Investigating Causal Relations Between Location, Steps Taken, and Food Purchased

Julia Ruiter

ADS, 2021-2022

Contents

0. Prepare

There are three fundamental constants in the life of Julia: 1. Julia uses (public) OV transit near-daily and is constantly bouncing all over the Randstad region. 2. Julia is really bad at carrying their phone with them. 3. Julia is always hungry.

These are three disparate, but easily tracked ideas–so I thought it would be interesting to investigate what correlations exist between them. I believe I have a fairly predictable schedule/behaviour pattern in these categories, but I'm interested to see if these correlations can be statistically confirmed.

On November 16 2021, I was tasked with finding a suitable data set for time series analysis, so I started gathering sources that were already tracking my daily life and documented some initial hypothesis, making explicit note of the date in case knowing I was tracking my steps would lead to any difference in carrying my phone with me and thus "increasing" the steps I took each day.

0.1 Dataset overview

I chose to focus in on the time frame from September 1 2021 to December 31 2021 so that I have a complete dataset of my (phone-logged) steps per day, OV travels, and debit card history for this 4-month period. Though I would have liked to see what effect quitting work (in Rotterdam) and starting my master's programme (in Utrecht) would have on these variables, I unfortunately do not have access to any consistent transit data prior to 1 September.

NOTE: originally, I did all the data wrangling and dataframe edits/concatenations in Excel, but I decided to challenge myself to get better at R. My apologies for the messy and long preprocessing steps.

0.2 Initial Hypotheses

1. Steps per day will correlate highly with distance from home. Because I generally only have my phone on me when I am traveling long distances or for long periods of time, I can only then log steps on days meeting these criteria. This means the strongest correlation should be (same-day) on days where I am (partially) in Amsterdam or other day trips. Utrecht will probably be the exception to this rule as day in Utrecht are spent mostly in the UU library or in lecture.

2. Food purchases will correlate highly with days out. Large-non-essential expenditure days should correlate with being further afield (such as restaurants), while days spent in multiple locations should correlate highly with small nonessential food purchases (I have a horrible addiction to kaasbroodjes at the train stations). There should also be a consistent 1-day lag in moderate food purchases after consecutive days away (post-trip empty fridge).

3. There should be a small increase in steps per day after 16 November 2021 as I make a more concerted effort to carry my phone with me (an unexpected intervention).

0.3 Lots and lots of data wrangling

▶ Load the R-packages you will use.

```
library(tidyverse)
library(fpp3)
library(tseries)
library(expsmooth)
library(CausalImpact)
require(fpp3)
require(forecast)
library(jsonlite)
```

▶ Include R-code you used to load (and prepare) the data. The data I will be using for time series analysis actually comes from 3 different sources (attached). I will import the 3 csv files then combine them into one large dataset describing "What does Julia do (if and) when they are out an about?"

```
steps_df <- read.csv('Pedometer.csv')
transit_df <- read.csv('OV.csv')
expenses_df <- read.csv('Expenses.csv')
```

'steps_df' has one entry per day, and is ready to use. However, some additional preprocessing is needed on the 'transit_df' dataset since it often has multiple entries per day (or none). To extract the important information (where is Julia?), this data will be transformed using dummy-coding for all locations visited in the day since it is unclear yet if steps and purchases will correlate.

```
# first, drop irrelevant columns
cols_to_drop <- c('Card.Type', 'Weekday', 'Departure', 'Destination',
                  'Check.out', 'Type', 'Product')
transit_df <- transit_df[ , !(names(transit_df) %in% cols_to_drop)]

# then, total transit stops per day
transit_stops <- transit_df %>%
  group_by(Date) %>%
  mutate(Transit.Count = n())

# then, create dummy variable columns for locations
#(Departure AND/OR Arrival City)
```

```r
transit_df_dummies <- transit_stops %>%
  # Home
  mutate(Voorburg = ifelse(Arrival.City == 'Voorburg',1,
                    ifelse(Departure.City == 'Voorburg',1,0)))  %>%
  # Close to Home
  mutate(Den.Haag = ifelse(Arrival.City == 'Den Haag',1,
                    ifelse(Departure.City == 'Den Haag',1,
                          ifelse(Arrival.City == 'Rijswijk',1,
                          ifelse(Departure.City == 'Rijswijk',1,
                                 ifelse(Arrival.City == 'Leidschendam',1,
                                 ifelse(Departure.City == 'Leidschendam',1,
                                        0)))))) %>%
  # School
  mutate(Utrecht = ifelse(Arrival.City == 'Utrecht',1,
                   ifelse(Departure.City == 'Utrecht',1,0)))  %>%#
  # Partner
  mutate(Amsterdam = ifelse(Arrival.City == 'Amsterdam',1,
                     ifelse(Departure.City == 'Amsterdam',1,0)))  %>%#
  # Day Trip
  mutate(Day.Trip = ifelse(Arrival.City == 'Maastrict',1,
                    ifelse(Departure.City == 'Maastricht',1,
                          ifelse(Arrival.City == 'Eindhoven',1,
                          ifelse(Departure.City == 'Eindhoven',1,
                                 ifelse(Arrival.City == 'Zoetermeer',1,
                                 ifelse(Departure.City == 'Zoetermeer',1,
                                        ifelse(Arrival.City == 'Gouda',1,
                                        ifelse(Departure.City == 'Gouda',1,
                                               ifelse(Arrival.City ==
                                                        'Rotterdam',1,
                                               ifelse(Departure.City ==
                                                        'Rotterdam',1,
                                                         0))))))))))


used_cols_to_drop <- c('Departure.City', 'Arrival.City')
transit_locations_df <-
  transit_df_dummies[ , !(names(transit_df_dummies) %in% used_cols_to_drop)]
```

```r
# condense to one line per day using aggregate and max values
condensed_locations_df <-
  aggregate(transit_locations_df[,4:8], transit_locations_df[2:3],
  function(x){
    if(class(x) == 'numeric'){
      return(  names(which.max(table(x))) ) }
    if(class(x) != 'numeric'){
      return( x ) }})

# preview the final transit dataframe:
transit_df <- condensed_locations_df[order(condensed_locations_df$Date),]
```

However, there are still many gaps in the data; these will be filled in using locations from ‘expenses_df’. ‘expenses_df’ has already been stripped of automatic withdrawals as these charges occur regardless of Julia’s location. This will then be combined with the earlier “transit_df” to make one final location dataset.

```r
# rename 'Datum' column to 'Date'
names(expenses_df)[names(expenses_df) == 'Datum'] <- 'Date'

# repeat the location dummy variable process from before
expenses_locations_df <- expenses_df %>%
  # Home
  mutate(Voorburg = ifelse(Where == 'Voorburg',1,0))  %>%
  # Close to Home
  mutate(Den.Haag = ifelse(Where == 'Den Haag',1,
                    ifelse(Where == 'Rijswijk',1,
                    ifelse(Where == 'Delft',1,
                    ifelse(Where == 'Leidschendam',1,0))))) %>%
  # School
  mutate(Utrecht = ifelse(Where == 'Utrecht',1,0))  %>%#
  # Partner
  mutate(Amsterdam = ifelse(Where == 'Amsterdam',1,0))  %>%#
  # Day Trip
  mutate(Day.Trip = ifelse(Where == 'Maastrict',1,
                    ifelse(Where == 'Arcen',1,
                    ifelse(Where == 'Eindhoven',1,
                    ifelse(Where == 'Zoetermeer',1,
                    ifelse(Where == 'Gouda',1,
                    ifelse(Where == 'Rotterdam',1,0)))))))

# aggregate and drop non-relevant columns
locations_from_expenses_df <-
  aggregate(expenses_locations_df[,8:12], expenses_locations_df[5],
  function(x){
    if(class(x) == 'numeric'){
      return(  names(which.max(table(x))) ) }
    if(class(x) != 'numeric'){
      return( x ) }})
```

```
# preview the dummy-variable location transit dataframe:
exp_loc_df <-
  locations_from_expenses_df[order(locations_from_expenses_df$Date), ]

# append a blank 'Transit.Count' column
#(transit = 0 unless otherwise specified in earlier df)
exp_loc_df$Transit.Count <- 0
```

…almost done with location preprocessing:

```
# remove lines from exp_loc_df where date is already in transit df
reduced_exp_loc_df <- exp_loc_df[!(exp_loc_df$Date %in% transit_df$Date),]

# now combine the two datafames.  this results in 1 line per date
combined_location_dfs <- rbind(transit_df, reduced_exp_loc_df)
combined_location_dfs <- combined_location_dfs[order(combined_location_dfs$Date), ]
```

Returning to 'expenses_df', we still need to sum and count purchases per date

```
# drop irrelevant columns
exp_cols_to_drop <- c('Category_Specific', 'Where', 'Volgnr',
                      'Naam.tegenpartij')
expenses_reduced_df <-
  expenses_df[ , !(names(expenses_df) %in% exp_cols_to_drop)]

# drop all rows not pertaining to food
food_exp_prelim <- expenses_reduced_df[grepl('food', expenses_reduced_df$Category_Broad)

# change 'Bedgrag' format from Dutch comma to standard decimal
# and change type to numeric (and positive)
food_exp_prelim$Bedrag <- gsub(',', '.', food_exp_prelim$Bedrag)
food_exp_prelim <-
  transform(food_exp_prelim, Bedrag = as.numeric(Bedrag) * (-1))

# sumarize stats by food type (essential/nonessential)
food_exp_nonessential <-
  food_exp_prelim[grepl('_non', food_exp_prelim$Category_Broad),] %>%
  group_by(Date) %>%
  summarize(Nonessential.Food = sum(Bedrag))
food_exp_essential <-
  food_exp_prelim[grepl('_es', food_exp_prelim$Category_Broad),] %>%
  group_by(Date) %>%
  summarize(Essential.Food = sum(Bedrag))
```

```r
# outer join the charges and replace 'NA' with 0
all_food_exp <-
  merge(food_exp_nonessential, food_exp_essential, by = 'Date', all=TRUE)
all_food_exp[is.na(all_food_exp)] <- 0
```

The individual dataframes are now good to go! They just need to be combined into one large tsibble in order to perform timeseries analysis:

```r
# outer join the charges and replace 'NA' with 0
#steps_df <- read.csv('Pedometer.csv')
steps_and_food <-
  merge(steps_df, all_food_exp, by = 'Date', all=TRUE)
life_of_Julia_df <-
  merge(steps_and_food, combined_location_dfs, by = 'Date', all=TRUE)
life_of_Julia_df[is.na(life_of_Julia_df)] <- 0
life_of_Julia_df$Date <-
  ymd(life_of_Julia_df$Date)  # convert to date type
life_of_Julia_df <-
  tibble::rowid_to_column(life_of_Julia_df, "ID")  # set index
life_of_Julia <-
  as_tsibble(life_of_Julia_df, index = Date, key=NULL)

# preview resulting tsibble
life_of_Julia
```

```
## # A tsibble: 122 x 12 [1D]
##       ID Date       Day.Number Daily.Steps Nonessential.Food Essential.Food
##    <int> <date>          <int>       <int>             <dbl>          <dbl>
##  1     1 2021-09-01          4       13593              8.15           0
##  2     2 2021-09-02          5        4199              0              0
##  3     3 2021-09-03          6       10965             29.4            0
##  4     4 2021-09-04          7        7230              0             19.4
##  5     5 2021-09-05          1          85             12              0
##  6     6 2021-09-06          2         132              2.6            0
##  7     7 2021-09-07          3       10958              0              0
##  8     8 2021-09-08          4        3363              0              3.09
##  9     9 2021-09-09          5        5922              0             30.4
## 10    10 2021-09-10          6       11898              0              0
## # ... with 112 more rows, and 6 more variables: Transit.Count <dbl>,
## #   Voorburg <chr>, Den.Haag <chr>, Utrecht <chr>, Amsterdam <chr>,
## #   Day.Trip <chr>
```

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.

Quick summary of all the preprocessing done above:

'life_of_Julia' is a tsibble that tracks Julia's (my) daily steps. Other measured variables (not directly plotted) are locations Julia visited that day and amount of money Julia spent on essential and nonessential food items

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 of this time series is the number of steps logged on my phone pedometer ('Daily.Steps'). The value is a daily aggregate of all steps taken on a particular day (while I had my phone on my anyways).

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

Location is the most sensible predictor variable for daily steps taken because the biggest factor in whether steps are logged is if I have my phone on me at all. As described in the 'Preprocessing' section, location is a proxy for if I'm out of the house for long periods of time as I often don't carry my phone with me for excursion under 1 hour. Because location is a categorical variable, it has instead been split into a set of dummy variables ('Voorburg', 'Den.Haag', 'Utrecht', 'Amsterdam', and a catch-all-else 'Day.Trip'). Because I may be in several cities on any given day, more than one column may have a value of '1'.

Food expenditures (split into 'Essential.Food' and 'Nonessential.Food') may also be valid predictor as 'Essential.Food' and 'Daily.Steps' share the common cause of location.

'Day.Number' (or 'Date' when taken on a season of 7) is also a significant contributer as my general location is only 'Utrecht' on weekdays and I generally only go to 'Amsterdam' on weekend days. However, this variable is not a good predictor in practice because I am very inconsistent in which weekdays I make the trek, and I do not go to Amsterdam every weekend.

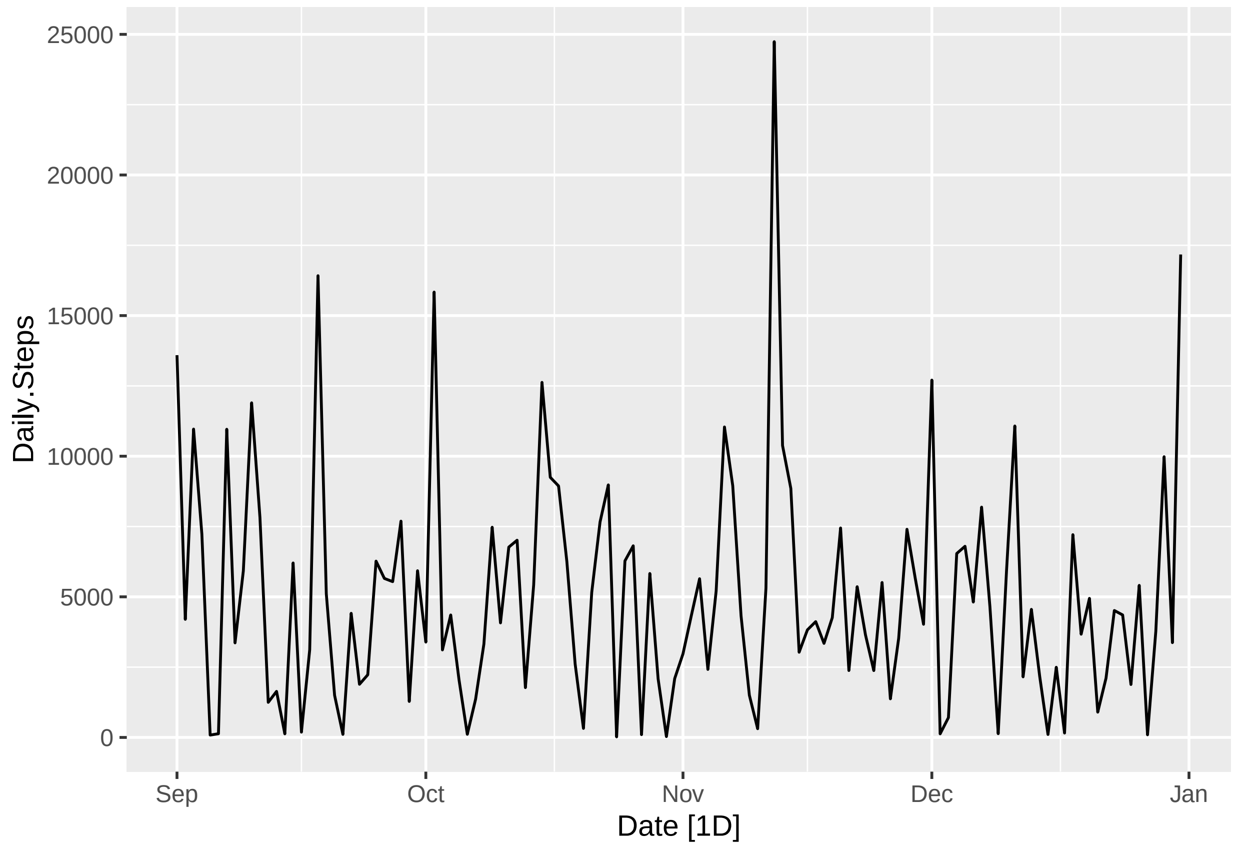
▶ What are the cause(s) you will consider? Why would this make sense as a cause?

For the reasons listed above, the location variables of 'Voorburg', 'Den.Haag', 'Utrecht', 'Amsterdam', and 'Day.Trip' (and the linear combinations of these columns) are my strongest cause of steps taken.

1.2. Visualize your data

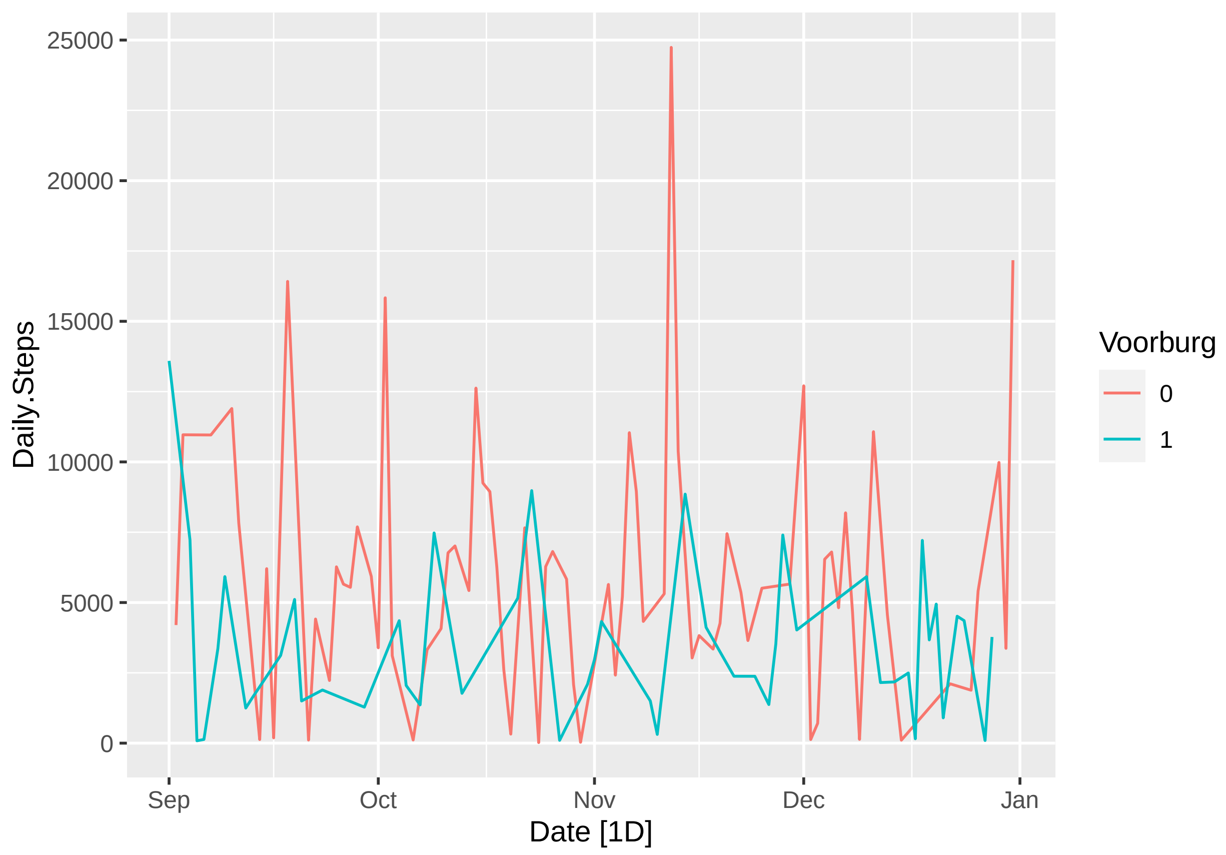
▶ Create a sequence plot of the data with the function autoplot(). Interpret the results.

```
autoplot(life_of_Julia, Daily.Steps)
```



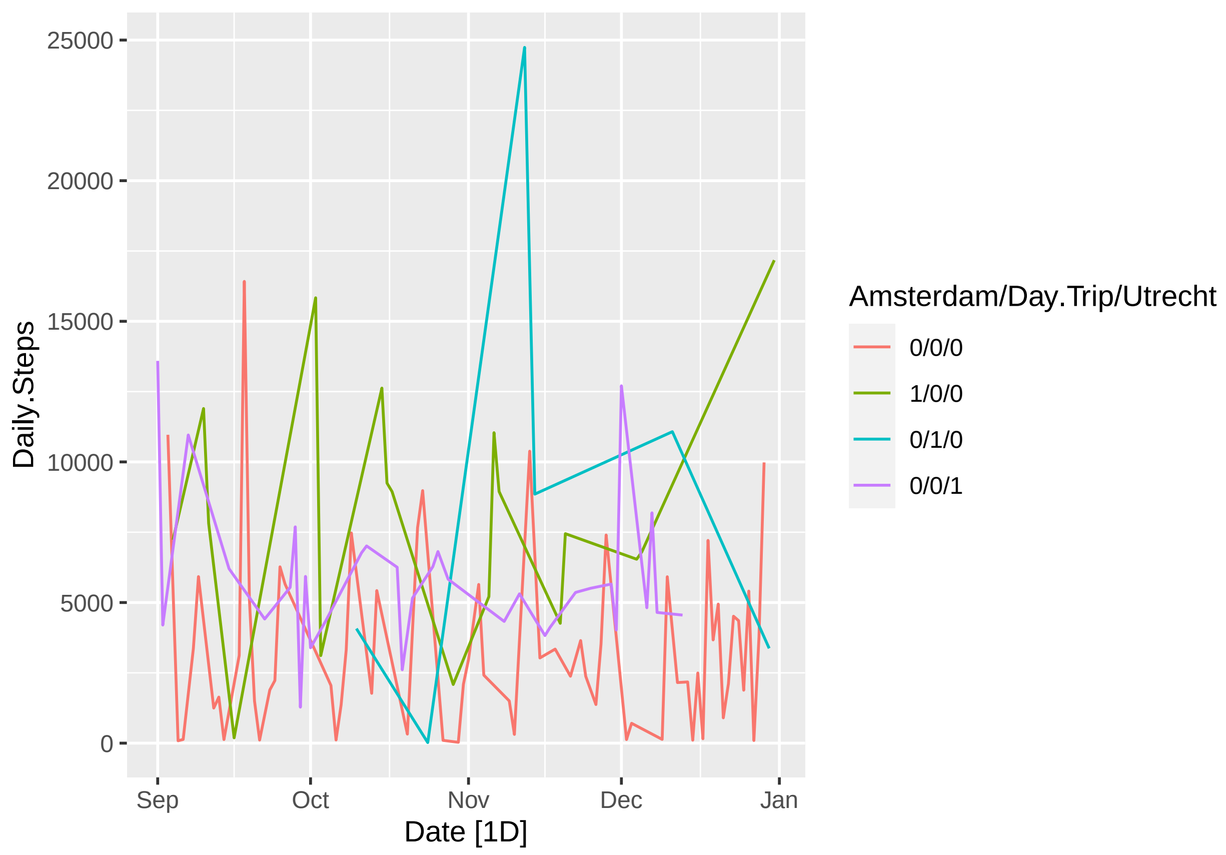
As predicted, there is definitely some kind of cycle, but the peaks occur every one or two weeks instead of every week. This makes sense because if I only have my phone on my if I'm away and am only away in Amsterdam every other week or so, supporting my initial hypothesis that steps are tied to location. I will create a new tsibble below with locations = Voorburg as a key variable, and see if that confirms anything.

```
life_of_Julia_Voorburg <-
  as_tsibble(life_of_Julia, index = Date, key='Voorburg')
autoplot(life_of_Julia_Voorburg, Daily.Steps)
```



Indeed! 'Voorburg' = 0 indicates days where I was entirely out of the house (usually in Amsterdam) and not in Voorburg at all, and indeed these tend to be the day with the most steps! Let's look a bit close at which days are 'Amsterdam' and which days are 'Day.Trip':

```
life_of_Julia_away <-
  as_tsibble(life_of_Julia, index = Date,
             key=c('Amsterdam','Day.Trip','Utrecht'))
autoplot(life_of_Julia_away, Daily.Steps)
```

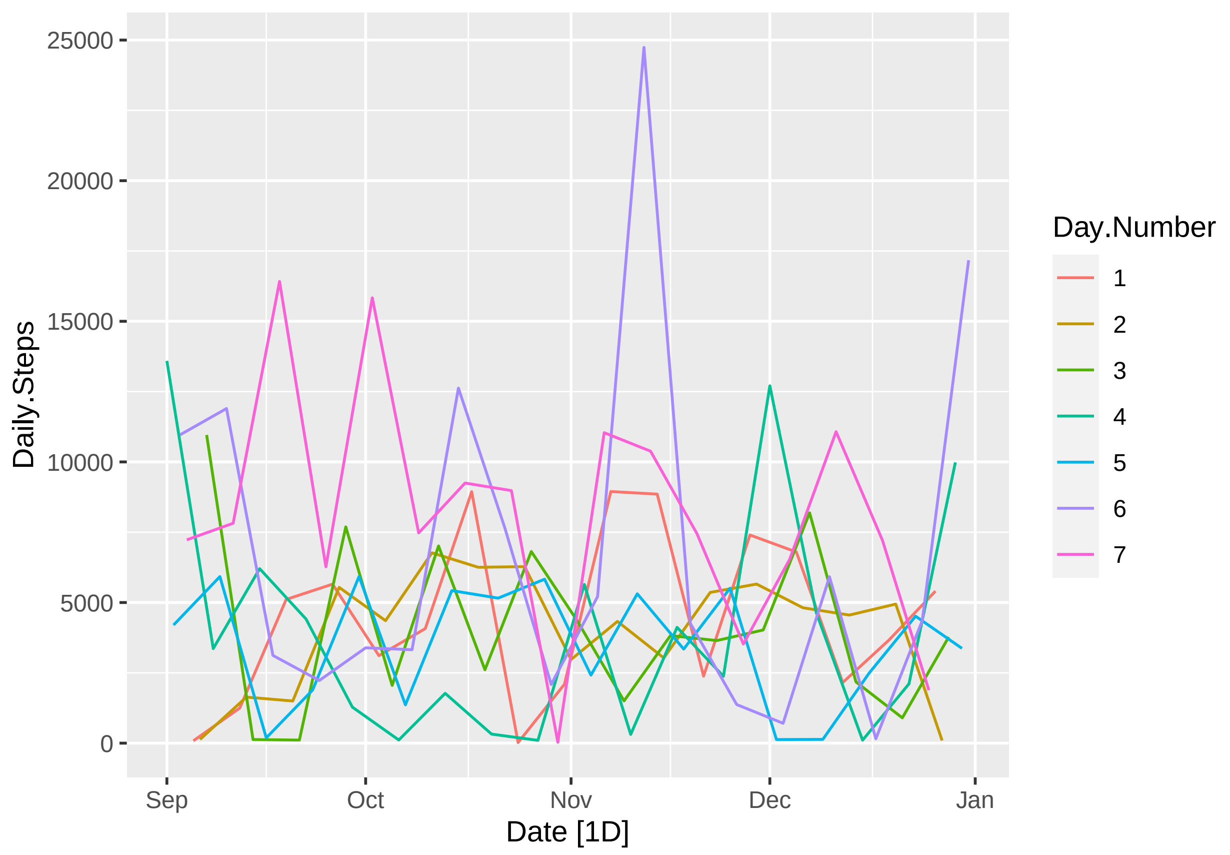


Off the top of my head, I can see there's a small error in the data: 18 September should be a Day.Trip = 1 (I was in Arcen for a Rennaisance Faire), but since I didn't take the train and didn't pay anything with card, it makes sense that this day is not flagged as anything at all.

Splitting data into categories like this makes it harder to visualize continuity, but it is clear that most of the peaks are associated with 'Amsterdam' = 1. Additionally, it confirms that 'Utrecht' is also a contributor to steps (on non-weekend days).

Let's do one last visualization of week days:

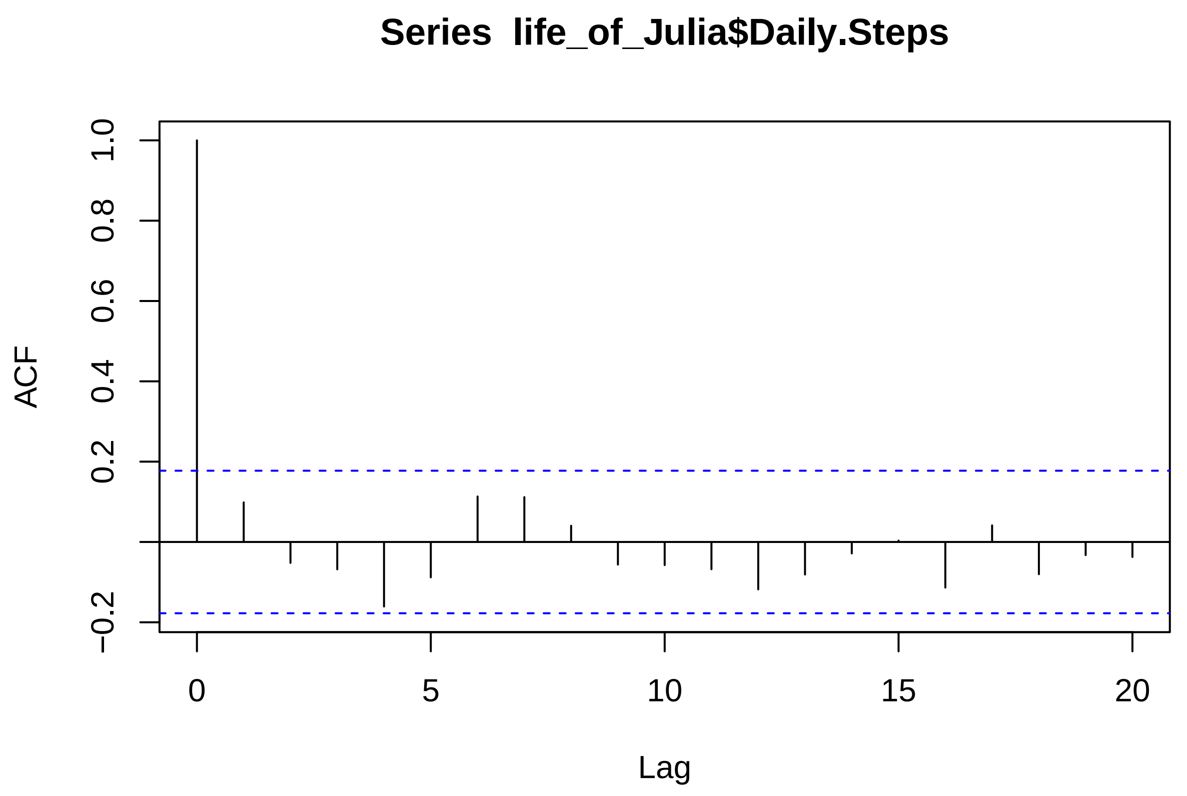
```
life_of_Julia_days <-
  as_tsibble(life_of_Julia, index = Date, key=Day.Number)
autoplot(life_of_Julia_days, Daily.Steps)
```



Again, continuity is a bit hard to visualize here, but what is clear is that Day.Number = 6 (Saturday) and Day.Number = 7 (Sunday) are the days associated with the most amount of steps. These observations together conclude that 'Voorburg' is the strongest cause/correlation to daily steps, followed by 'Utrecht' and 'Day.Number'. It is unclear at this time whether 'Amsterdam' is a mediator between 'Day.Number' and 'Daily.Steps' or a collider, but either way, it does not need to be considered separately. Food expenditures may also contribute, but that will be analyzed later.

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

```
acf(life_of_Julia$Daily.Steps)
```



Unsuprisingly, most of the ACF are close to 0, meaning that ‘Daily.Steps’ looks rather like white noise. This is good news because this means that the data has little sequential dependency which implies some kind of stationarity. The bad news is that this means it is difficult to highly unlikely to forecast future steps on its own (but thankfully we already confirms=ed some strong causal relations that *can* be used for forecasting).

▶ 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?

Though I hypothesized earlier that there may be a seasonality of period 7 in the data, I am not suprised that this is missing in the ACF; this seasonality is missing because my weekends in Voorburg versus away are neither predictably weekly nor predictably biweekly, thus no clear period can form (for steps alone). Once again, the lack of periods and low auto-correlation indicate some kind of stationarity.

2. Forecasting

2.1. SARIMA modeling

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

```
adf.test(life_of_Julia$Daily.Steps)
```

```
## Warning in adf.test(life_of_Julia$Daily.Steps): p-value smaller than printed p-
## value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  life_of_Julia$Daily.Steps
## Dickey-Fuller = -5.7825, Lag order = 4, p-value = 0.01
## alternative hypothesis: stationary
```

The Dickey-Fuller test outputs a p-value of less than 0.1. p=0.1 is the threshold for rejecting the NULL hypothesis, and since this value is lower, the NULL hypothesis for the presence of a unit root is rejected. This means that we can assume the data is stationary (or close to it) and proceed with analysis without further differencing on Daily.Steps.

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

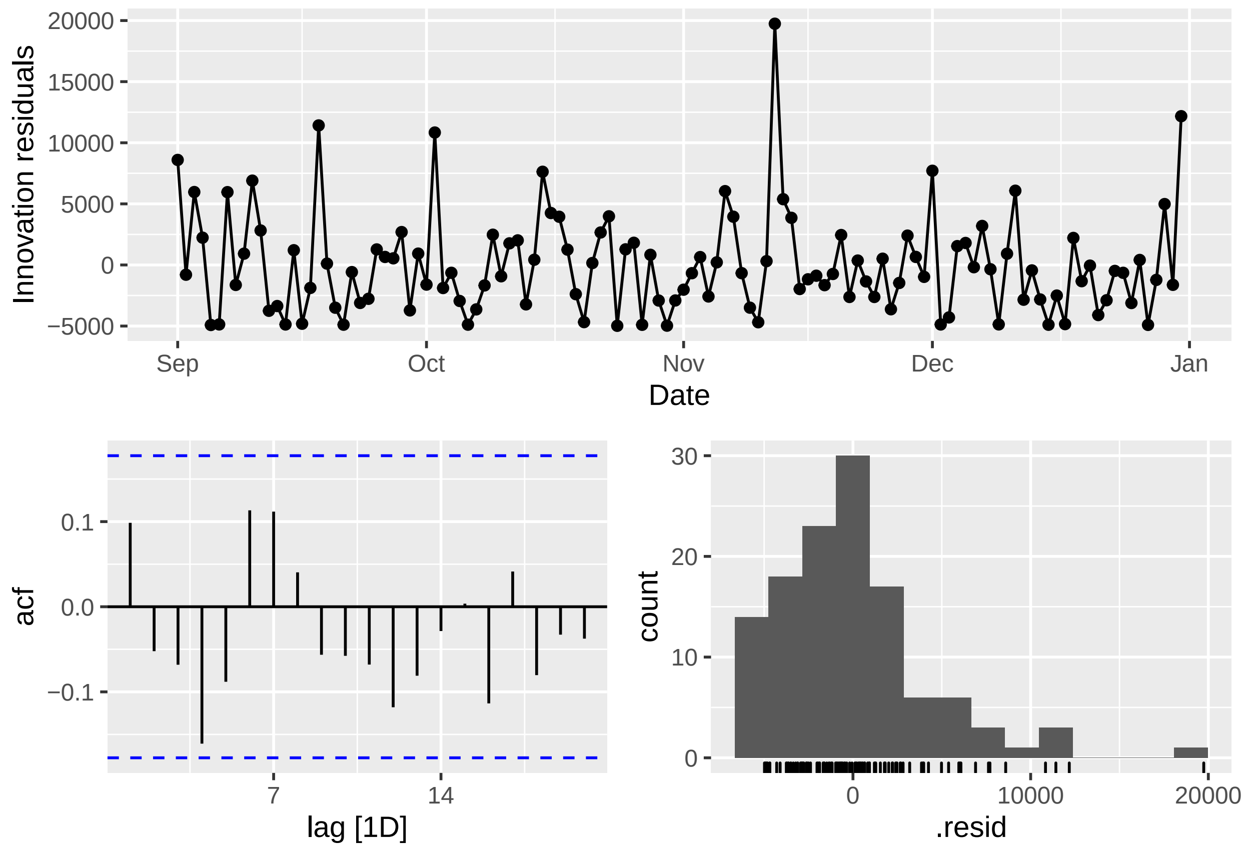
```
life_of_Julia %>% model(ARIMA(Daily.Steps)) -> steps_ARIMA
report(steps_ARIMA)
```

```
## Series: Daily.Steps
## Model: ARIMA(0,0,0) w/ mean
##
## Coefficients:
##        constant
##       4998.3115
## s.e.   371.9103
##
## sigma^2 estimated as 17014157:  log likelihood=-1188.23
## AIC=2380.46   AICc=2380.56   BIC=2386.07
```

Well....this means the steps data is pure white noise. the d=0 in ARIMA(0,0,0) was expected due to the lack of unit root. Autoregression of 0 also makes sense considering how close to zero the ACF plot was, and that it was neither consistently above nor below the origin. However, having a moving average (error term) of 0 is disappointing. Once again, this is likely due to my inconsistent weekly schedule, so not very suprising. Perhaps more insights will appear when Granger causailty is used later.

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

```
gg_tsresiduals(steps_ARIMA)
```

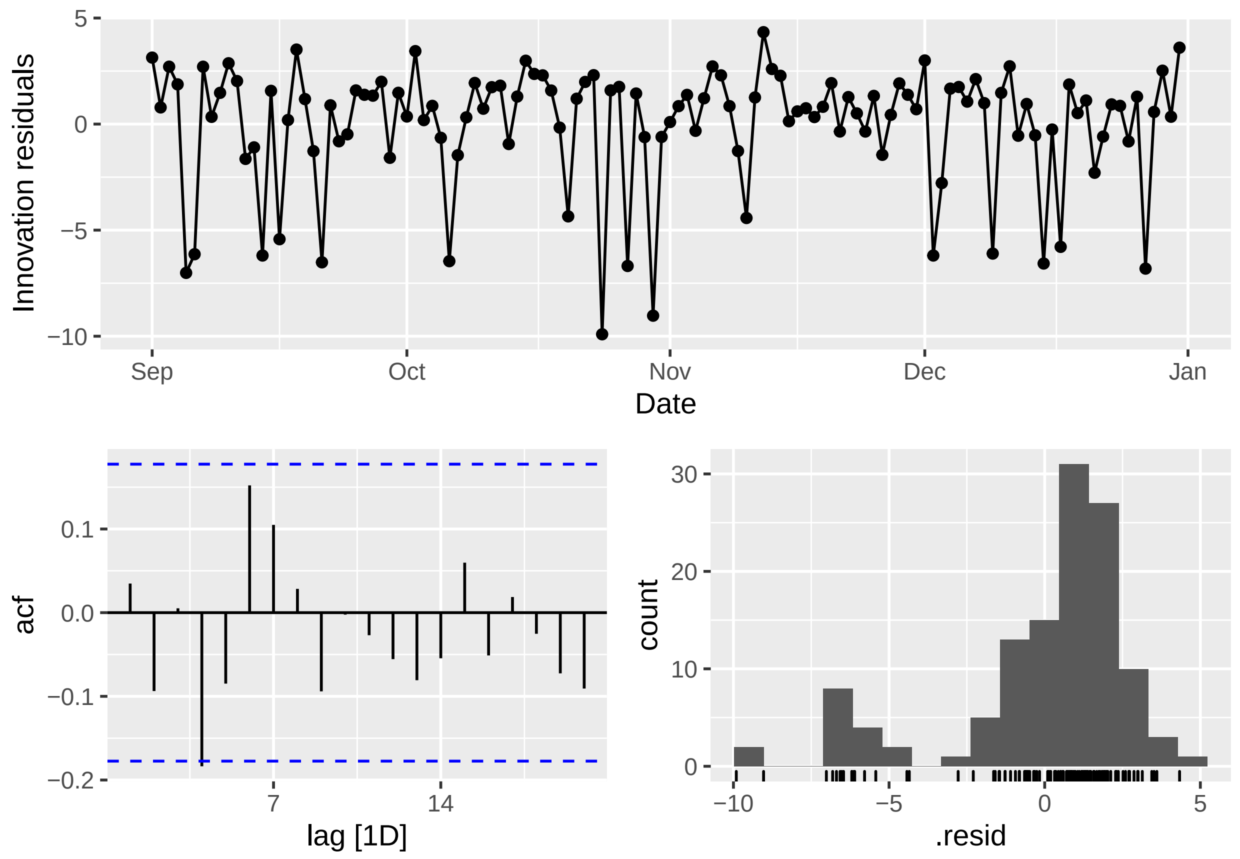


The biggest “flag” here is the non-normal, non-symmetrical residual distribution; this implies some further transformation of the data is needed in order to output anything of substance (and with conviction). The highly skewed nature of the residuals indicates that a log-transform may be beneficial:

```
logged_steps <- log(life_of_Julia$Daily.Steps, base = exp(0.5))
life_of_Julia %>%
  model(ARIMA(logged_steps)) -> steps_ARIMA_logged
report(steps_ARIMA_logged)
```

```
## Series: logged_steps
## Model: ARIMA(0,0,0) w/ mean
##
## Coefficients:
##       constant
##        15.8996
## s.e.    0.2587
##
## sigma^2 estimated as 8.232:  log likelihood=-301.2
## AIC=606.39   AICc=606.49   BIC=612
```

```
gg_tsresiduals(steps_ARIMA_logged)
```



Alas, the residuals are still distinctly non-normal after various log transforms (0.5, 5, 10). This once again confirms that the steps data is/behaves like white noise.

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?

Using the correlations explored earlier, I will attempt to create a dynamic regression model on key locations and day of week.

```
life_of_Julia %>%
  model(ARIMA(Daily.Steps ~ Day.Number)) -> weekday_regressed_ARIMA
report(weekday_regressed_ARIMA)
```

```
## Series: Daily.Steps
## Model: LM w/ ARIMA(0,0,0) errors
##
```

```
## Coefficients:
##       Day.Number  intercept
##         564.9724  2724.5289
## s.e.    180.1985   808.6743
##
## sigma^2 estimated as 15876702:  log likelihood=-1183.5
## AIC=2373.01   AICc=2373.21   BIC=2381.42
```

```
life_of_Julia %>%
  model(ARIMA(Daily.Steps ~ Voorburg)) -> location_regressed_ARIMA_V
report(location_regressed_ARIMA_V)
```

```
## Series: Daily.Steps
## Model: LM w/ ARIMA(0,0,0) errors
##
## Coefficients:
##        Voorburg1  intercept
##       -2420.4329  5930.7733
## s.e.    732.1282   454.4187
##
## sigma^2 estimated as 15745341:  log likelihood=-1183
## AIC=2372   AICc=2372.2   BIC=2380.41
```

```
life_of_Julia %>%
  model(ARIMA(Daily.Steps ~ Amsterdam)) -> location_regressed_ARIMA_A
report(location_regressed_ARIMA_A)
```

```
## Series: Daily.Steps
## Model: LM w/ ARIMA(0,0,0) errors
##
## Coefficients:
##       Amsterdam1  intercept
##        3677.8825   4455.673
## s.e.    994.4162    381.966
##
## sigma^2 estimated as 15426283:  log likelihood=-1181.75
## AIC=2369.5   AICc=2369.7   BIC=2377.91
```

```
life_of_Julia %>%
  model(ARIMA(Daily.Steps ~ Utrecht)) -> location_regressed_ARIMA_U
report(location_regressed_ARIMA_U)
```

```
## Series: Daily.Steps
## Model: LM w/ ARIMA(0,0,0)(0,0,1)[7] errors
##
## Coefficients:
##         sma1   Utrecht1  intercept
##       0.2208  1843.0658  4550.8864
## s.e.  0.1016   892.4539   494.0234
##
## sigma^2 estimated as 16401296:  log likelihood=-1185.15
## AIC=2378.3   AICc=2378.65   BIC=2389.52
```

Unfortunately, Voorburg, Amsterdam, and days of the week are all insubstantial in helping predict daily steps. However! Utrecht does have some relevance–using this variable helps us figure out a weekly seasonal pattern (with a seasonal moving average component = 1, and everything else 0).

Now let's test the remaining possible predictors: number of transit stops, essential food purchases, and nonessential food purchases.

```
life_of_Julia %>%
  model(ARIMA(Daily.Steps ~ Transit.Count)) -> transit_regressed_ARIMA
report(transit_regressed_ARIMA)
```

```
## Series: Daily.Steps
## Model: LM w/ ARIMA(0,0,0) errors
##
## Coefficients:
##       Transit.Count  intercept
##            970.2199   3113.540
## s.e.       146.7997    427.991
##
## sigma^2 estimated as 12632880:  log likelihood=-1169.56
## AIC=2345.13   AICc=2345.33   BIC=2353.54
```

```
life_of_Julia %>%
  model(ARIMA(Daily.Steps ~ Essential.Food)) -> essential_food_regressed_ARIMA
report(essential_food_regressed_ARIMA)
```

```
## Series: Daily.Steps
## Model: LM w/ ARIMA(0,0,0) errors
##
## Coefficients:
##       Essential.Food  intercept
##             -33.3274  5195.7625
## s.e.         39.5804   438.7558
##
## sigma^2 estimated as 17056817:  log likelihood=-1187.88
## AIC=2381.76   AICc=2381.96   BIC=2390.17
```

```
life_of_Julia %>%
  model(ARIMA(Daily.Steps ~ Nonessential.Food)) -> nonessential_food_regressed_ARIMA
report(nonessential_food_regressed_ARIMA)
```

```
## Series: Daily.Steps
## Model: LM w/ ARIMA(0,0,0) errors
##
## Coefficients:
##       Nonessential.Food  intercept
##                132.8723  4492.8522
## s.e.            44.6834   397.3194
##
## sigma^2 estimated as 1.6e+07:  log likelihood=-1183.96
## AIC=2373.93   AICc=2374.13   BIC=2382.34
```

Though I got ARIMA(0,0,0) for ‘Nonessential.Food’, when I ran it earlier in conjuction with weekdays I got ARIMA(0,0,0)(0,0,1)[7]. This means that the seasonality could be tenuous, but I will include it in the aggregate model as a potential predictor anyways.

All regressed variables which yielded the same ARIMA model as the original aurocorrelation (ARIMA(0,0,0)) do not add to the prediction and can thus be ignored. Instead, let’s take a closer look at the residuals when considering ‘Utrecht’ and ‘Nonessential.Food’ (together) below:

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

Both ‘Daily.Steps’ ~ ‘Utrecht’ and ‘Daily.Steps’ ~ ‘Nonessential.Food’ have order SARIMA(0,0,0)(0,0,1)[7], meaning that the previous week’s residuals are the only discernible effect on a particular day’s steps.

The strongest predictors will be combined into one new regression model:

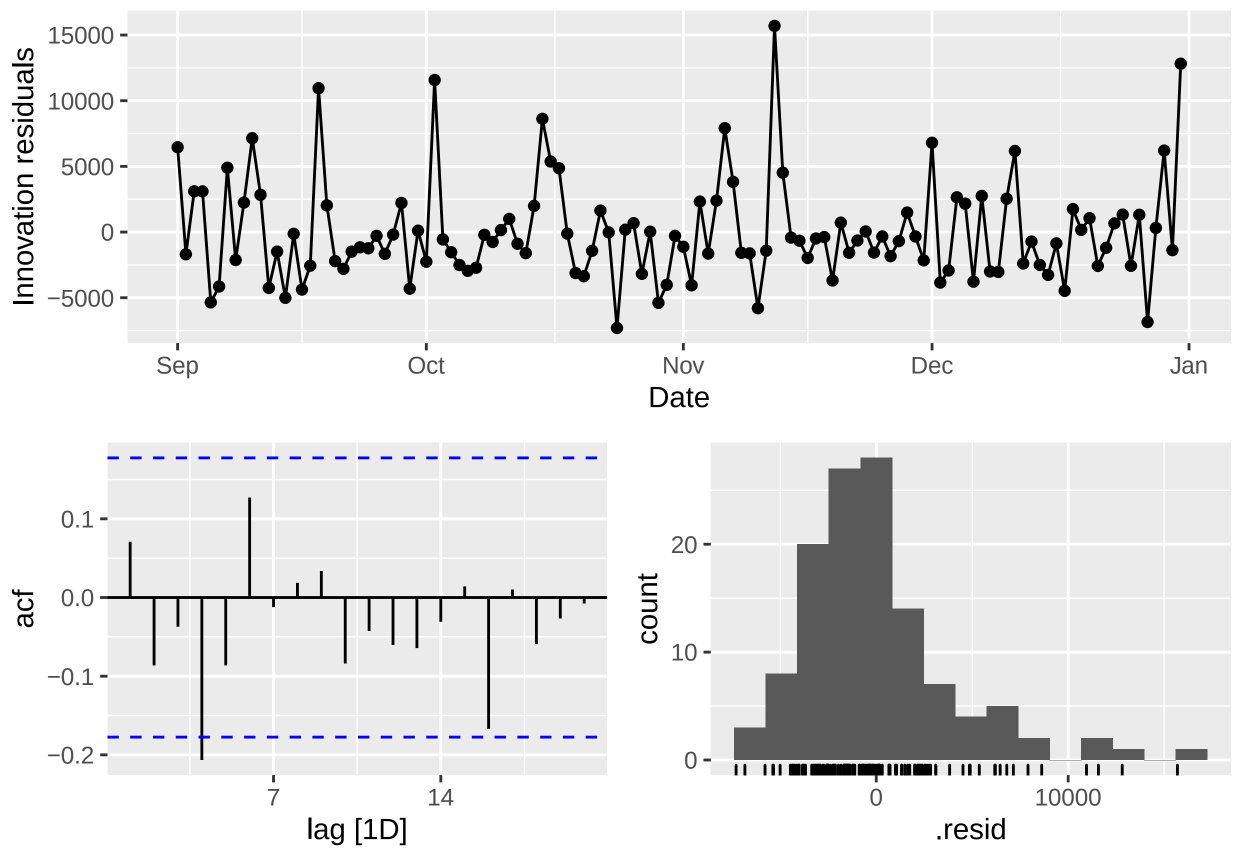
```
life_of_Julia %>%
  model(ARIMA(Daily.Steps ~ Utrecht + Nonessential.Food)) -> regressed_ARIMA
report(regressed_ARIMA)
```

```
## Series: Daily.Steps
## Model: LM w/ ARIMA(0,0,0)(0,0,1)[7] errors
##
## Coefficients:
```

```
##          sma1   Utrecht1  Nonessential.Food  intercept
##        0.2288  1881.3606           127.0375  4052.1252
## s.e.   0.1055   848.1481            41.6341   502.8799
##
## sigma^2 estimated as 15357751:  log likelihood=-1180.64
## AIC=2371.28   AICc=2371.8   BIC=2385.3
```

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

```
gg_tsresiduals(regressed_ARIMA)
```



Unfortunately, we run into the same problem as the original model: the residuals are non-normal and no amount of log transformations will help it. Both 'Utrecht' and 'Nonessential.Food' (and of course, the combined regression model) have similar innovation residuals, but unlike the original, are entirely in the positive. Still, there is no obvious trend, so there is nothing further we can do.

2.3. Forecasts

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

Unfortunately, due to the highly unstructured nature of the data, any forecasts given from steps alone will be poor and generally close to the mean. However, since some weekly seasonality was pickup up on when considering predictor variables, a forecast of at least one week is reasonable (meaning one forecast value per day of week), but it's best to forecast several periods since my behavior is not consistent from week to week. For this reason, a 1-month test 'forecast' period has been chosen.

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

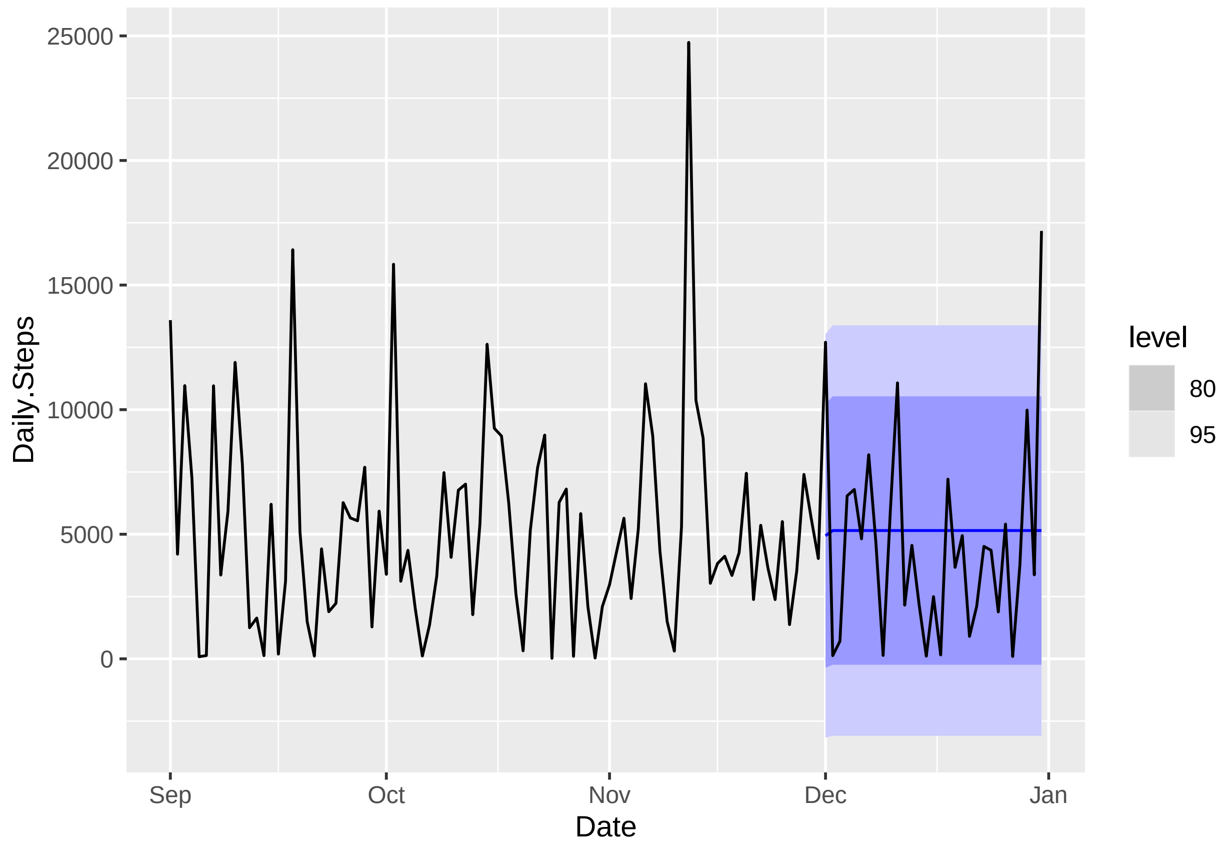
```
# split 'life_of_Julia' into test and training sets -- this requires us to return to t
# split with 1 month in the test set (roughly 4 seasonal periods)
life_of_Julia_train <-
  life_of_Julia_df[life_of_Julia_df$Date < ymd('2021-12-01'), ]
life_of_Julia_train <-
  as_tsibble(life_of_Julia_train, index = Date, key=NULL)

life_of_Julia_test <-
  life_of_Julia_df[life_of_Julia_df$Date >= ymd('2021-12-01'), ]
life_of_Julia_test <-
  as_tsibble(life_of_Julia_test, index = Date, key=NULL)

# retrain ARIMA on training set
life_of_Julia_train %>%
  model(ARIMA(Daily.Steps)) -> steps_ARIMA_training
report(steps_ARIMA_training)
```

```
## Series: Daily.Steps
## Model: ARIMA(0,0,1) w/ mean
##
## Coefficients:
##          ma1   constant
##       0.1847  5148.5548
## s.e.  0.1107   506.7923
##
## sigma^2 estimated as 17078594:  log likelihood=-885.86
## AIC=1777.71   AICc=1777.99   BIC=1785.25
```

```
# forecast using test set and compare with actual values
prediction <-
  forecast(steps_ARIMA_training, new_data = life_of_Julia_test)
autoplot(prediction, life_of_Julia)
```



Without any predictors, the daily steps behave similarly to white noise, so forecast values are expected to be given as the mean value of the set, with an acceptable range of plus or minus a ~5000 step interval rectangle. This is not a *bad* prediction because nearly all daily steps fall in the 80% confidence interval, but it's not a very *smart* prediction.

NOTE: this is the raw steps data, not the log-transformed data.

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

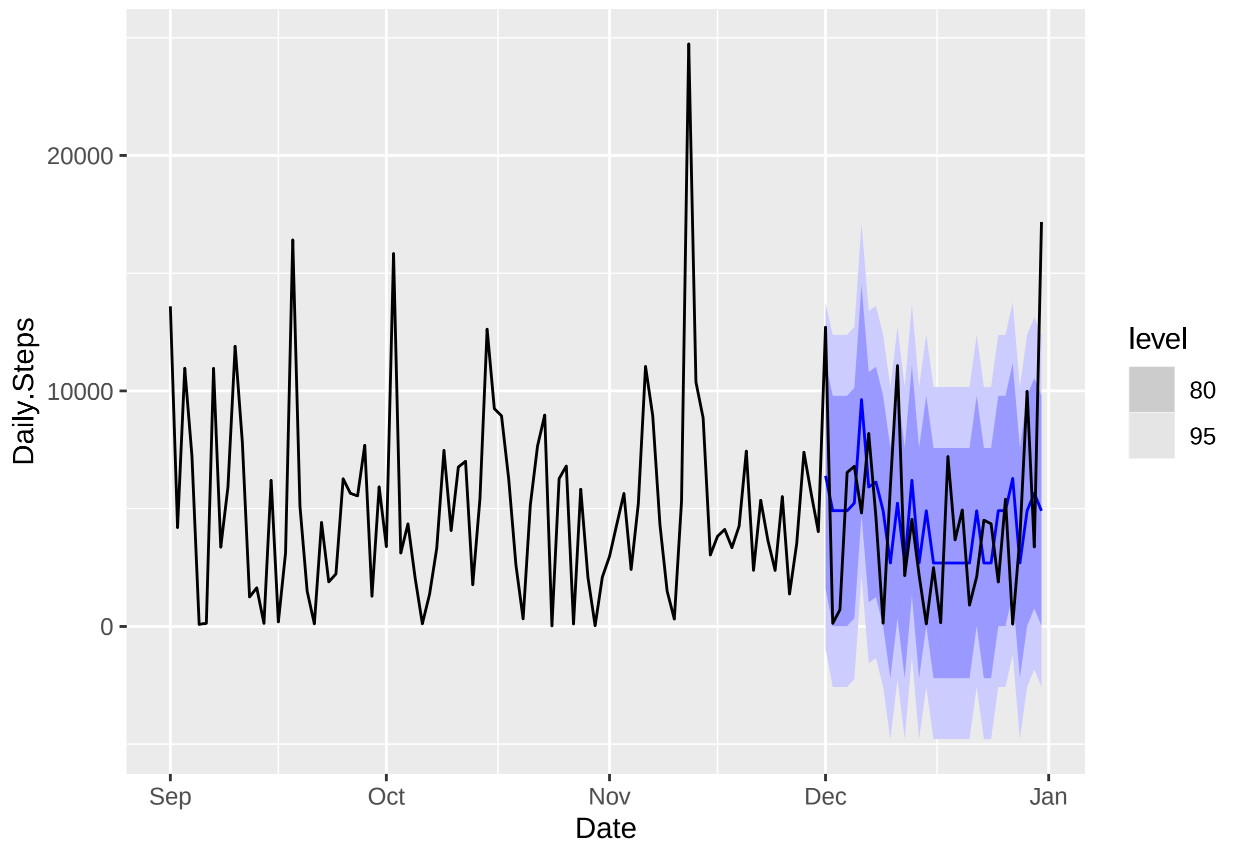
Let us use 'Voorburg', 'Utrecht', and 'Nonessential.Food' as the predictor variables for this example. For these to be valid predictors, we need to split the original data set into train and test datasets, else there is no predictor variable data to forecast on.

```
# retrain ARIMA on training set
life_of_Julia_train %>%
  model(ARIMA(Daily.Steps ~ Utrecht + Voorburg + Nonessential.Food)) ->
  regressed_ARIMA_training
report(regressed_ARIMA_training)
```

```
## Series: Daily.Steps
## Model: LM w/ ARIMA(0,0,1) errors
##
## Coefficients:
```

```
##          ma1   Utrecht1   Voorburg1  Nonessential.Food  intercept
##       0.1973  1007.5312  -2218.1307           172.3483  4909.3232
## s.e.  0.1251   904.1812    825.7313            42.7003   673.8413
##
## sigma^2 estimated as 14016246:  log likelihood=-875.31
## AIC=1762.62   AICc=1763.62   BIC=1777.68
```

```r
# forecast using test set and compare with actual values
prediction_variables <-
  forecast(regressed_ARIMA_training, new_data = life_of_Julia_test)
autoplot(prediction_variables, life_of_Julia)
```



Unlike the previous forecast, this regression model features fluctuations in predicted value, though the peaks in fluctuations do not always match up with the actual step values. This make regression a *smarter* model, but not a *better* one.

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

The regression forecast has a more plausible outcome as the values differ by day and do bear some resemblance to the actual steps taken. For a short term forecast, though, there is no discernible difference in accuracy between the two models.

3. Causal Modeling

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

From the original hypothesis, two causal questions stick out: 1) Did my choosing to track steps lead to an increased average daily steps? 2) Does my being out of the house cause me to spend more money on food?

▶ 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?

Question 1 is a simple interrupted time series: we consider the variable of 'Daily.Steps' before and after the intervention date of 16 November 2021 (deciding to track steps). The intervention should only affect steps and not travel nor expenditure because those variable are two of the "fundamental constants in the life of Julia" stated in the intro.

Question 2 can be answered with Granger Causality. Nonessential spending occurs same-day as location, so there's no lag to consider there; however, Essential spending may have a 1-day lag with respect to Voorburg = 0 because I try to empty my fridge before being away, and an empty fridge is not systainable to a hungry Julia, so essential grocery shopping must occur soon after.

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.

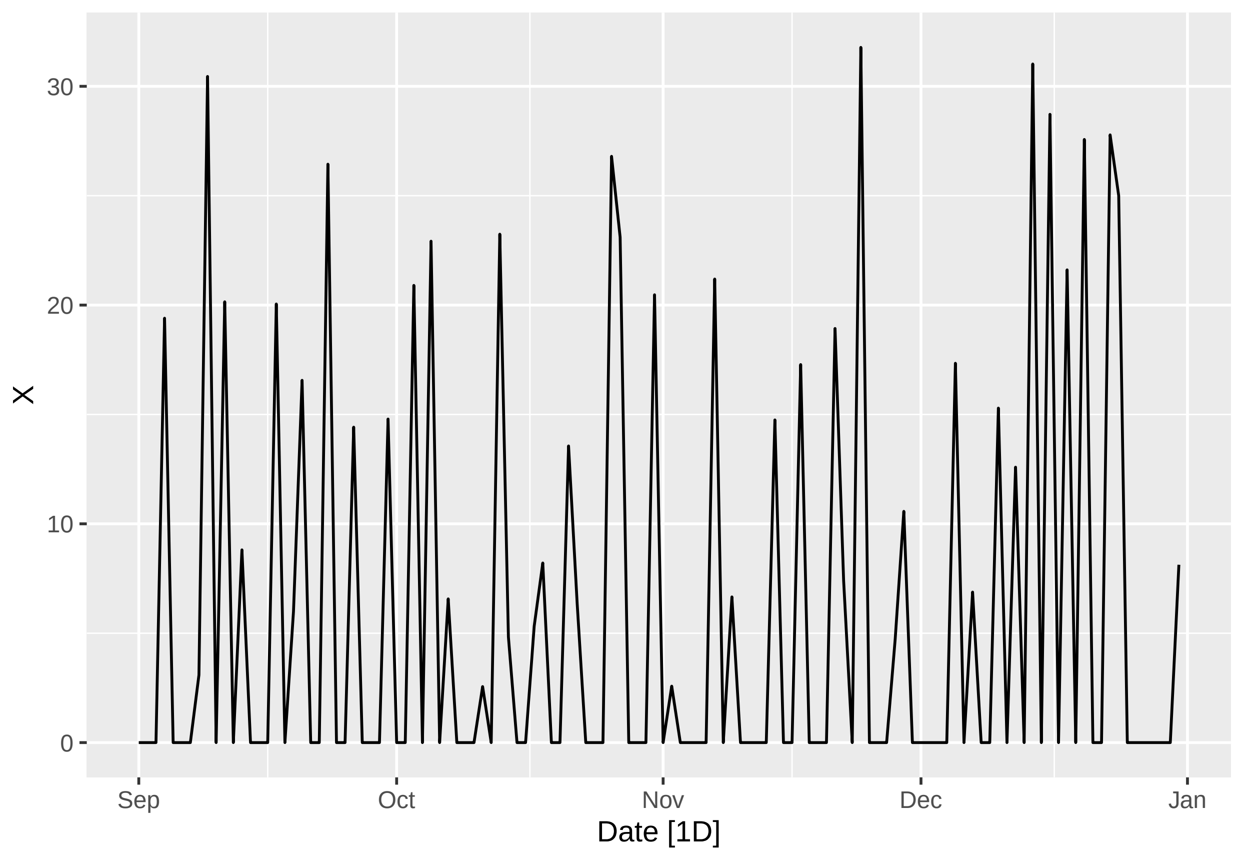
3.2a Granger Causal analysis

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

Let's consider the second question first: Does my being out of the house cause me to spend more money on food?

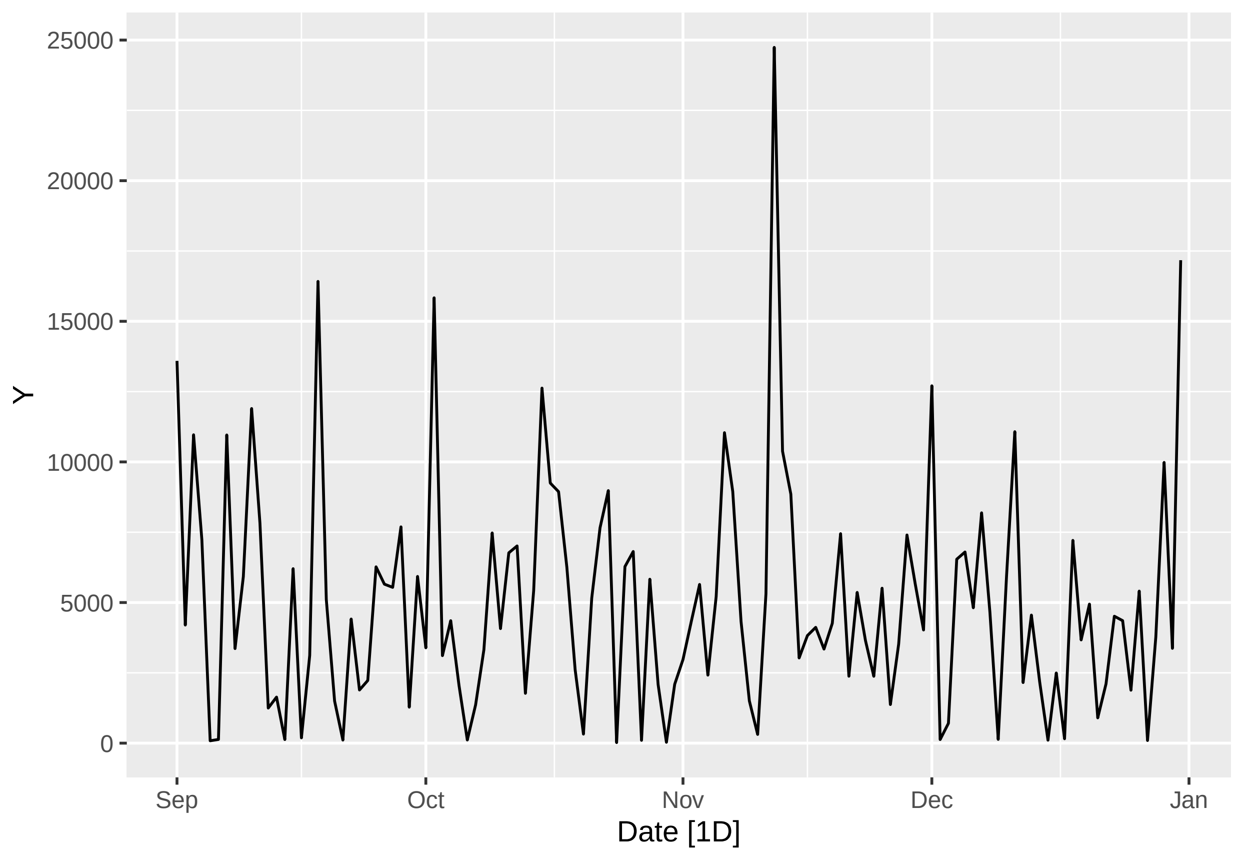
For an overview, let's plot X = Essential.Food:

```
X = life_of_Julia$Essential.Food
autoplot(life_of_Julia, X)
```



And once again plot Y = Daily.Steps:

```
Y = life_of_Julia$Daily.Steps
autoplot(life_of_Julia, Y)
```



▶ 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.

The steps causal model in section 2 considered location (Utrecht) and nonessential food purchases as covariates. Since this already excludes X (essential food purchases), we can reuse the ARIMA model from before–p, d, and q are all 0 except for a seasonal Q=1 and period of 7, indicating that noise from the previous week has an effect on today's steps.

```
#life_of_Julia %>%
# model(ARIMA(Daily.Steps ~ Utrecht + Nonessential.Food)) ->
# regressed_ARIMA
report(regressed_ARIMA)
```

```
## Series: Daily.Steps
## Model: LM w/ ARIMA(0,0,0)(0,0,1)[7] errors
##
## Coefficients:
##          sma1   Utrecht1  Nonessential.Food  intercept
##        0.2288  1881.3606           127.0375  4052.1252
## s.e.   0.1055   848.1481            41.6341   502.8799
```

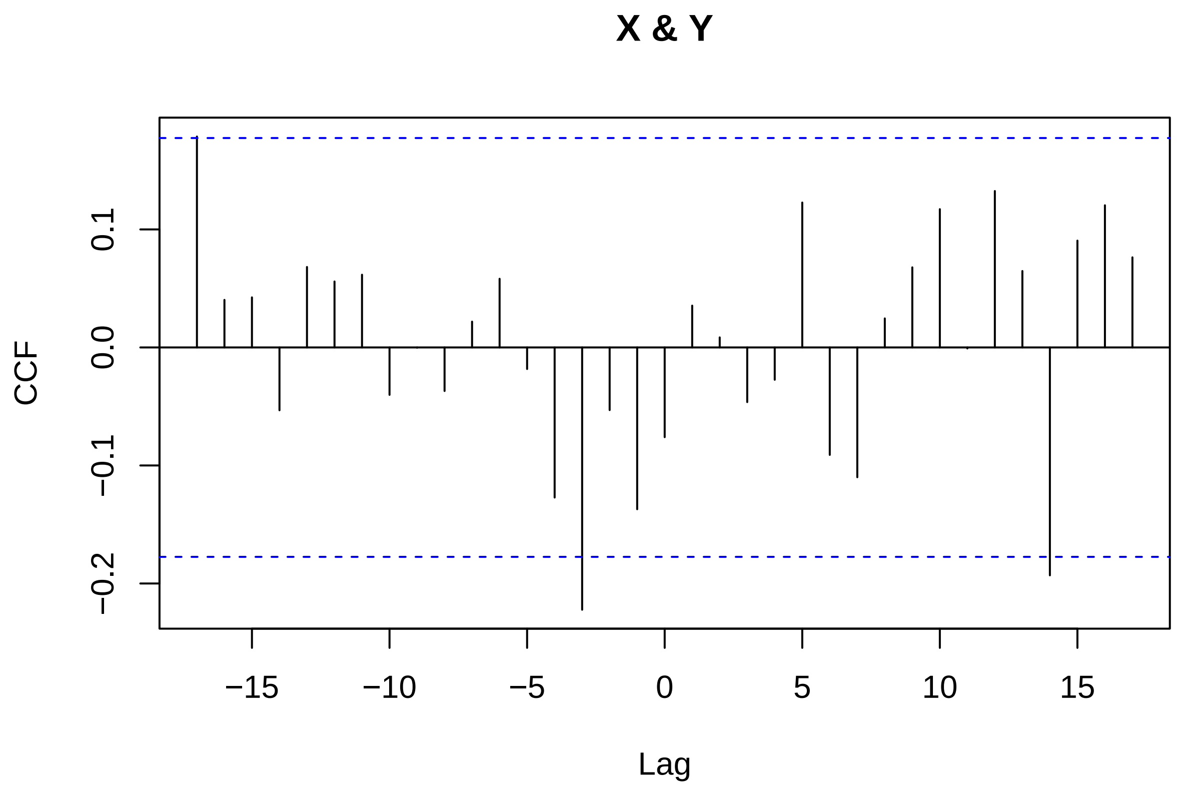
```
##
## sigma^2 estimated as 15357751:  log likelihood=-1180.64
## AIC=2371.28   AICc=2371.8   BIC=2385.3
```

▶ 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.

Because the SARIMA model for the base model (Daily.Steps regressed on Utrecht and Nonessential.Food) has a seasonal component that depends on residual values from one week prior, lag -7 must definitely be considered (and possibly lag -14 since events are somewhat bi-weekly). And because Essential.Food is hypothesized as actually following the travel (and thus increased daily steps), a positive lag must also be considered.

Let's try CCF to see for sure:

```
ccf(X, Y, ylab = "CCF")
```



Surprisingly, lag =-3 is the strongest predictor. In a way, this makes sense because I'm not doing much grocery shopping before I leave for a trip or a weekend.

Lag = 14 is also a contender, but this can be ignored as the SARIMA model for Y already has a seasonality of 7. Lag = -17 can also be ignored for this reason, as lag -3 is 2 seasonal periods from lag -17.

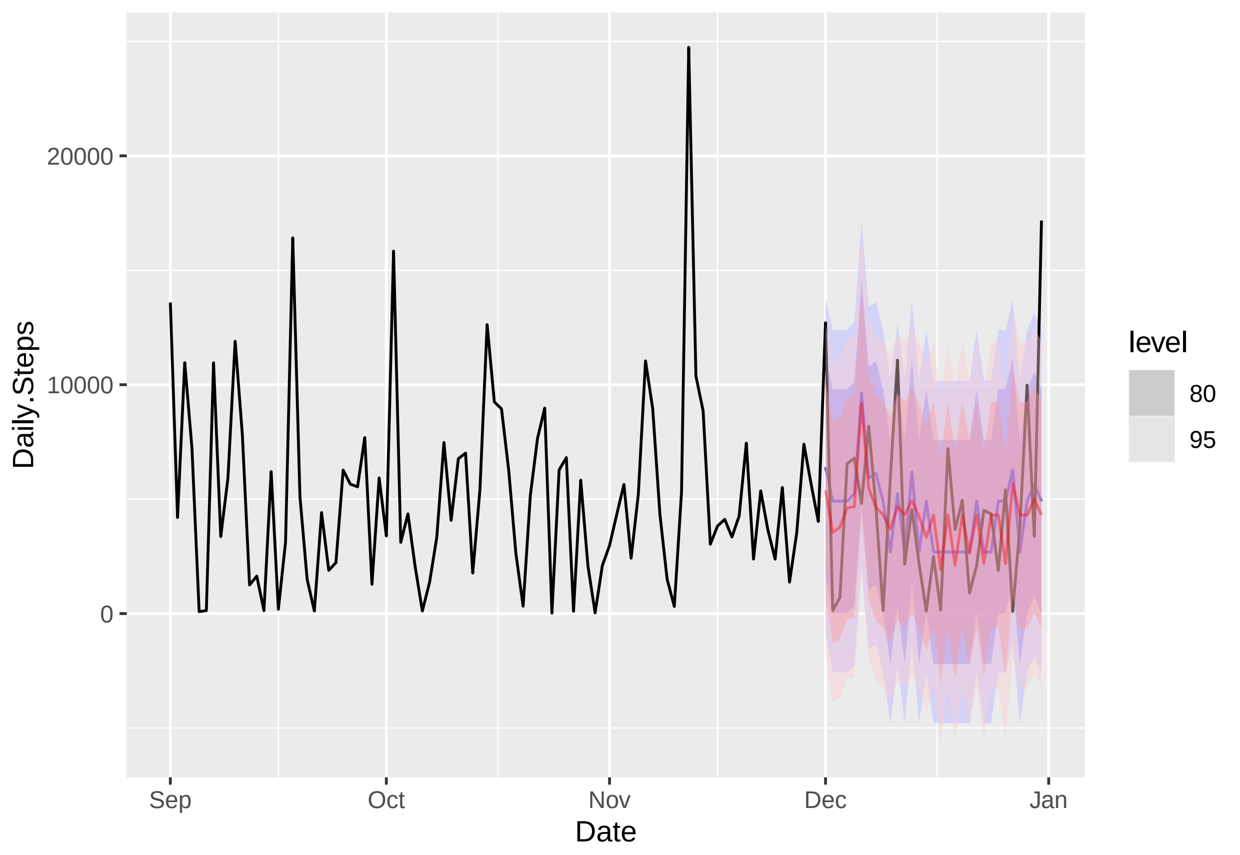
▶ Investigate whether adding your lagged “cause’’ variables (X) improve the prediction of your effect variable(s) Y. Use model selection based on information criteria. Describe your final chosen model

```
# incorporate the Essential.Food lag 3 days behind Daily.Steps (training data)
life_of_Julia_train %>%
  model(ARIMA(Daily.Steps ~ Utrecht + Nonessential.Food + lag(Essential.Food,3))) ->
  regressed_ARIMA_lagged
report(regressed_ARIMA_lagged)
```

```
## Series: Daily.Steps
## Model: LM w/ ARIMA(0,0,1)(1,0,0)[7] errors
##
## Coefficients:
##          ma1    sar1   Utrecht1  Nonessential.Food  lag(Essential.Food, 3)
##       0.2238  0.1821  1547.9966           158.1889                -76.4790
## s.e.  0.1251  0.1141   951.1626            43.3011                 43.6366
##       intercept
##       4309.3171
## s.e.   709.5613
##
## sigma^2 estimated as 13687339:  log likelihood=-846.49
## AIC=1706.98   AICc=1708.33   BIC=1724.56
```

```
# plot original forecast and overlay new forecast
lagged_prediction <-
  forecast(regressed_ARIMA_lagged, new_data = life_of_Julia_test)

autoplot(prediction_variables, life_of_Julia, alpha = 0.75, color = 'blue') +
  autolayer(lagged_prediction, life_of_Julia, alpha = 0.4, color = 'red')
```



Blue shows the original prediction just using ‘Utrecht’ and ‘Nonessential.Food’ while Red shows the model after including ‘Essential.Food’ with lag = -3. As you can see, the new model with lag does a MUCH better job emulating steps taken, correctly accounting for daily increases or decreases.

Granger causality would lead to the incorrect conclusion that buying less groceries causes me to be really active three days later. The logical interpretation of this would be that going on a trip causes me to put off grocery shopping. However, since the future cannot cause the past, this indicates that there is an unobserved confounder in effect–the act of planning to go away. That means that while there is no causation from food spending, a third party (someone monitoring my debit card charges, for instance) could use the principle of Granger Causality to predict when I will be gone for a few days, and thus when I will increase my daily steps.

3.2b Interrupted Time Series analysis

▶ Partition your dataset into pre- and post- intervention time periods and visualize this. Describe what the intervention is and when it takes place

Now we can consider the first question: Did my choosing to track steps lead to an increased average daily steps?

To do this, the data must be split into 2 sets: pre- and post- 16 November 2021 (date of intervention).

```
# we go back to the original tibble to create a new time series division
life_of_Julia_pre <-
  life_of_Julia_df[life_of_Julia_df$Date < ymd('2021-11-16'), ]
life_of_Julia_pre <-
  as_tsibble(life_of_Julia_pre, index = Date, key=NULL)

life_of_Julia_post <-
  life_of_Julia_df[life_of_Julia_df$Date >= ymd('2021-11-16'), ]
life_of_Julia_post <-
  as_tsibble(life_of_Julia_post, index = Date, key=NULL)
```

▶ Train an appropriate ARIMA model on pre-intervention data of your outcome variable Y. If using the same model as fit in part 2, briefly describe that model again here.

Once again, we use 'Utrecht' and 'Nonessential.Food' to predict 'Daily.Steps'. The ARIMA model must be retrained on the pre-intervention dataset:

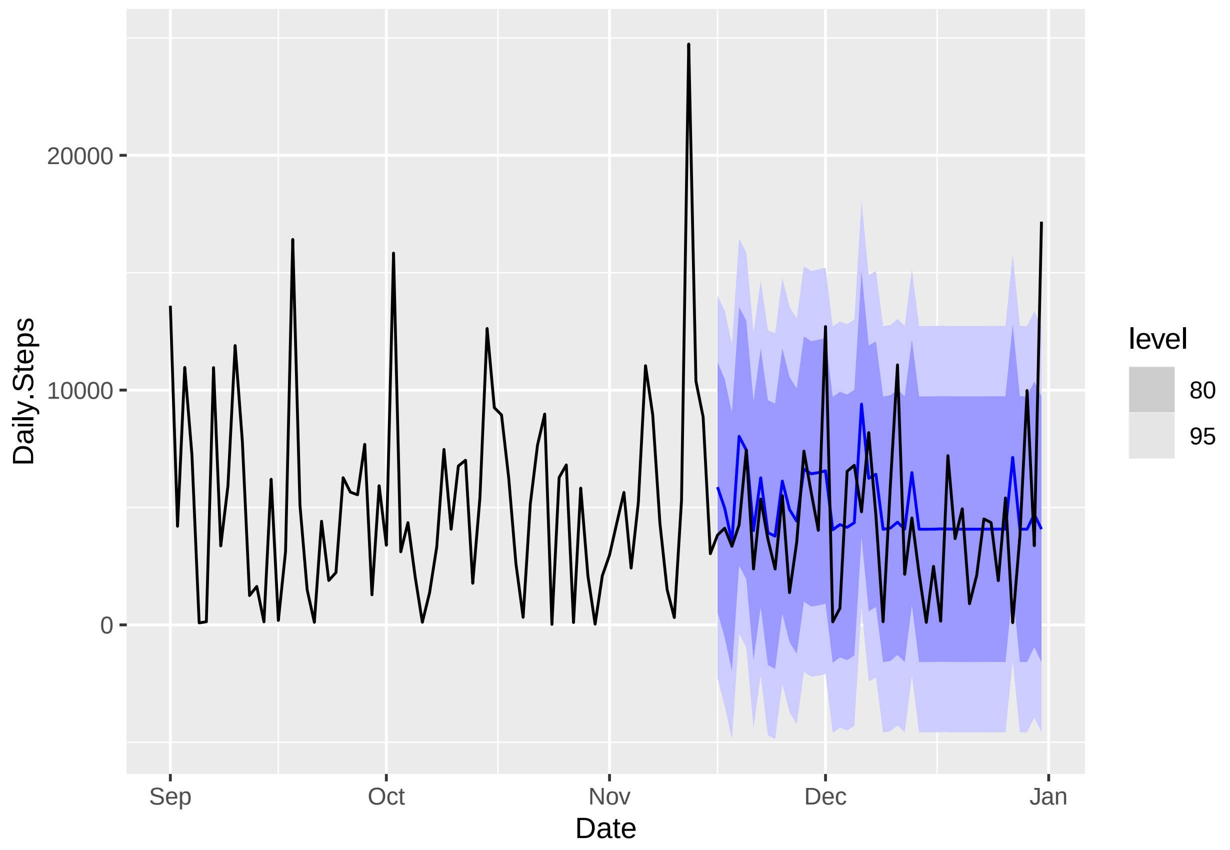
```
life_of_Julia_pre %>%
  model(ARIMA(Daily.Steps ~ Utrecht + Nonessential.Food)) ->
  preintervention_ARIMA
report(preintervention_ARIMA)
```

```
## Series: Daily.Steps
## Model: LM w/ ARIMA(0,0,1)(1,0,0)[7] errors
##
## Coefficients:
##          ma1    sar1  Utrecht1  Nonessential.Food  intercept
##       0.2587  0.2321  2171.971           146.9275  4076.5674
## s.e.  0.1322  0.1369  1153.863            46.9457   836.1334
##
## sigma^2 estimated as 17281212:  log likelihood=-738.76
## AIC=1489.51   AICc=1490.73   BIC=1503.5
```

Interestingly enough, the ARIMA order has changed from ARIMA(0,0,0)(0,0,1)[7] for the full dataset to ARIMA(0,0,0)(0,1,0)[7] for the pre-intervention data. This indicates that there is a stronger weekly pattern (differencing on one period earlier) in the pre-intervention dataset.

▶ Use this model to create forecasts for the post-intervention time period. Visualize your forecasts (both point predictions and intervals) and the observed post-intervention data in a single plot.

```
preintervention_prediction <-
  forecast(preintervention_ARIMA, new_data = life_of_Julia_post)
autoplot(preintervention_prediction, life_of_Julia)
```



Shockingly, the pre-intervention SARIMA model does a pretty job predicting post-intervention steps, with many of the peaks relatively close to actual values–except for the last half of December. The reason for this is probably that I did a lot more Grocery shopping than usual to prepare for making special holiday meals for vairous groups of people. This meant that for this half of the month the relation between spending and steps was broken. If we were to analyze data from a larger time period, it would likely become apparent that this two-week holiday period is a collective outlier in every sense, and should be disregarded in assessing model accuracy.

▶ Compare your forecasts (both point predictions and intervals) to the observed post-intervention data. Describe if and how these differ from one another.

The verdict is that deciding to log my steps did NOT lead to an increase in daily steps. If anything, you could actually say that I walked *less* after the intervention, which would imply that the act of tracking was not an intervention at all, and that instead the changes could be attributed to something else (like a new course schedule in the new quarter).

3.2c Synthetic Control Analysis

Not applicable for this data set.

3.3 Conclusion and critical reflection

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

In summary:

1) Did my choosing to track steps lead to an increased average daily steps?
   Not in the slightest.

2) Does my being out of the house cause me to spend more money on food?
   No! But there is a different causal relation between the two variables: knowing I will be out of the house in the future, I spend less on essential food (so a decrease in essential food spending is an early indication for increased steps 3 days later).

▶ 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)

The basic assumption underlying all of these analysis is that I predictably and consistently carry my phone with me based on some sort of a pattern of dates or rules regarding location. However, we saw that this wasn't always true, making my measurement of daily steps erratic and equivalent to white noise. Much better associations could be made (and extended to non-Julia people) if there were a more reliable method of tracking steps such as a smart-watch worn at all hours.
