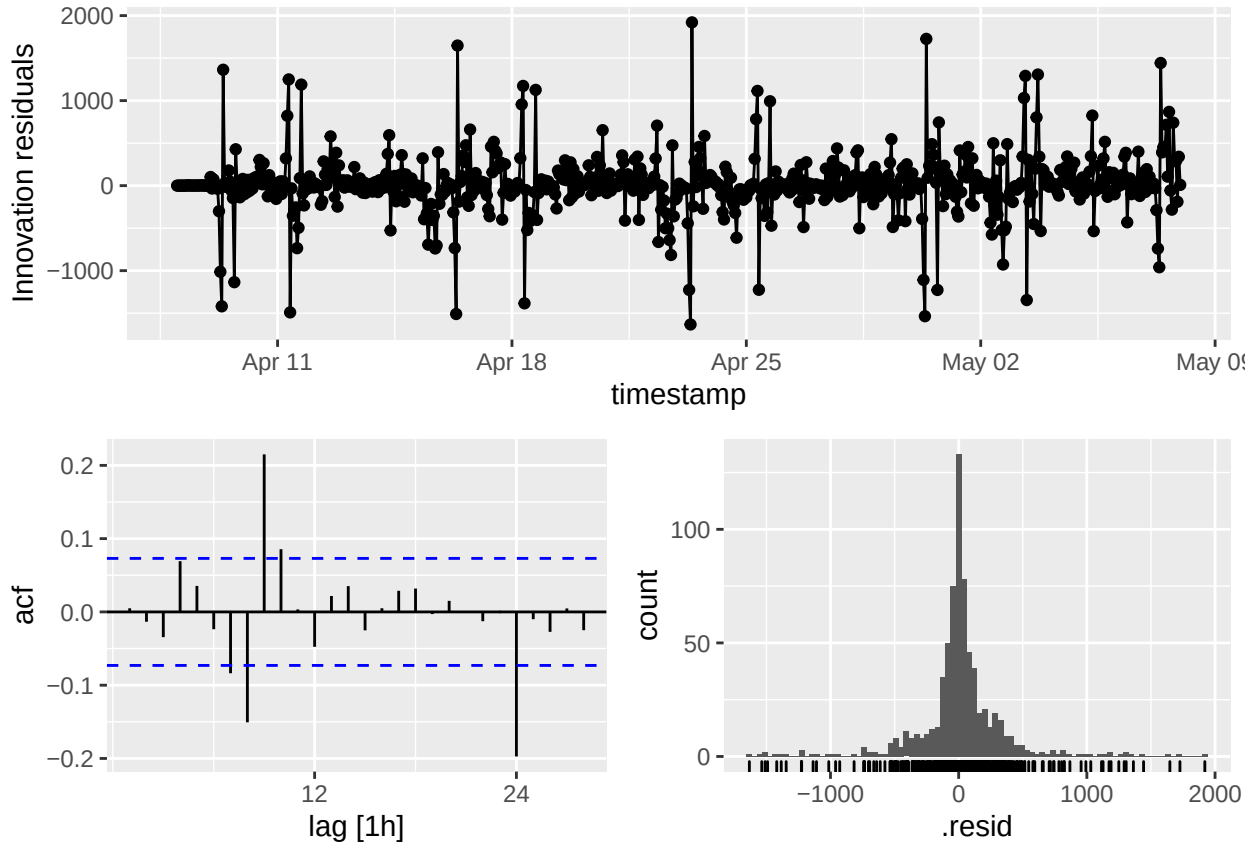
```
fit_202410 = bike_rentals %>% model(ARIMA(cnt ~ 0 + pdq(2,0,2) + PDQ(4,1,0)))
report(fit_202410)
```

```
## Series: cnt
## Model: ARIMA(2,0,2)(4,1,0)[24]
##
## Coefficients:
```

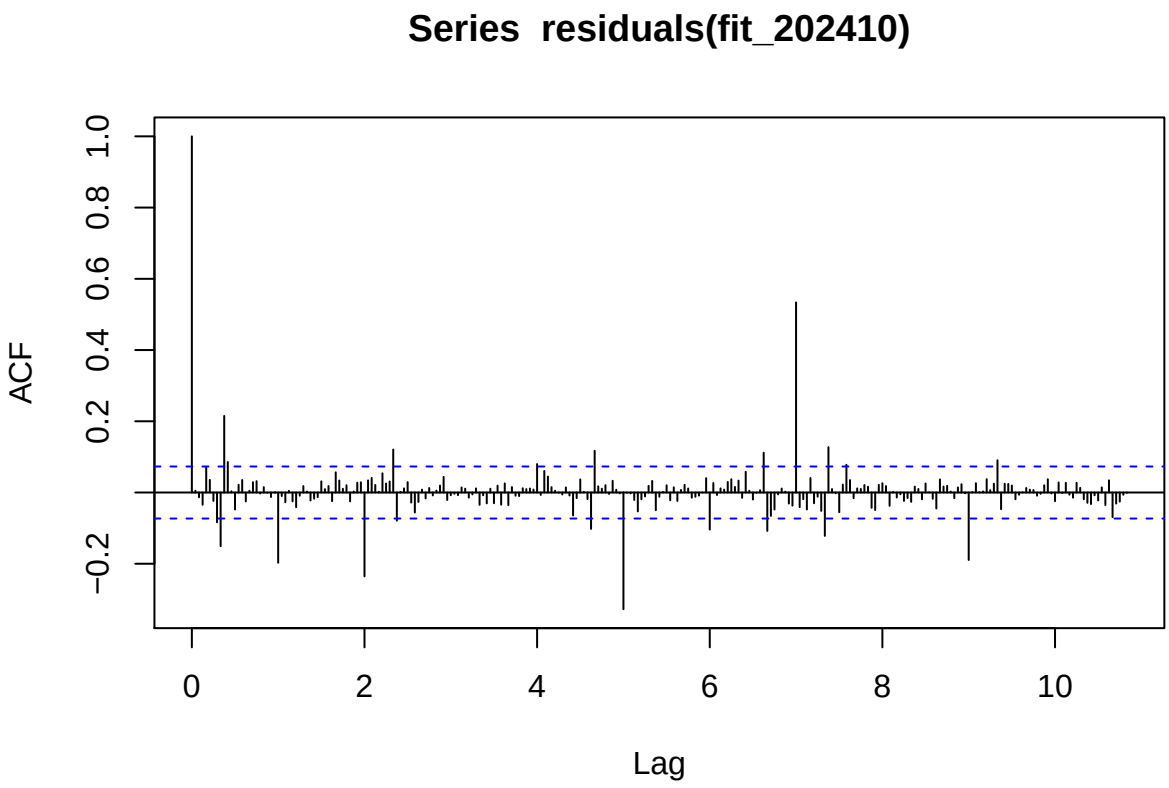
```
##          ar1      ar2      ma1      ma2     sar1     sar2     sar3     sar4
##       1.3091  -0.5097  -0.1610  -0.2533  -0.1404  -0.3193  -0.2880  -0.3766
## s.e.  0.1181   0.0704   0.1243   0.0776   0.0380   0.0367   0.0369   0.0381
##
## sigma^2 estimated as 127703:  log likelihood=-5087.42
## AIC=10192.85   AICc=10193.11   BIC=10233.76
```

As we can see, this is an improvement over the model selected automatically by the ARIMA() function which had an AICc of 10319.21. This, model is also has a lower BIC.

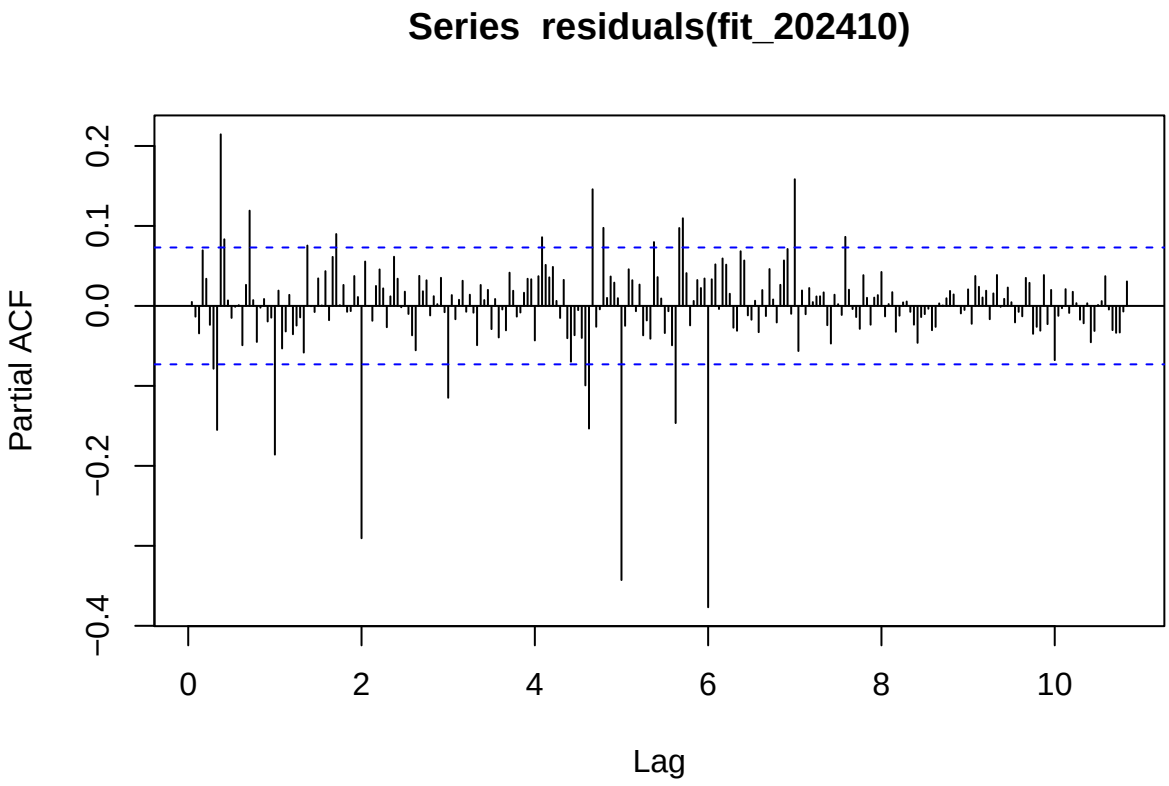
```
gg_tsresiduals(fit_202410)
```




```
acf(residuals(fit_202410), lag.max = 260)
```




```
pacf(residuals(fit_202410), lag.max = 260)
```

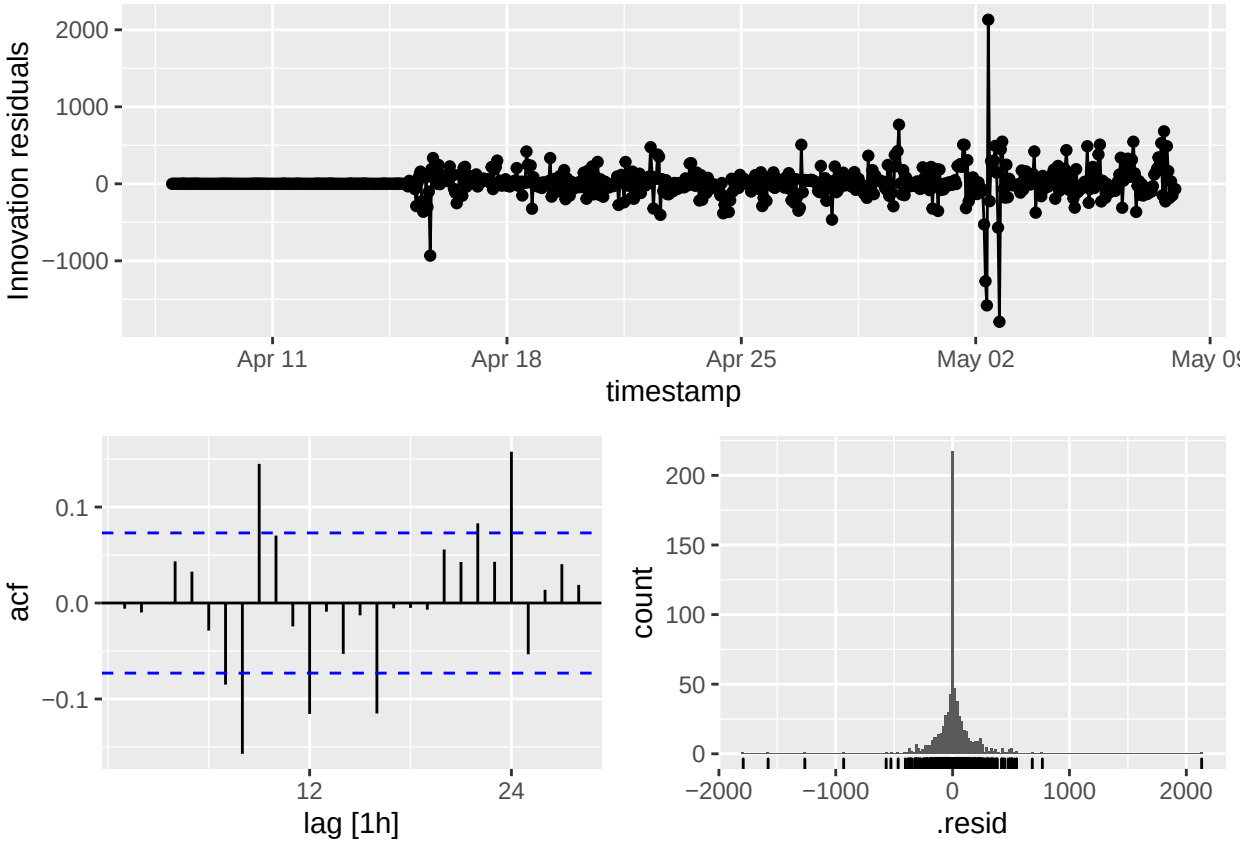


However, the residuals are still not white noise. Another thing we can try is to manually change the seasonal period to 168 (a weekly season).

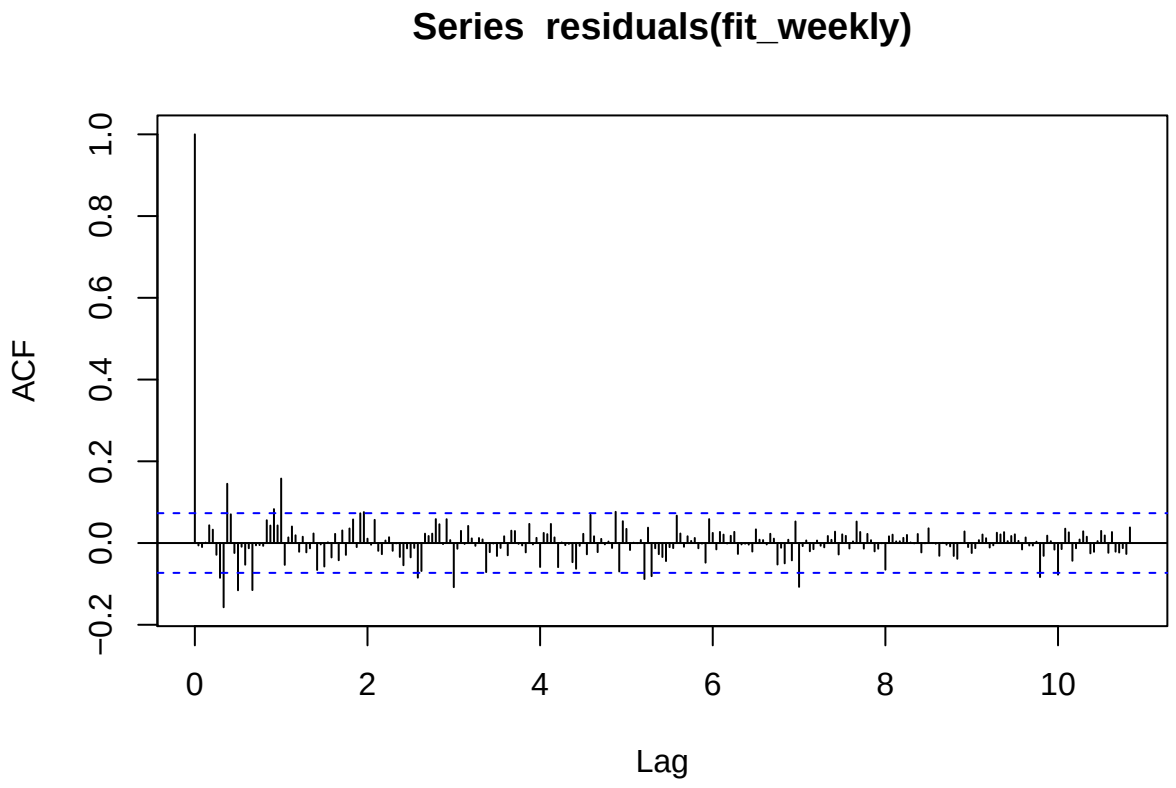
```
fit_weekly = bike_rentals %>% model(ARIMA(cnt ~ 0 + pdq() + PDQ(period=168)))
report(fit_weekly)
```

```
## Series: cnt
## Model: ARIMA(1,1,3)(0,1,0)[168]
##
## Coefficients:
##          ar1      ma1      ma2      ma3
##       0.5823  -0.3291  -0.4090  -0.2065
## s.e.  0.0629   0.0719   0.0393   0.0602
##
## sigma^2 estimated as 51694:  log likelihood=-3774.45
## AIC=7558.9   AICc=7559.01   BIC=7580.46
```

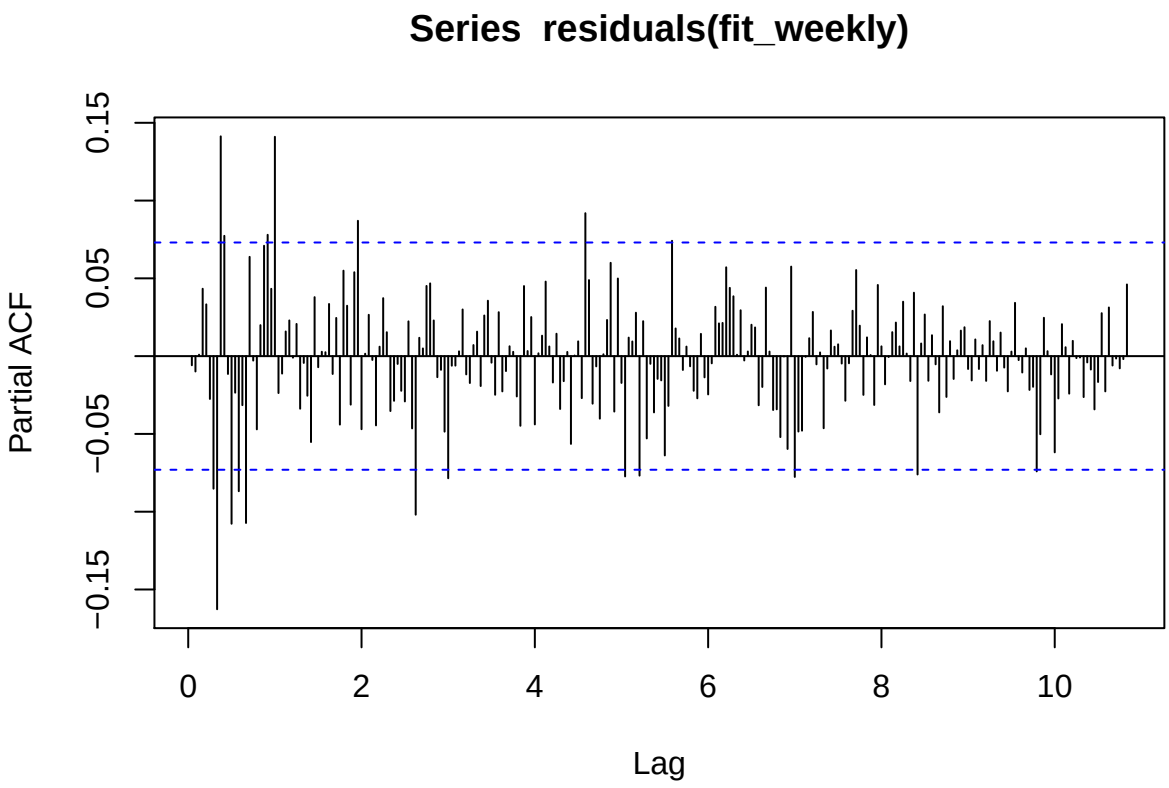
```
gg_tsresiduals(fit_weekly)
```




```
acf(residuals(fit_weekly), lag.max = 260)
```




```
pacf(residuals(fit_weekly), lag.max = 260)
```



The residuals seem much closer to white noise now! Moreover, the BICs and AICs are much lower. However, I am not sure if the BICs and AICs for models with different seasonal periods are comparable as I think the number of observations used to calculate these metrics would be different (n=720-24 in first case and n=720-168 in the second case).

A more adequate approach would be to compare their performance on a test set or using cross validation. As we have used the whole dataset for training, an alternative (although less reliable due to the risk of overfitting) approach would be to just look at the mean absolute error on the training set.

```
mean(abs(residuals(fit_202410)$.resid)[25:720])
```

```
## [1] 205.6894
```

```
mean(abs(residuals(fit_weekly)$.resid)[169:720])
```

```
## [1] 129.3549
```

We can see that the mean absolute error for the ARIMA(1,1,3)(0,1,0)[168] model is much lower. Because the MAE is lower and the ACF and PACF plots are much more resemblant of white noise - I would prefer the ARIMA(1,1,3)(0,1,0)[168] model for predictions.

2.2. Dynamic regression

▶ Include the predictor in an dynamic regression model (i.e., allow for (S)ARIMA residuals); what is the effect of the predictor?

```
fit_dynamic <- bike_rentals %>% model(ARIMA(cnt ~ t1 + hum + wind_speed + weather_code +
report(fit_dynamic)
```

```
## Series: cnt
## Model: LM w/ ARIMA(2,0,2)(2,0,0)[24] errors
##
## Coefficients:
##          ar1      ar2      ma1      ma2    sar1     sar2       t1      hum
##       1.3429  -0.5469  -0.2783  -0.3452  0.8790  -0.1310  37.3836  -9.9190
## s.e.  0.0626   0.0401   0.0698   0.0541  0.0406   0.0414  12.0542   2.0766
##       wind_speed  weather_code2  weather_code3  weather_code4  weather_code7
##           1.8864         2.6525        -3.8570       -45.7262      -118.0519
## s.e.      2.7480        29.8156        43.4262        80.4936        42.7888
##       weather_code10  is_weekend1  intercept
##            -194.4803      -3.7684  1443.6287
## s.e.        113.0201      56.2255   245.4198
##
## sigma^2 estimated as 136865:  log likelihood=-5283.35
## AIC=10600.7   AICc=10601.58   BIC=10678.55
```

The coefficients for the predictors are: * t1: 37.3836 * hum: -9.9190 * wind_speed: 1.8864 * weather_code2 (scattered clouds): 2.6525 * weather_code3 (broken clouds): -3.8570 * weather_code4 (cloudy): -45.7262 * weather_code7 (rain): -118.0519 * weather_code10 (rain with thunderstorm): -194.4803 * is_weekend1: -3.7684

Hence, the variables with a positive “effect” on bike rentals are temperature, wind_speed and weather_code2, while the predictors with a negative “effect” on bike rentals are humidity, weather_code3, weather_code4, weather_code7, weather_code10 and is_weekend.

The variables with a somewhat large coefficient relative to their standard error are temperature, humidity and weather_code7, while the variables with a low coefficient relative to their standard error are wind_speed, weather_code2, weather_code3, weather_code4, weather_code10 and is_weekend.

Let us also try the same approach as in the previous step and set the seasonal period to 168 in the dynamic regression model.

```
fit_dynamic_weekly <- bike_rentals %>% model(ARIMA(cnt ~ t1 + hum + wind_speed + weather
report(fit_dynamic_weekly)
```

```
## Series: cnt
## Model: LM w/ ARIMA(2,0,4) errors
##
```

```
## Coefficients:
##          ar1      ar2      ma1      ma2     ma3     ma4       t1       hum
##       1.7170  -0.9629  -0.7772  -0.3151  0.2291  0.3231  90.5599  -20.2725
## s.e.  0.0141   0.0139   0.0398   0.0477  0.0603  0.0470  12.7259    2.9312
##       wind_speed  weather_code2  weather_code3  weather_code4  weather_code7
##          22.1740        66.6115        73.0356       -92.3268       -14.2767
## s.e.      4.8604        58.8339        79.5363       169.9501        75.4878
##       weather_code10  is_weekend1  intercept
##              -7.2773     -35.6927  1210.2871
## s.e.        229.1345      81.3164   310.2223
##
## sigma^2 estimated as 309726:  log likelihood=-5566.23
## AIC=11166.45   AICc=11167.33   BIC=11244.3
```

When trying to set the seasonal period to 168, the function fails to estimate a SARIMA model for the residuals and instead selects a regular ARIMA model without a seasonal component.

It can be seen that the coefficients for most predictors are much larger than those in the former SARIMA model, both in absolute terms and relative to their standard error. This is likely because a larger part of the variance is attributed to the predictors rather than the ARIMA component.

```
mean(abs(residuals(fit_dynamic)$.resid)[25:720])
```

```
## [1] 229.2997
```

```
mean(abs(residuals(fit_dynamic_weekly)$.resid))
```

```
## [1] 362.5422
```

The mean absolute error is also lower for the dynamic regression model with SARIMA(2,0,2)(2,0,0)[24] errors.

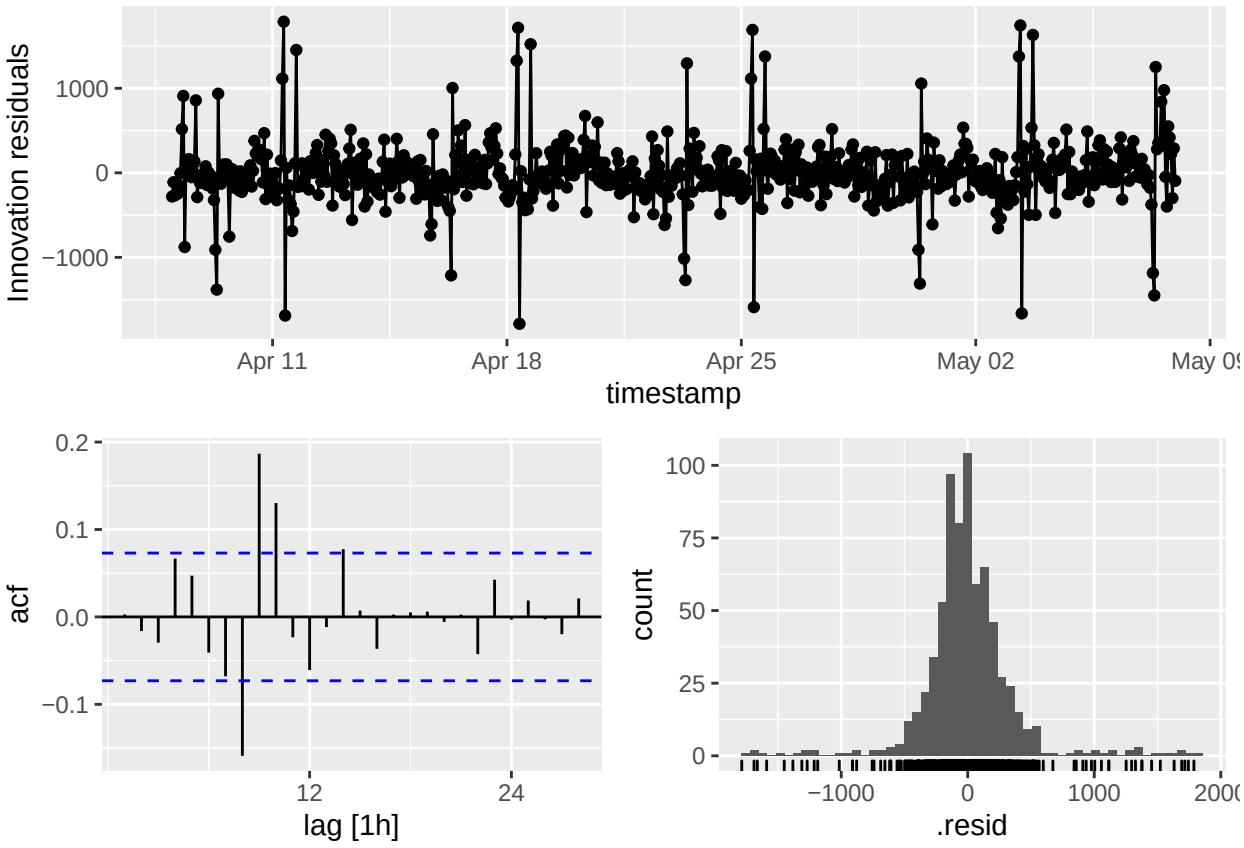
As it has a lower AICc, BIC, and MAE, only the dynamic regression model with SARIMA(2,0,2)(2,0,0)[24] residuals will be considered going forward in the rest of part 2.

▶ What order is the (S)ARIMA model for the residuals?

The SARIMA model for the residuals that was selected when we did not specify the seasonal period was a SARIMA(2,0,2)(2,0,0)[24].

▶ Check the residuals of the model using the function gg_tsresiduals(). What is your conclusion?

```
gg_tsresiduals(fit_dynamic)
```



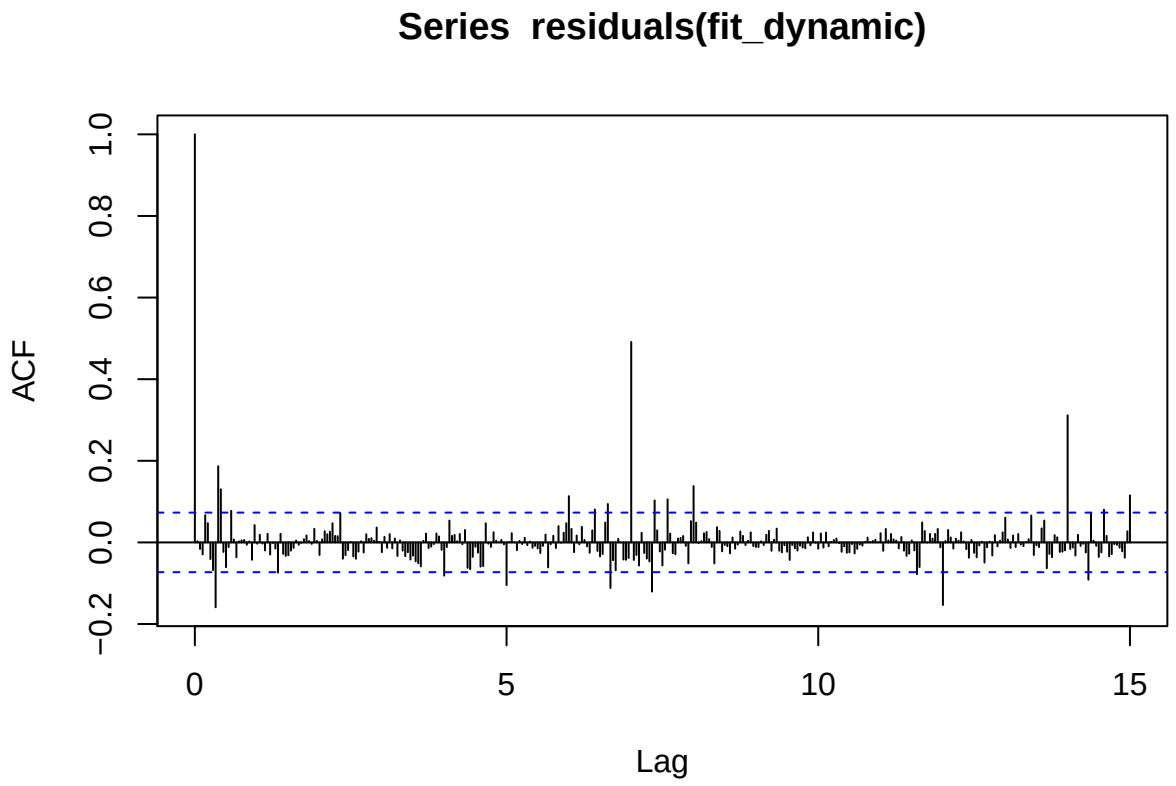
The sequence plot of the residuals shows observations with a stable mean and variance across time and no visible seasonality. Hence the sequence plot suggests that the data are stationary.

The histogram shown is less symmetric than what would be expected from a normal distribution.

In the ACF plot we can see significant autocorrelation at lags 8, 9 and 10. This suggests that the data are not white noise and there is still information that can be used to improve the model.

Let us also at farther lags.

```
acf(residuals(fit_dynamic), lag.max = 360)
```



Again the scale of the X axis is in multiples of the seasonal period (24). We see highly significant autocorrelations at seasonal lags 7 and 14. This suggests a weekly seasonality that is not accounted for by the SARIMA(2,0,2)(2,0,0)[24] model for the residuals.

2.3. Forecasts

▶ Choose a forecasting horizon, and indicate why this is a reasonable and interesting horizon to consider.

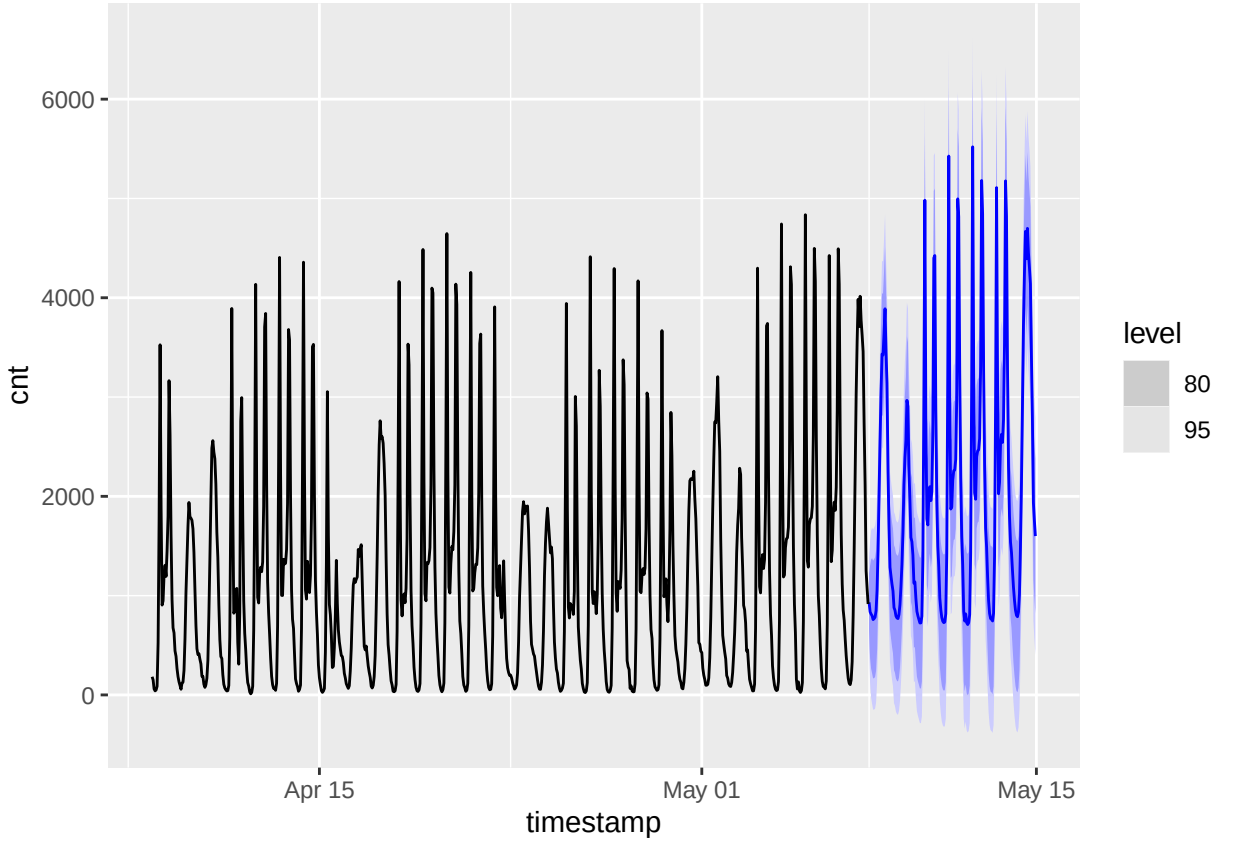
Being able to forecast demand for the week ahead can be a relevant timeframe from a business perspective. Knowing what the demand will be in the next week may be informative as to whether additional bikes must be supplied for a given area (e.g., through relocation of bikes from another area).

However, it is quite a short horizon which limits the forecasts' usefulness in deciding whether additional bikes must be purchased - as procurement decisions cannot be taken so fast and frequently, let alone executed by the supplier.

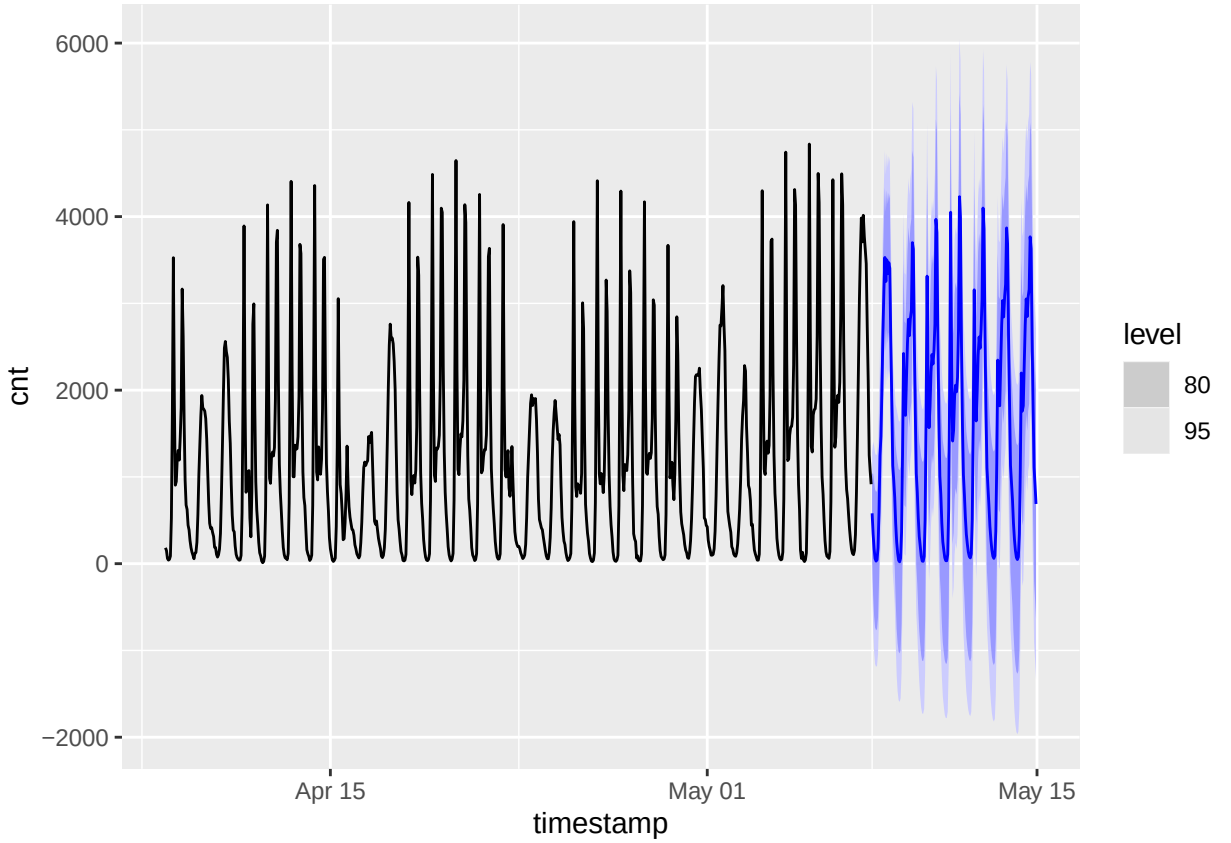
Other forecasting timeframes might be much more relevant and informative from a business perspective, especially monthly, quarterly and yearly forecasts. A more reasonable approach for these types of forecasts may be to aggregate the hourly data to a daily or weekly level and create forecasts from there. In this scenario a much larger dataset than a single month of data would also be required.

▶ Create forecasts based on the model without the predictor and plot these.

```
fit_weekly %>% forecast(h=168) %>% autoplot(bike_rentals)
```




```
fit_202410 %>% forecast(h=168) %>% autoplot(bike_rentals)
```



▶ Create forecasts based on the model with the predictor and plot these.

Creating new data

For the continuous variables I will create new data based on ARIMA forecasts. So let us fit an ARIMA model for each of the continuous variables.

```
fit_t1 = bike_rentals %>% model(ARIMA(t1))
fit_hum = bike_rentals %>% model(ARIMA(hum))
fit_wind_speed = bike_rentals %>% model(ARIMA(wind_speed))
```

For is_weekend I will simply create new timestamps for the next 168 hours and extract the weekday information from the timestamps

```
new_timestampts = seq(
     from=as.POSIXct("2016-5-8 0:00"),
     to=as.POSIXct("2016-5-14 23:00"),
     by="hour"
   )

new_is_weekend = ifelse(weekdays(new_timestampts) %in% c('Saturday', 'Sunday'),1,0)
```

As for the weather_code I will simply set all values to 1 (clear weather). This is known as

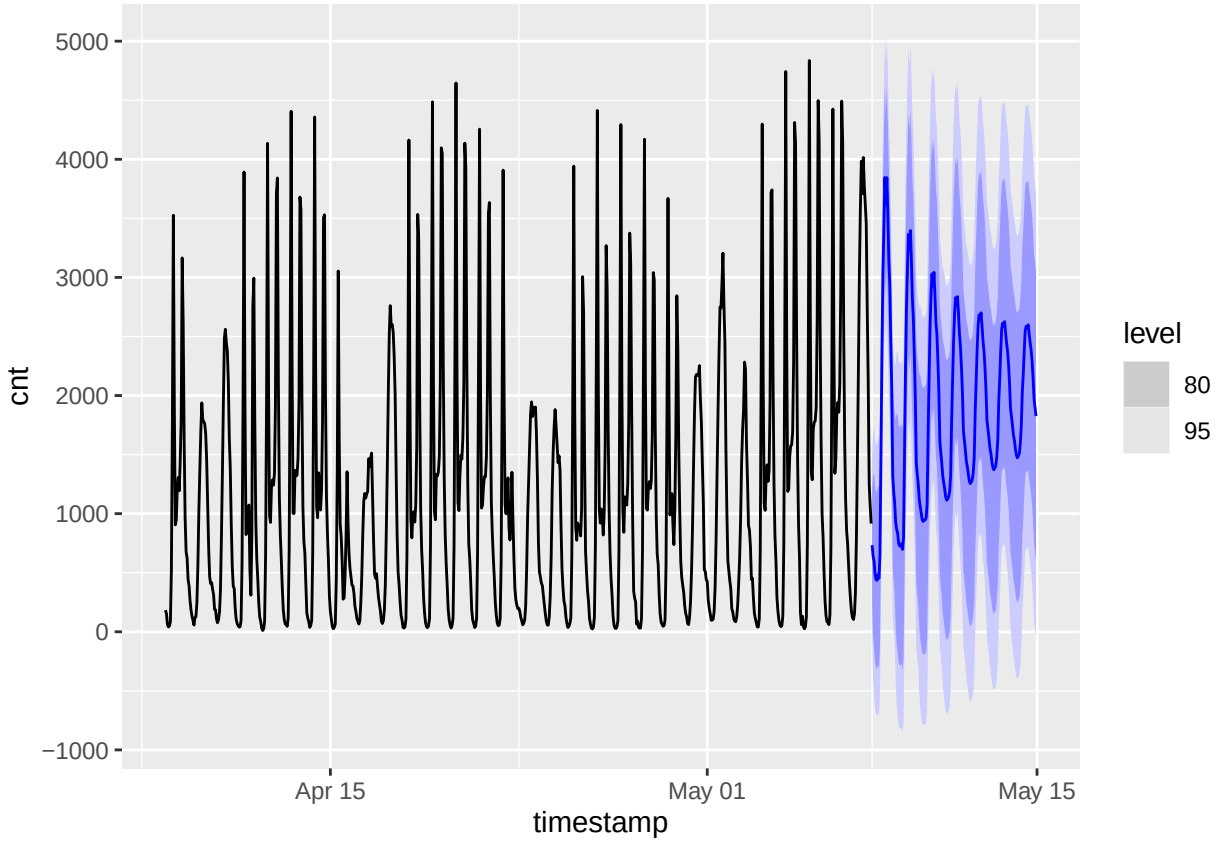
“scenario forecasting” - we are computing forecasts under the scenario that the weather will be clear the entire week.

So let us create the new data for our predictors.

```
X_future <- new_data(bike_rentals, 168) %>%
  mutate(t1 = (fit_t1 %>% forecast(h=168))$.mean,
         hum = (fit_hum %>% forecast(h=168))$.mean,
         wind_speed = (fit_wind_speed %>% forecast(h=168))$.mean,
         is_weekend = as.factor(new_is_weekend),
         weather_code = factor(1, levels=c(1, 2, 3, 4, 7, 10))
         )
```

Forecasting bike rentals and plotting forecasts

```
forecast(fit_dynamic, new_data = X_future) %>% autoplot(bike_rentals)
```



▶ Compare the plots of both forecasts (visually), and discuss how they are similar and/or different.

What is interesting about the dynamic regression model (fit_dynamic) is that the forecasted values exhibit decreasing variance and seem to be converging towards a stable constant. These appear to be fairly flawed forecasts considering that the original data does not ex-

hibit this decreasing variance. One possible reason for this is that the forecasted values for the predictors themselves converge towards a stable constant, in turn affecting the predictions for bike rentals. The prediction intervals are much narrower than they should be as they do not take into account the uncertainty in the forecasted values for the predictors.

The forecasts from the ARIMA(1,1,3)(0,1,0)[168] model (fit_weekly) do not exhibit this decreasing variance, however, there is a sudden shift in the mean and the troughs are now near the 800 bike rentals as opposed to 0. Perhaps this could be due to some irregularities with last few observed values.

The SARIMA(2,0,2)(4,1,0)[24] (fit_202410) model, on the other hand, does not exhibit either of these unwanted behaviors and appears to capture the seasonal pattern in the data rather well.

3. Causal Modeling

▶ Formulate a causal research question(s) involving the time series variable(s) you have measured.

Does temperature cause bike rentals?

▶ Which method we learned about in class (Granger causal approaches, interrupted time series, synthetic controls) is most appropriate to answer your research question using the data you have available? Why?

As my cause variable is continuous, Granger causal analysis would be most suitable to answer my causal research question.

Interrupted time series and synthetic control are suitable for situations in which we observe an intervention at a given point in time and can split the data into a pre- and post-intervention period. My data is not an example of such a situation.

3.2 Analysis

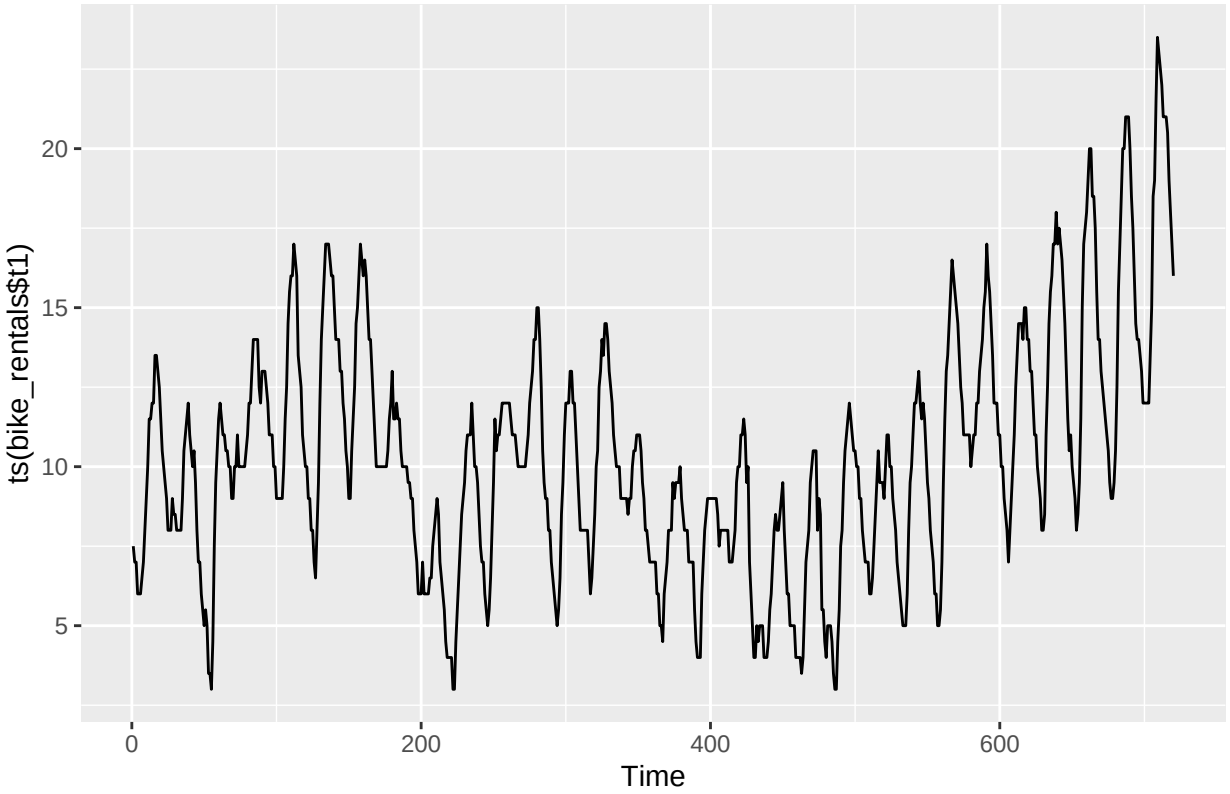
Depending on the choice you made above, follow the questions outlined in 3.2a, 3.2b or 3.2c. If you chose a Granger causal analysis, it is sufficient to assess Granger causality in one direction only: you may evaluate a reciprocal causal relationship, but then answer each question below for both models.

3.2a Granger Causal analysis

▶ Visualize your putative cause variable(s) X and outcome variables Y.

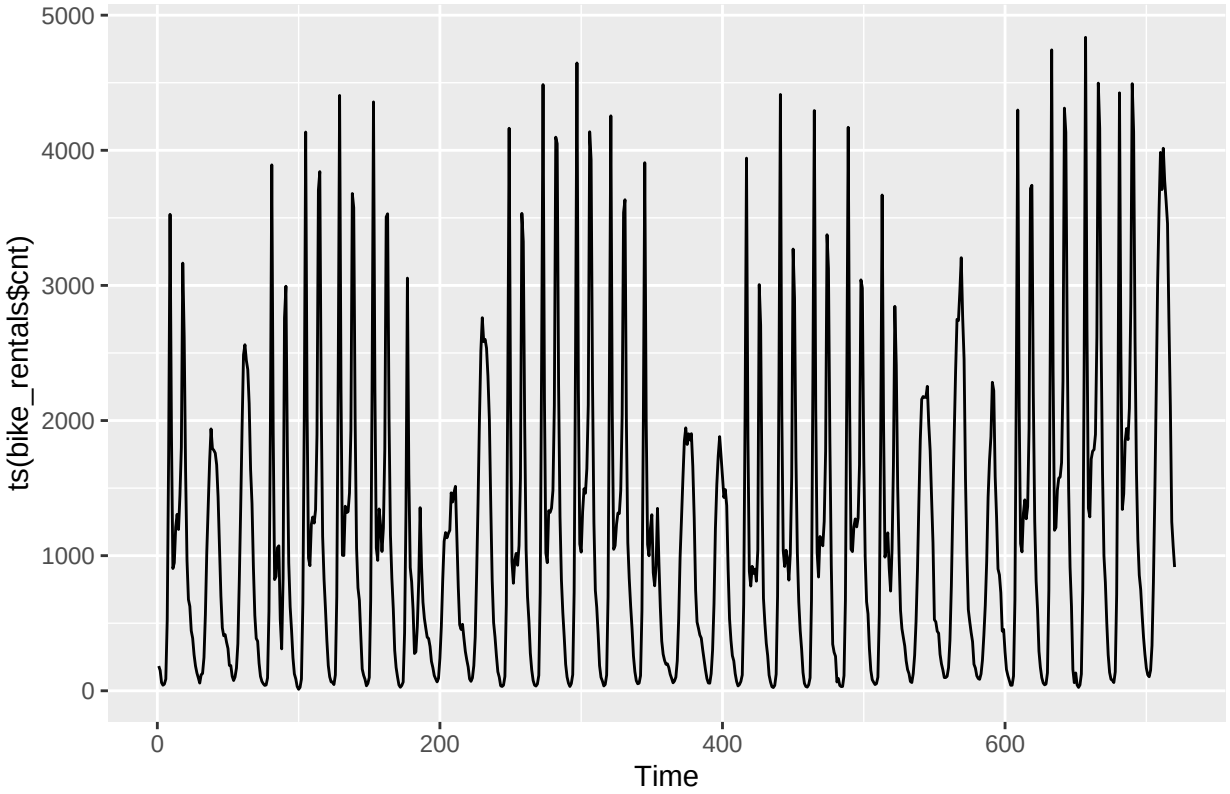
- Plotting temperature

```
autoplot(ts(bike_rentals$t1))
```



* Plotting bike rentals

```
autoplot(ts(bike_rentals$cnt))
```



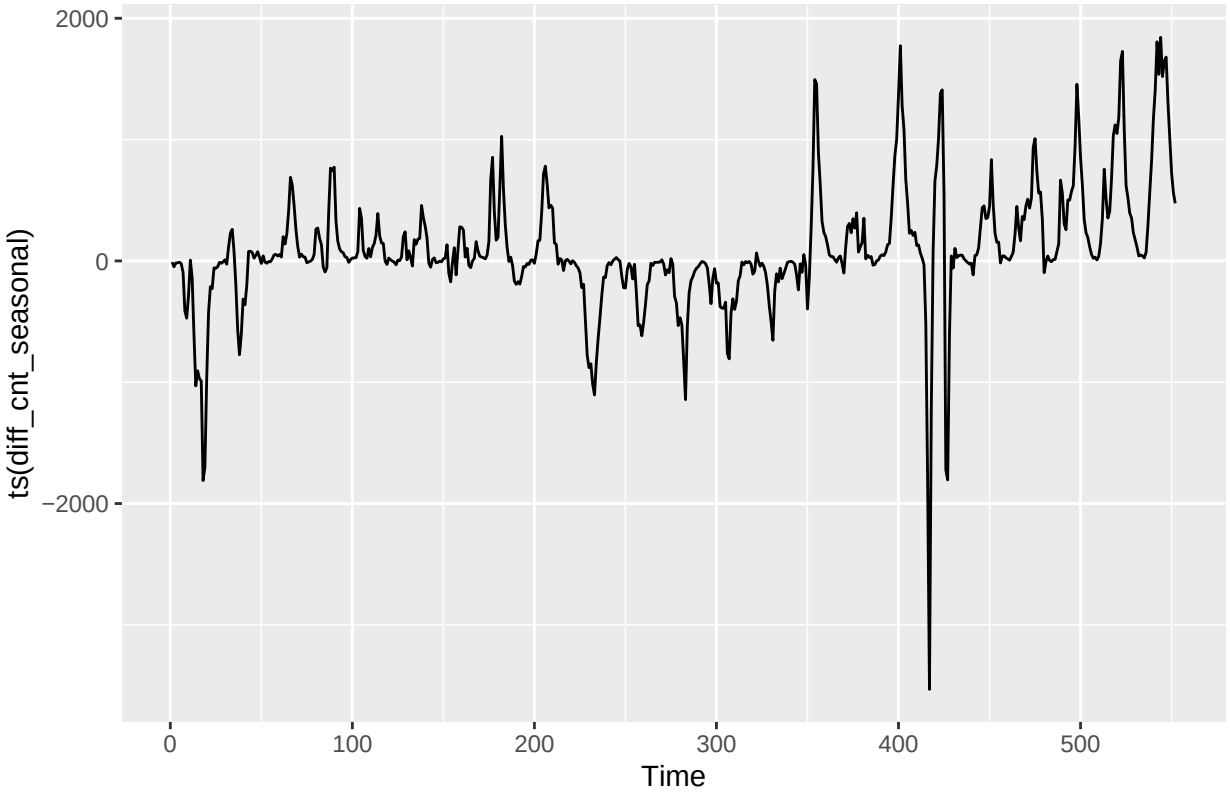
▶ Train an appropriate ARIMA model on your outcome variable(s) Y, ignoring the putative cause variable(s) (X) but including, if appropriate, any additional covariates. If using the same model as fit in part 2, briefly describe that model again here.

In contrast to my models for part 2, I will train a dynamic regression model with ARIMA residuals on all predictors excluding my cause variable of interest - temperature. Moreover, I will include lags of the continuous predictors in an attempt to better control for any possible confounding.

CCF with covariates To choose the appropriate lag level for the continuous predictors (humidity and wind speed), I will examine their CCF with bike rentals (cnt).

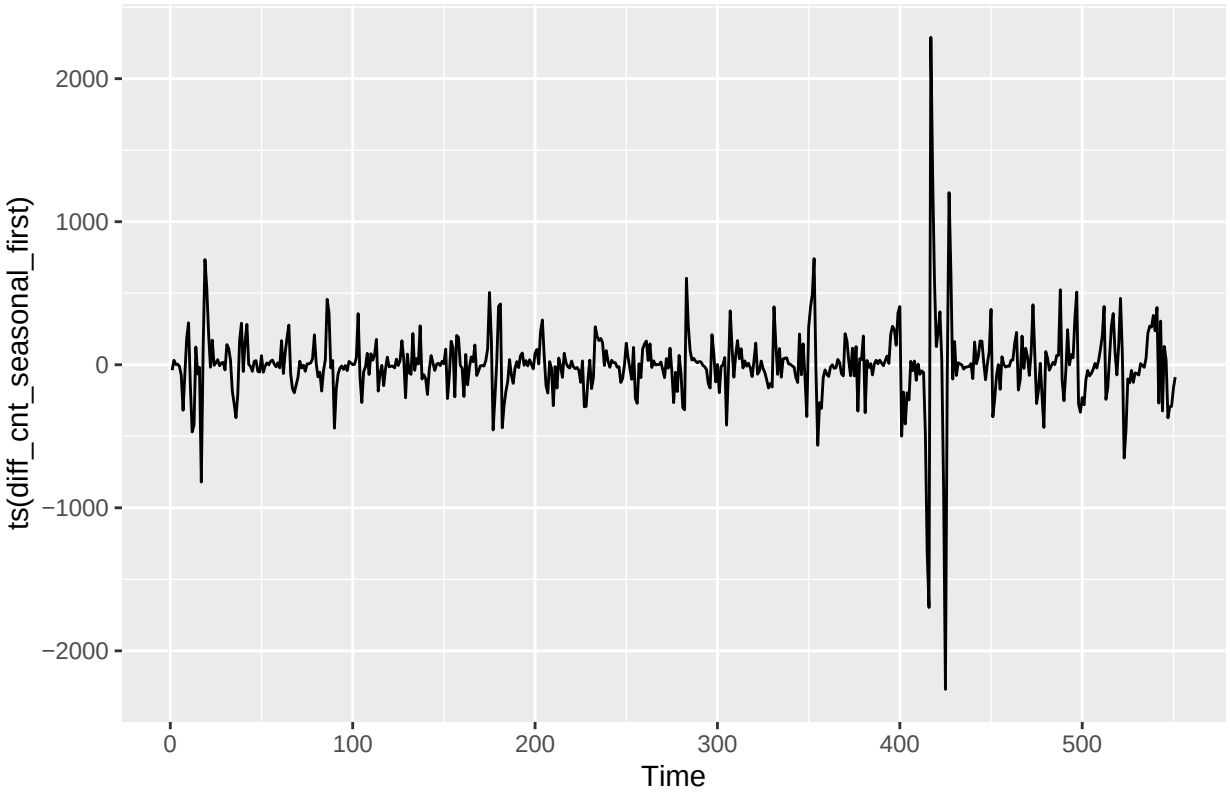
Before computing the CCF, our time series must be stationary. As I know that there is a seasonal component with a period of 168 to the bike rentals time series, I will first perform a seasonal differencing and examine the resulting time series

```
diff_cnt_seasonal = diff(bike_rentals$cnt, lag=168)
autoplot(ts(diff_cnt_seasonal))
```



We can see that the time series is still not stationary, so we can try to perform a first differencing on top of the seasonal differencing.

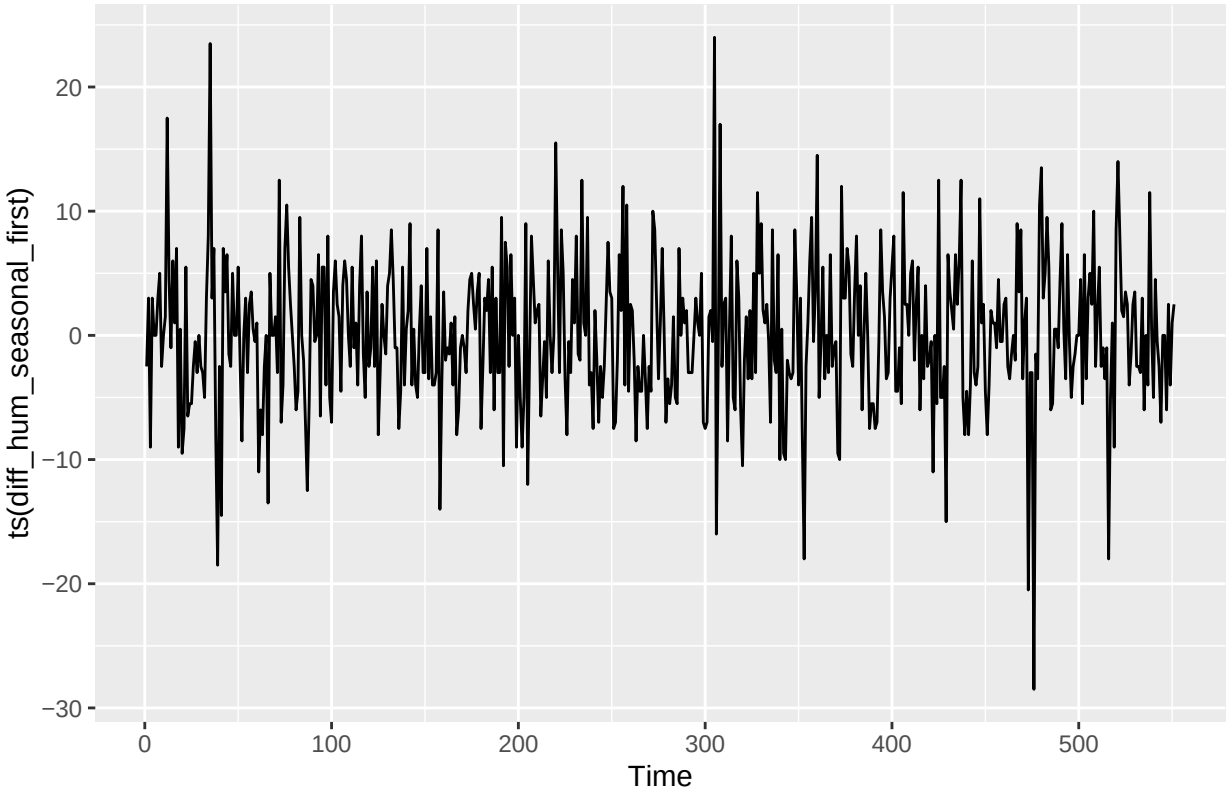
```
diff_cnt_seasonal_first = diff(diff_cnt_seasonal, lag=1)
autoplot(ts(diff_cnt_seasonal_first))
```



The time series looks stationary now! We can check whether the seasonal + first differencing results in stationary time series for the covariates as well.

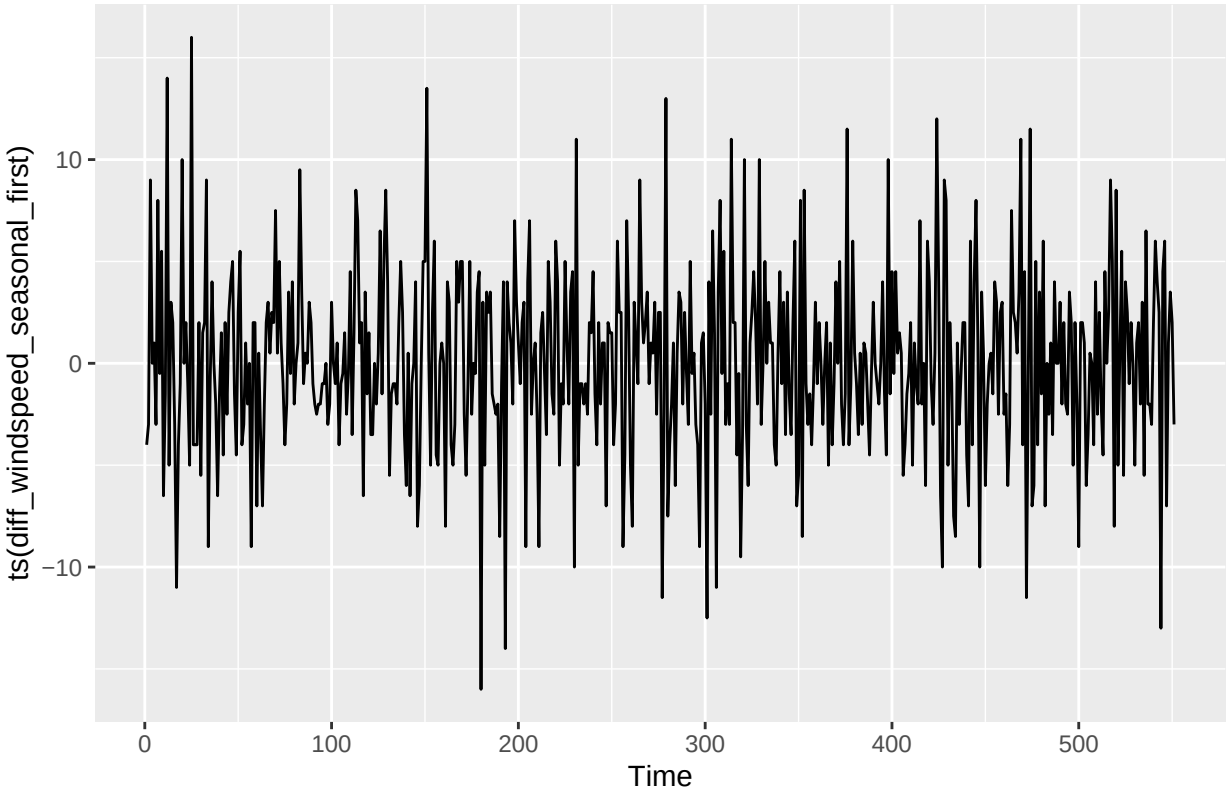
- Humidity

```
diff_hum_seasonal = diff(bike_rentals$hum, lag=168)
diff_hum_seasonal_first = diff(diff_hum_seasonal, lag=1)

autoplot(ts(diff_hum_seasonal_first))
```



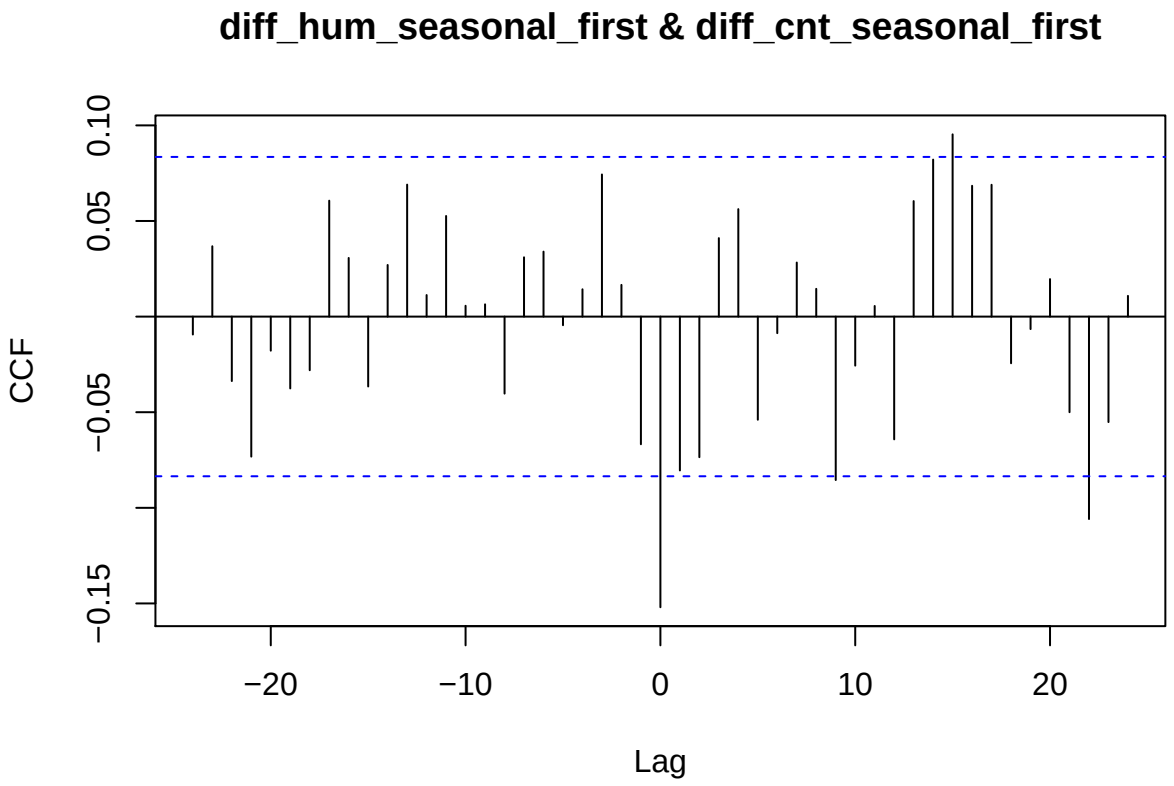
- Wind Speed

```
diff_windspeed_seasonal = diff(bike_rentals$wind_speed, lag=168)
diff_windspeed_seasonal_first = diff(diff_windspeed_seasonal, lag=1)

autoplot(ts(diff_windspeed_seasonal_first))
```



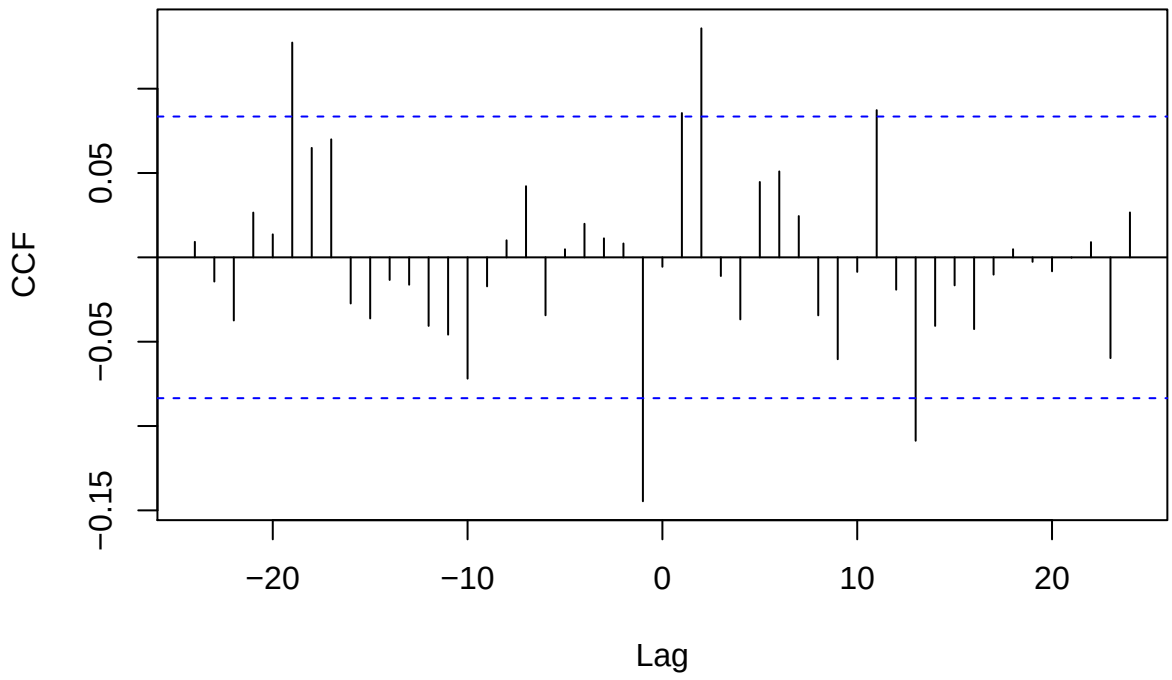
Yes! We can proceed with examining the CCFs.

```
ccf(diff_hum_seasonal_first, diff_cnt_seasonal_first, ylab = "CCF")
```



We do not see any significant lags to the left of 0 on the X axis. This means that no lags of humidity are significantly correlated with bike rentals. Hence, we will only consider the current value of humidity in our model.

```
ccf(diff_windspeed_seasonal_first, diff_cnt_seasonal_first, ylab = "CCF")
```



We can see that the first lag of wind speed is significantly correlated with bike rentals. This is also the case for lag 19 but that is likely a fluke, as it is not reasonable to expect for the wind of 19 hours ago to influence current bike rentals. Thus, only the first lag of wind_speed will be included in our model.

```
fit_covariates <- bike_rentals %>% model(ARIMA(cnt ~ hum + wind_speed + lag(wind_speed)
report(fit_covariates)
```

Training dynamic regression model

```
## Series: cnt
## Model: LM w/ ARIMA(4,0,0)(2,0,0)[24] errors
##
## Coefficients:
##          ar1      ar2     ar3      ar4    sar1     sar2       hum  wind_speed
##       1.1036  -0.6340  0.2472  -0.1328  0.8911  -0.1328  -11.3202      2.8687
## s.e.  0.0389   0.0554  0.0552   0.0375  0.0402   0.0413    2.0837      2.8527
##       lag(wind_speed)  weather_code2  weather_code3  weather_code4
##               -3.2047        17.2983        19.2889       -18.8563
## s.e.           2.8437        29.1866        42.2552        79.0099
##       weather_code7  weather_code10  is_weekend1  intercept
```

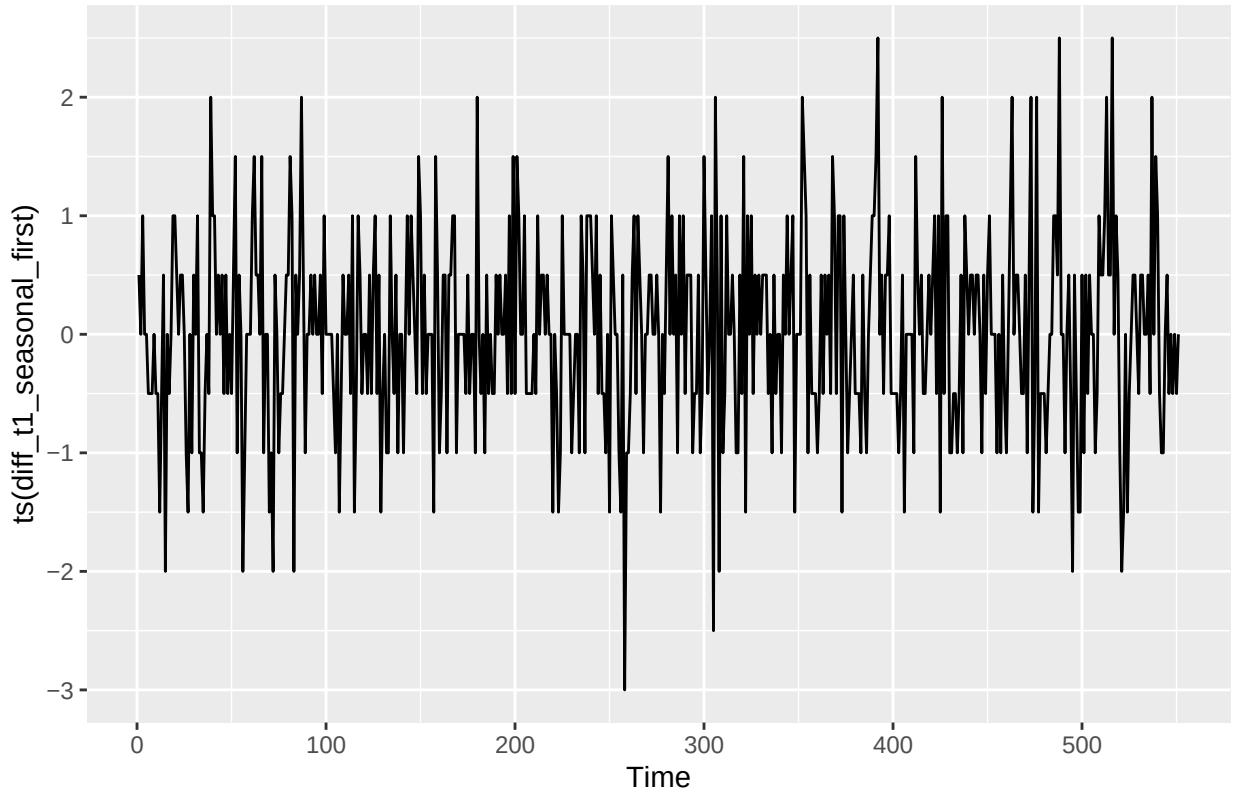
```
##             -97.1008       -183.4657     -45.7010  1982.5495
## s.e.         42.0405        110.6300      62.1695   198.1811
##
## sigma^2 estimated as 138510:  log likelihood=-5281.29
## AIC=10596.59   AICc=10597.46   BIC=10674.44
```

The model produced is a dynamic regression model with SARIMA(4,0,0)(2,0,0)[24] residuals. The model's AICc and BIC scores are 10597.46 and 10674.44 respectively.

▶ Justify what range of lags to consider for the lagged predictor(s). Use the CCF, but you may also justify this based on domain knowledge or substantive theory.

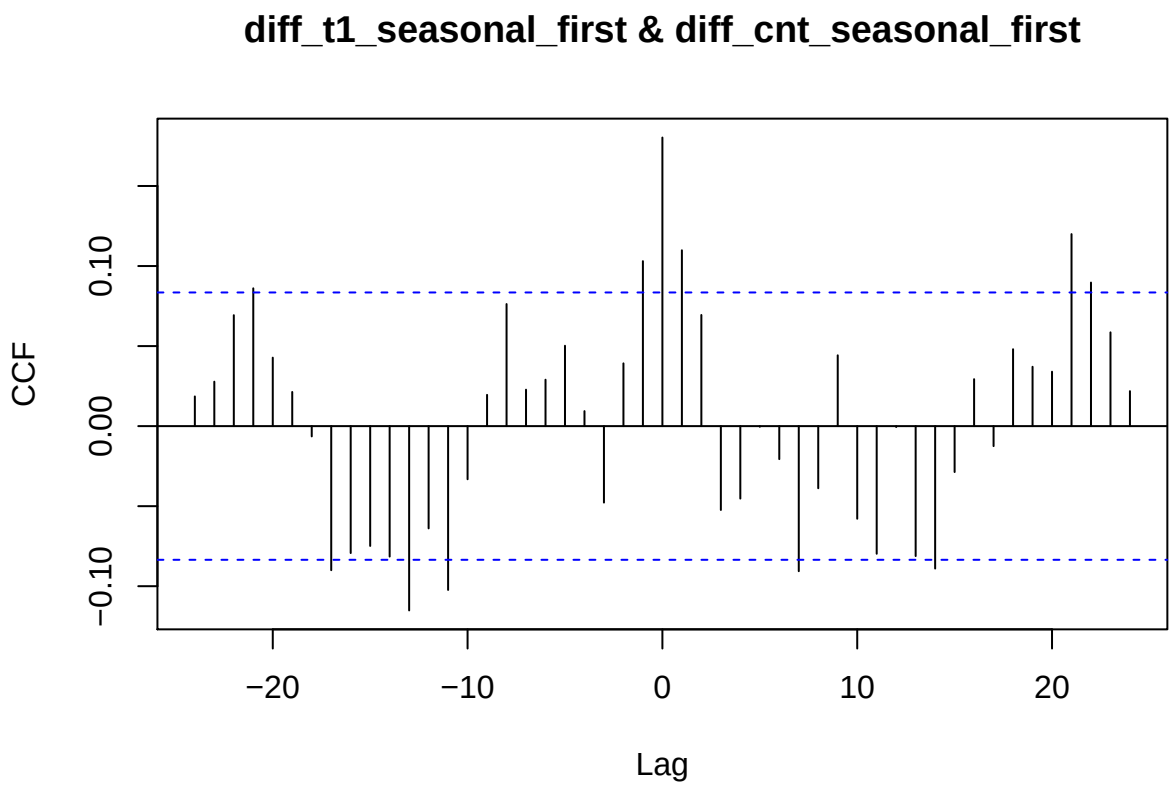
As demonstrated in the previous section, a seasonal differencings in addition to a first differencing is required to make the bike rentals time series stationary. Let us examine what happens when we apply the same differencing to the temperature time series.

```
diff_t1_seasonal = diff(bike_rentals$t1, lag=168)
diff_t1_seasonal_first = diff(diff_t1_seasonal, lag=1)

autoplot(ts(diff_t1_seasonal_first))
```



Also stationary! So we can go ahead and examine the CCF.

```
ccf(diff_t1_seasonal_first, diff_cnt_seasonal_first, ylab = "CCF")
```



We can see somewhat of a sinusoidal pattern in the CCF. Moreover, see that lags 1, 11 and 13, 18 and 21 of temperature are significantly correlated with the present values of bike rentals. Thus, I will consider lags of temperature up to lag(21)

▶ Investigate whether adding your lagged "cause'' variables (X) improve the prediction of your effect variable(s) Y. Use model selection based on information criteria. Describe your final chosen model

Fitting models

```
fit_causal <- bike_rentals %>%
# Restrict data so models use same fitting period
mutate(cnt = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,

# Estimate models
model(

  covariates = ARIMA(cnt ~ hum + wind_speed + lag(wind_speed) + weather_code + is_week
  lag1 = ARIMA(cnt ~ hum + wind_speed + lag(wind_speed) + weather_code + is_weekend +
  lag2 = ARIMA(cnt ~ hum + wind_speed + lag(wind_speed) + weather_code + is_weekend +
  lag3 = ARIMA(cnt ~ hum + wind_speed + lag(wind_speed) + weather_code + is_weekend +
```

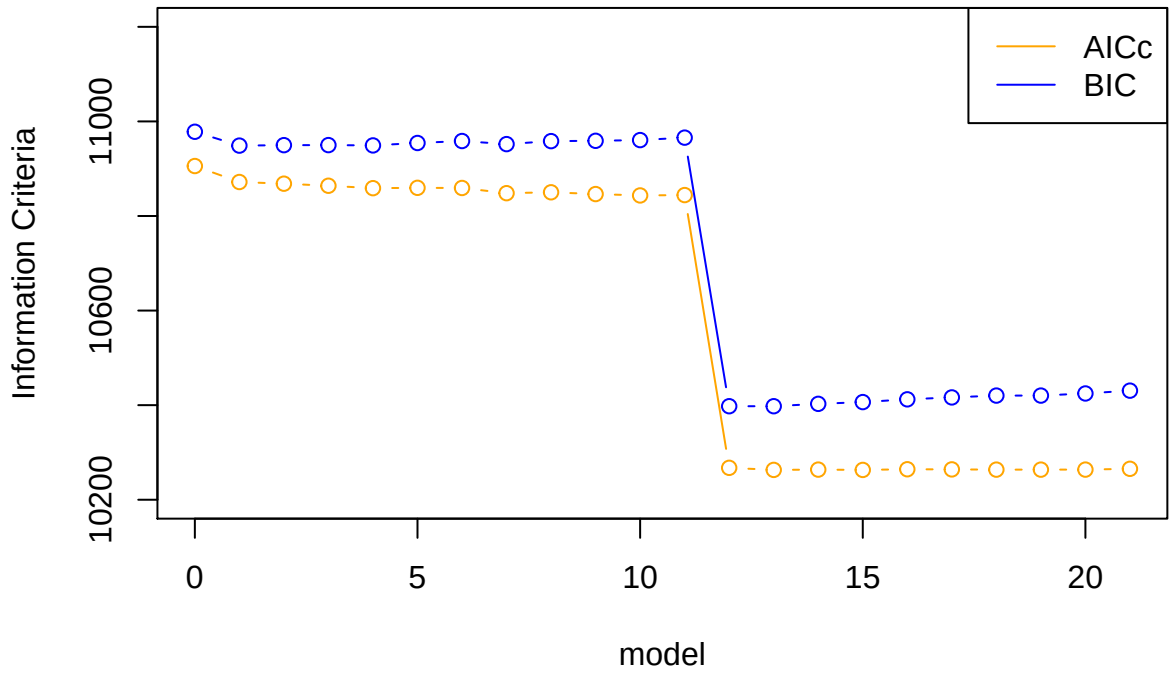
```
    lag4 = ARIMA(cnt ~ hum + wind_speed + lag(wind_speed) + weather_code + is_weekend +
    lag5 = ARIMA(cnt ~ hum + wind_speed + lag(wind_speed) + weather_code + is_weekend +

    lag6 = ARIMA(cnt ~ hum + wind_speed + lag(wind_speed) + weather_code + is_weekend +

    lag7 = ARIMA(cnt ~ hum + wind_speed + lag(wind_speed) + weather_code + is_weekend +

    lag8 = ARIMA(cnt ~ hum + wind_speed + lag(wind_speed) + weather_code + is_weekend +

    lag9 = ARIMA(cnt ~ hum + wind_speed + lag(wind_speed) + weather_code + is_weekend +

    lag10 = ARIMA(cnt ~ hum + wind_speed + lag(wind_speed) + weather_code + is_weekend +

    lag11 = ARIMA(cnt ~ hum + wind_speed + lag(wind_speed) + weather_code + is_weekend +

    lag12 = ARIMA(cnt ~ hum + wind_speed + lag(wind_speed) + weather_code + is_weekend +

    lag13 = ARIMA(cnt ~ hum + wind_speed + lag(wind_speed) + weather_code + is_weekend +

    lag14 = ARIMA(cnt ~ hum + wind_speed + lag(wind_speed) + weather_code + is_weekend +

        lag15 = ARIMA(cnt ~ hum + wind_speed + lag(wind_speed) + weather_code + is_weeke

        lag16 = ARIMA(cnt ~ hum + wind_speed + lag(wind_speed) + weather_code + is_weeke

        lag17 = ARIMA(cnt ~ hum + wind_speed + lag(wind_speed) + weather_code + is_weeke

        lag18 = ARIMA(cnt ~ hum + wind_speed + lag(wind_speed) + weather_code + is_weeke

        lag19 = ARIMA(cnt ~ hum + wind_speed + lag(wind_speed) + weather_code + is_weeke

        lag20 = ARIMA(cnt ~ hum + wind_speed + lag(wind_speed) + weather_code + is_weeke

        lag21 = ARIMA(cnt ~ hum + wind_speed + lag(wind_speed) + weather_code + is_weeke
    )
```

```
glance(fit_causal)
```

```
## # A tibble: 22 x 8
##    .model      sigma2 log_lik    AIC   AICc    BIC ar_roots  ma_roots
##    <chr>        <dbl>   <dbl>  <dbl>  <dbl>  <dbl> <list>    <list>
##  1 covariates 329914.  -5436. 10905. 10906. 10978. <cpl [5]> <cpl [0]>
##  2 lag1       313819.  -5418. 10871. 10872. 10949. <cpl [5]> <cpl [0]>
##  3 lag2       311747.  -5416. 10867. 10868. 10950. <cpl [5]> <cpl [0]>
##  4 lag3       309294.  -5412. 10863. 10864. 10950. <cpl [5]> <cpl [0]>
```

```
##  5 lag4       306514.  -5409. 10858. 10859. 10949. <cpl [5]> <cpl [0]>
##  6 lag5       306306.  -5408. 10858. 10860. 10954. <cpl [5]> <cpl [0]>
##  7 lag6       305666.  -5407. 10858. 10859. 10959. <cpl [5]> <cpl [0]>
##  8 lag7       300404.  -5400. 10847. 10848. 10952. <cpl [5]> <cpl [0]>
##  9 lag8       300715.  -5400. 10848. 10850. 10958. <cpl [5]> <cpl [0]>
## 10 lag9       298659.  -5397. 10845. 10846. 10959. <cpl [5]> <cpl [0]>
## # ... with 12 more rows
```

```
plot(seq(0,21), glance(fit_causal)$AICc,
     col = "orange", type = "b",
     ylab = "Information Criteria", xlab = "model",
     ylim = c(10200,11200))
lines(seq(0,21), glance(fit_causal)$BIC, col = "blue", type = "b")
legend("topright", c("AICc","BIC"), col = c("orange","blue"), lty = 1)
```



As we can see from the plot, adding lagged versions of temperature did indeed improve our model. There is a large improvement in BIC and AICc going from the lag 11 to lag 12, after which the performance stays stable. Let us inspect the model with lags of temperature up to lag 12.

```
report(fit_causal[13])
```

```
## Series: cnt
```

```
## Model: LM w/ ARIMA(4,0,0)(2,0,0)[24] errors
##
## Coefficients:
##          ar1      ar2     ar3      ar4    sar1     sar2      hum  wind_speed
##       1.0940  -0.6399  0.2360  -0.1371  0.8812  -0.1420  -9.1267      1.1328
## s.e.  0.0385   0.0558  0.0559   0.0386  0.0403   0.0412   2.0024      2.7907
##       lag(wind_speed)  weather_code2  weather_code3  weather_code4
##               -4.7031        19.5464       -14.5111       -23.4835
## s.e.           2.7876        28.8391        43.0069        89.5226
##       weather_code7  weather_code10  is_weekend1  lag(t1)  lag(t1, 2)
##            -77.5378       -154.0364     -20.2773  48.6142     38.9219
## s.e.        41.1887        108.7828      62.9348  17.8548     18.1786
##       lag(t1, 3)  lag(t1, 4)  lag(t1, 5)  lag(t1, 6)  lag(t1, 7)  lag(t1, 8)
##          -8.7322      3.3966     10.3748     -6.0411      4.7613    -13.3210
## s.e.     20.2564     20.9722     20.8860     20.9022     20.8302     21.0956
##       lag(t1, 9)  lag(t1, 10)  lag(t1, 11)  lag(t1, 12)  intercept
##         -26.8827      -7.5833      -5.9621     -11.8849  1563.7022
## s.e.     20.8955      20.0776      17.9605      17.8947   259.8265
##
## sigma^2 estimated as 125504:  log likelihood=-5103.48
## AIC=10264.95   AICc=10267.47   BIC=10397.75
```

We can see that the model that was estimated is a dynamic regression model with SARIMA(4,0,0)(2,0,0)[24] residuals. The AICc and BIC scores of the model are 10267.47 and 10397.75 respectively. What is interesting is that despite the major improvement from adding the 12th lag of temperature to the model, the coefficient for the 12th lag itself is fairly small, both in absolute terms and relative to its standard error. This is also true for the coefficients of lags 3, 4, 5, 6, 7, 8, 10 and 11.

Upon manual inspection of the models it was discovered that for the models with lags up to lag 11, only a regular ARIMA model without a seasonal component was fit to the residuals, while for the models with lags 12 and up, a SARIMA model was fit to the residuals. Hence, the improvements in the AICc and BIC are not due to the inclusion of lag12, but due to the presence of the seasonal component in the SARIMA residuals from model 12 onwards.

We can try to re-run the model estimates a second time and "force" the ARIMA() function to estimate the same SARIMA(4,0,0)(2,0,0)[24] residuals for each of the models and see what comes out. As we know that the information criteria remain relatively stable after the model with 12 lags, we could only estimate the first, say, 15 models (plus the covariates-only model).

```
fit_causal_forced_SARIMA <- bike_rentals %>%
# Restrict data so models use same fitting period
mutate(cnt = c(NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,

# Estimate models
```

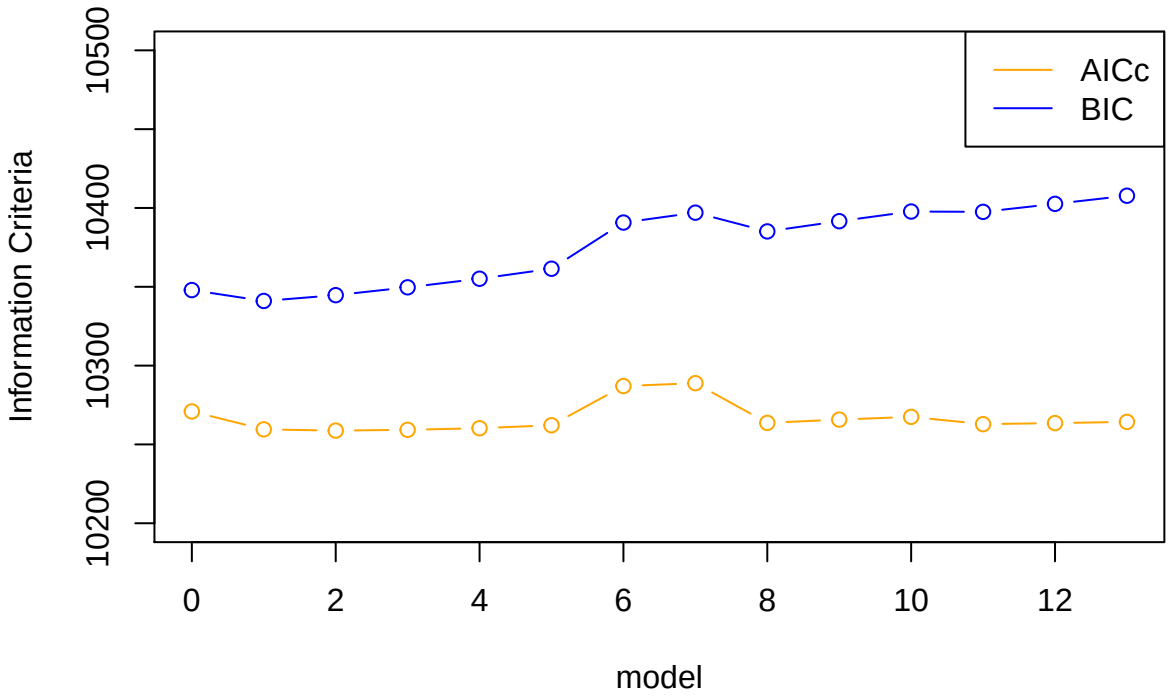
```
  model(

    covariates = ARIMA(cnt ~ hum + wind_speed + lag(wind_speed) + weather_code + is_week
    lag1 = ARIMA(cnt ~ hum + wind_speed + lag(wind_speed) + weather_code + is_weekend +
    lag2 = ARIMA(cnt ~ hum + wind_speed + lag(wind_speed) + weather_code + is_weekend +
    lag3 = ARIMA(cnt ~ hum + wind_speed + lag(wind_speed) + weather_code + is_weekend +
    lag4 = ARIMA(cnt ~ hum + wind_speed + lag(wind_speed) + weather_code + is_weekend +
    lag5 = ARIMA(cnt ~ hum + wind_speed + lag(wind_speed) + weather_code + is_weekend +

    lag6 = ARIMA(cnt ~ hum + wind_speed + lag(wind_speed) + weather_code + is_weekend +

    lag7 = ARIMA(cnt ~ hum + wind_speed + lag(wind_speed) + weather_code + is_weekend +

    lag8 = ARIMA(cnt ~ hum + wind_speed + lag(wind_speed) + weather_code + is_weekend +

    lag9 = ARIMA(cnt ~ hum + wind_speed + lag(wind_speed) + weather_code + is_weekend +

    lag10 = ARIMA(cnt ~ hum + wind_speed + lag(wind_speed) + weather_code + is_weekend +

    lag11 = ARIMA(cnt ~ hum + wind_speed + lag(wind_speed) + weather_code + is_weekend +

    lag12 = ARIMA(cnt ~ hum + wind_speed + lag(wind_speed) + weather_code + is_weekend +

    lag13 = ARIMA(cnt ~ hum + wind_speed + lag(wind_speed) + weather_code + is_weekend +

    lag14 = ARIMA(cnt ~ hum + wind_speed + lag(wind_speed) + weather_code + is_weekend +

        lag15 = ARIMA(cnt ~ hum + wind_speed + lag(wind_speed) + weather_code + is_weeke
```

```
## Warning: It looks like you're trying to fully specify your ARIMA model but have not s
## You can include a constant using `ARIMA(y~1)` to the formula or exclude it by adding

## Warning: It looks like you're trying to fully specify your ARIMA model but have not s
## You can include a constant using `ARIMA(y~1)` to the formula or exclude it by adding

## Warning: 1 error encountered for lag6
## [1] Could not find an appropriate ARIMA model.
## This is likely because automatic selection does not select models with characteristic
## For more details, refer to https://otexts.com/fpp3/arima-r.html#plotting-the-characte

## Warning: 1 error encountered for lag9
## [1] Could not find an appropriate ARIMA model.
## This is likely because automatic selection does not select models with characteristic
## For more details, refer to https://otexts.com/fpp3/arima-r.html#plotting-the-characte
```

```r
glance(fit_causal_forced_SARIMA)
```

```
## # A tibble: 14 x 8
##    .model      sigma2 log_lik    AIC   AICc    BIC ar_roots   ma_roots
##    <chr>        <dbl>   <dbl>  <dbl>  <dbl>  <dbl> <list>     <list>
##  1 covariates 128363.  -5118. 10270. 10271. 10348. <cpl [52]> <cpl [0]>
##  2 lag1       126234.  -5111. 10259. 10260. 10341. <cpl [52]> <cpl [0]>
##  3 lag2       125874.  -5110. 10258. 10259. 10345. <cpl [52]> <cpl [0]>
##  4 lag3       125765.  -5109. 10258. 10259. 10350. <cpl [52]> <cpl [0]>
##  5 lag4       125749.  -5108. 10259. 10260. 10355. <cpl [52]> <cpl [0]>
##  6 lag5       125897.  -5108. 10261. 10262. 10361. <cpl [52]> <cpl [0]>
##  7 lag7       130180.  -5120. 10285. 10287. 10391. <cpl [52]> <cpl [0]>
##  8 lag8       130312.  -5120. 10287. 10289. 10397. <cpl [52]> <cpl [0]>
##  9 lag10      125228.  -5104. 10261. 10264. 10385. <cpl [52]> <cpl [0]>
## 10 lag11      125397.  -5104. 10263. 10266. 10392. <cpl [52]> <cpl [0]>
## 11 lag12      125504.  -5103. 10265. 10267. 10398. <cpl [52]> <cpl [0]>
## 12 lag13      124546.  -5100. 10260. 10263. 10398. <cpl [52]> <cpl [0]>
## 13 lag14      124439.  -5099. 10261. 10264. 10403. <cpl [52]> <cpl [0]>
## 14 lag15      124397.  -5099. 10261. 10264. 10408. <cpl [52]> <cpl [0]>
```

```r
plot(seq(0,13), glance(fit_causal_forced_SARIMA)$AICc,
     col = "orange", type = "b",
     ylab = "Information Criteria", xlab = "model",
     ylim = c(10200,10500))
lines(seq(0,13), glance(fit_causal_forced_SARIMA)$BIC, col = "blue", type = "b")
legend("topright", c("AICc","BIC"), col = c("orange","blue"), lty = 1)
```



Disclaimer: the lag6 and lag9 models could not be estimated by forcing the ARIMA() function to estimate SARIMA(4,0,0)(2,0,0)[24] residuals. Hence why there are 13 instead of 16 models, and the x-axis is incorrect after the 5th lag model.

Now we can see a completely different chart. The values of the AICc start around 10270 for the covariates-only modelas opposed to 10843 when we didn't force SARIMA residuals. This confirms our observation that the improvements from lag 11 to lag 12 in the previous chart were not due to the extra lag but the seasonal component in the residuals, which was not present in models 0:11.

Here, the model with the best AIC (10258.76) is the one with the 1st and 2nd lags of temperature, while the model with the best BIC (10341.01) is the model with only the 1st lag. Both model have better scores on the information criteria than the covariates-only model (AICc: 10270.94 & BIC: 10347.91). Let us inspect these models:

```
model_best_aic <- fit_causal_forced_SARIMA[3]
report(model_best_aic)
```

```
## Series: cnt
## Model: LM w/ ARIMA(4,0,0)(2,0,0)[24] errors
##
## Coefficients:
##          ar1      ar2     ar3      ar4    sar1     sar2       hum  wind_speed
##       1.1020  -0.6387  0.2425  -0.1357  0.8816  -0.1302  -10.2262      2.3417
```

```
## s.e.  0.0389   0.0560  0.0560   0.0381  0.0396   0.0405    2.0064      2.8032
##       lag(wind_speed)  weather_code2  weather_code3  weather_code4
##               -3.7022        17.1006       -19.5386        -4.3955
## s.e.           2.8002        29.0225        42.9339        89.7812
##       weather_code7  weather_code10  is_weekend1  lag(t1)  lag(t1, 2)
##            -86.6296       -163.5139      34.6285  34.7033     24.1250
## s.e.        41.4234        108.7534      62.5018  13.7780     14.1221
##       intercept
##       1231.0394
## s.e.   253.3493
##
## sigma^2 estimated as 125874:  log likelihood=-5109.84
## AIC=10257.68   AICc=10258.76   BIC=10344.68
```

```
model_best_bic <- fit_causal_forced_SARIMA[2]
report(model_best_bic)
```

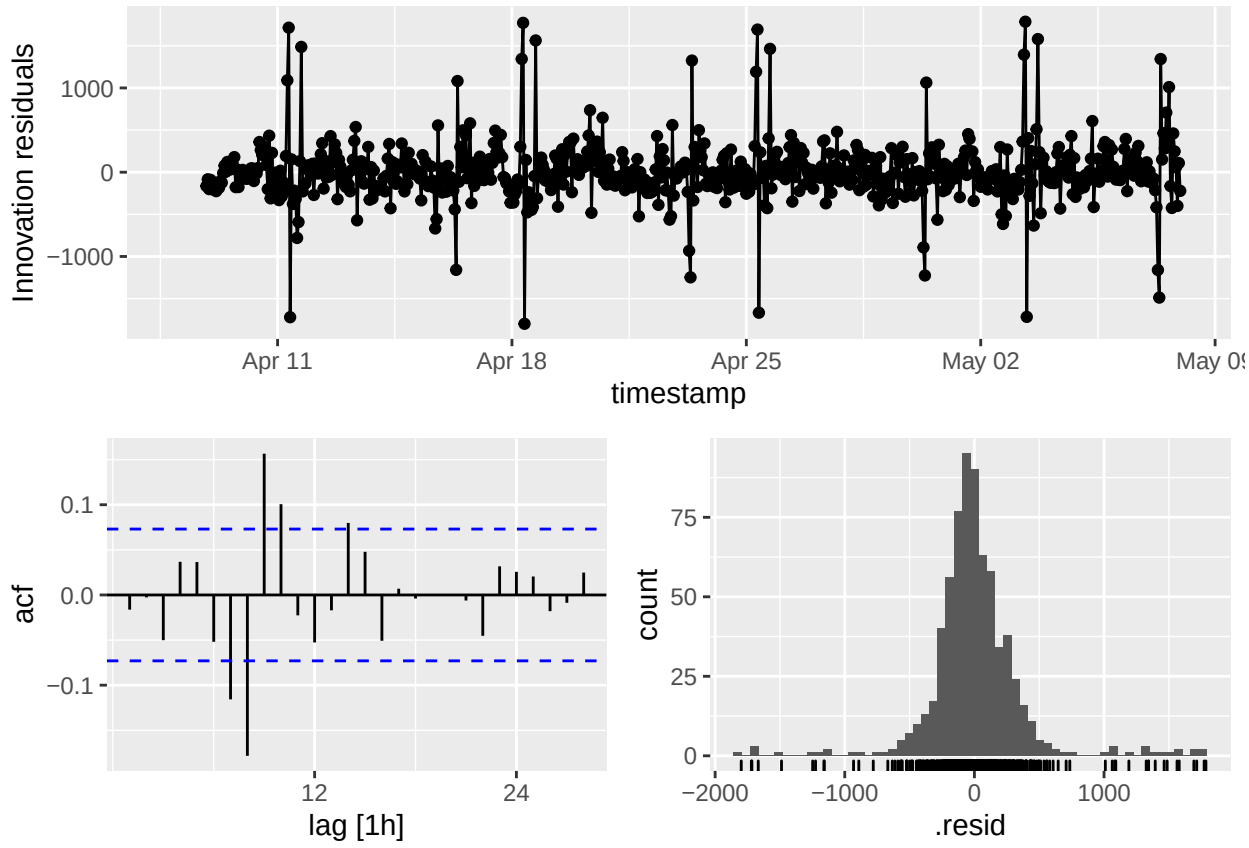
```
## Series: cnt
## Model: LM w/ ARIMA(4,0,0)(2,0,0)[24] errors
##
## Coefficients:
##          ar1      ar2     ar3      ar4    sar1     sar2       hum  wind_speed
##       1.0960  -0.6328  0.2381  -0.1353  0.8804  -0.1297  -10.8833      2.2972
## s.e.  0.0388   0.0559  0.0559   0.0381  0.0397   0.0405    1.9650      2.8054
##       lag(wind_speed)  weather_code2  weather_code3  weather_code4
##               -3.0245        15.0910       -25.9746         6.7611
## s.e.           2.7761        29.1375        42.9909        89.9949
##       weather_code7  weather_code10  is_weekend1  lag(t1)  intercept
##            -83.8076       -187.1014      16.1226  46.1056  1411.2641
## s.e.        41.5701        108.5499      61.4310  12.2016   227.9971
##
## sigma^2 estimated as 126234:  log likelihood=-5111.29
## AIC=10258.59   AICc=10259.56   BIC=10341.01
```

The model with best AIC (lag), has a lower coefficient for the 1st lag of temperature (34.7033) than the model with the best BIC (46.1056). Moreover the best AIC model has a coefficient of 24.1250 for the 2nd lag of temperature. Barring these differences, all other coefficients and standard errors are fairly similar for both models.

Let us inspect the residuals of both models.

```
gg_tsresiduals(model_best_aic)
```

```
## Warning: Removed 21 row(s) containing missing values (geom_path).

## Warning: Removed 21 rows containing missing values (geom_point).

## Warning: Removed 21 rows containing non-finite values (stat_bin).
```

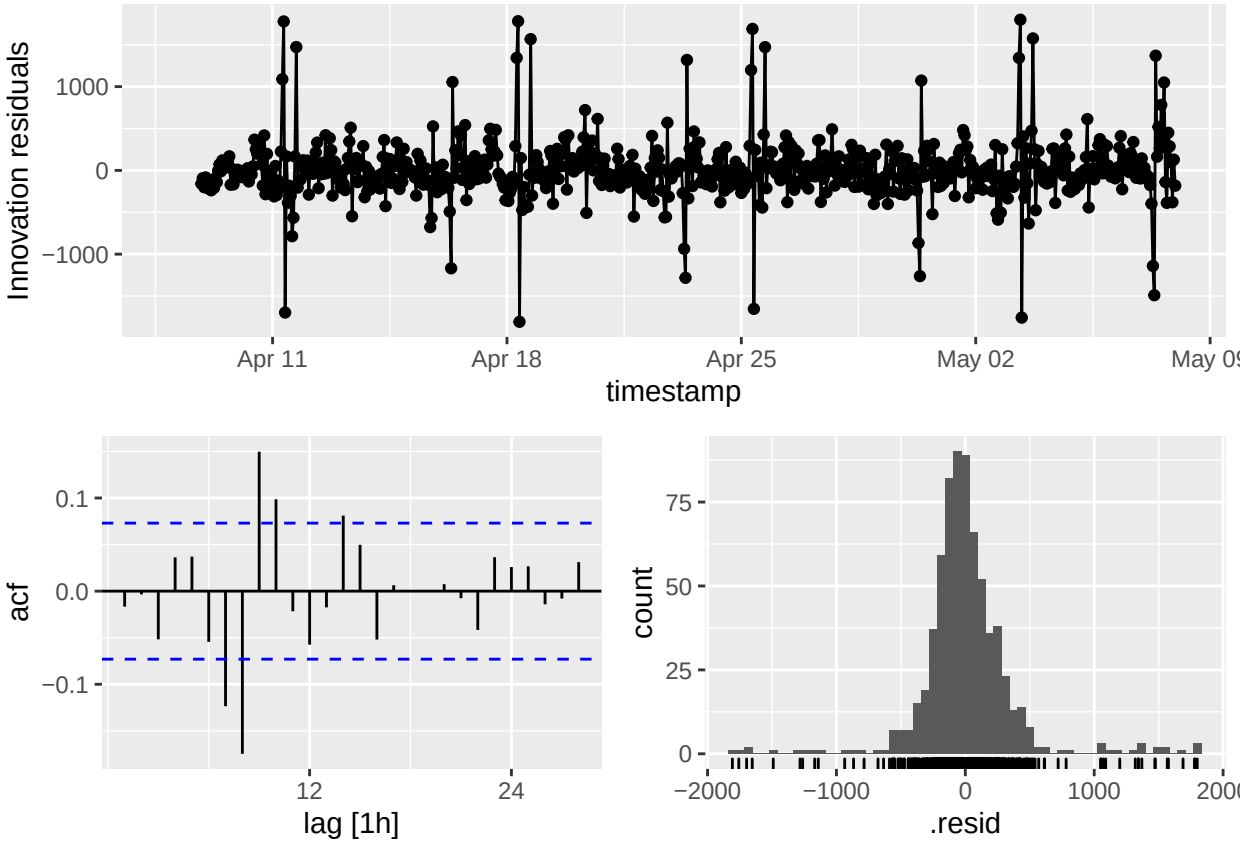



```
gg_tsresiduals(model_best_bic)
```

```
## Warning: Removed 21 row(s) containing missing values (geom_path).
```

```
## Warning: Removed 21 rows containing missing values (geom_point).
```

```
## Warning: Removed 21 rows containing non-finite values (stat_bin).
```



The two models have very similar residual plots, which show no indication of which model should be preferred. Thus, we can go with the simpler model which only has the 1st lag of temperature as a predictor.

It is also worth noting that, while the residuals appear to be stationary, they do not seem to be white noise due to the significant autocorellations at lags 7, 8, 9, 10 and 14 on the ACF plot.

3.3 Conclusion and critical reflection

▶ Based on the result of your analysis, how would you answer your causal research question?

We cannot conclude that temperature is not a cause of bike rentals as adding temperature as a predictor improves the AICc and BIC compared to the covariates-only model.

However, to conclude that temperature is a cause of bike rentals depends on the validity of the assumption of no unobserved confounding (also known as **sufficiency**). Namely, the assumption that any and all confounders are contained within the subset of covariates used to train the model. The plausibility of this assumption is discussed in the next section.

If we were to assume no unobserved confounding, we could say that the individual causal effect of temperature on bike rentals for the city of London is an increase of 46.1056 bike

rentals per hour for every degree that temperature goes up. However, this statement is reliant on an additional assumption - namely, correct model specification.

Moreover, if we were to interpret the coefficient as the individual causal effect, we would have to be careful as to how we frame it in terms of direct or total effect. There are two important points here: * We cannot interpret it as the total effect as we have conditioned on a variable that could potentially be a mediator in our dynamic regression model - namely, the present values of humidity. * We cannot (confidently) interpret it as the direct effect as there could be other mediators which have not been accounted for.

Thus, the coefficient could potentially be interpreted as the total effect excluding the mediated effect through humidity (again, assuming sufficiency and correct model specification).

▶ Making causal conclusions on the basis of your analysis is reliant on a number of assumptions. Pick a single assumption that is necessary in the approach you chose. Discuss the plausibility and possible threats to the validity of this assumption in your specific setting (< 75 words)

To determine whether the assumption of sufficiency is reasonable, we would have to ask what could be unobserved common causes of temperature and bike rentals? And the answer is – not much. Besides CO2 emissions, Earth's position in orbit and the time of day (i.e., the Earth's rotation), temperature is largely independent of all other variables. The first two are not relevant due to their large timescale, while the latter is controlled for through the presence of the SARIMA(4,0,0)(2,0,0)[24] component in the model. Thus, we can be fairly confident that the assumption of unobserved confounding is valid, and temperature is a cause of bike rentals.