Flights

Jonathan Fung

07/12/22

Contents

Import

Source: tidytuesday - July 12, 2022

```
library(tidyverse)
library(fpp3)
tuesdata <- tidytuesdayR::tt_load('2022-07-12')
flights <- tuesdata$flights

names(flights)
```

YEAR
MONTH_NUM
MONTH_MON
FLT_DATE
APT_ICAO
APT_NAME
STATE_NAME
FLT_DEP_1
FLT_ARR_1
FLT_TOT_1
FLT_DEP_IFR_2
FLT_ARR_IFR_2
FLT_TOT_IFR_2
Pivot Label

This dataset is a daily time series on each airport, with each record having total IFR movement, departures, and arrivals, from both "Network Manager" (1) and "Airport Operator" (2).

```
head(flights[,1:8])
```

2016	1	JAN	2016-01-01	EBAW	Antwerp	Belgium	4
2016	1	JAN	2016-01-01	EBBR	Brussels	Belgium	174
2016	1	JAN	2016-01-01	EBCI	Charleroi	Belgium	45
2016	1	JAN	2016-01-01	EBLG	Liège	Belgium	6
2016	1	JAN	2016-01-01	EBOS	Ostend-Bruges	Belgium	7
2016	1	JAN	2016-01-01	EDDB	Berlin - Brandenburg	Germany	98

Utils

Define `assignRegion`, using the CIA - The World Factbook to assign European regions to states. Also define `cleanState`, which cleans up some state names to work with the `rnaturalearth` package.

```r
assignRegion <- function (s) {
  region <- case_when(
s == "Belgium" ~ "Western Europe",
#...
s == "Israel" ~ "Not Europe" # Middle East
              )
  return(region)
}


# Convert STATE_NAMEs format in flights to work with rnaturalearth
cleanState <- function (s) {
  state <- case_when(
s == "Bosnia and Herzegovina" ~ "Bosnia and Herz.",
#...
s == "Türkiye" ~ "Turkey",
TRUE ~ s
              )
  return(state)
}
```

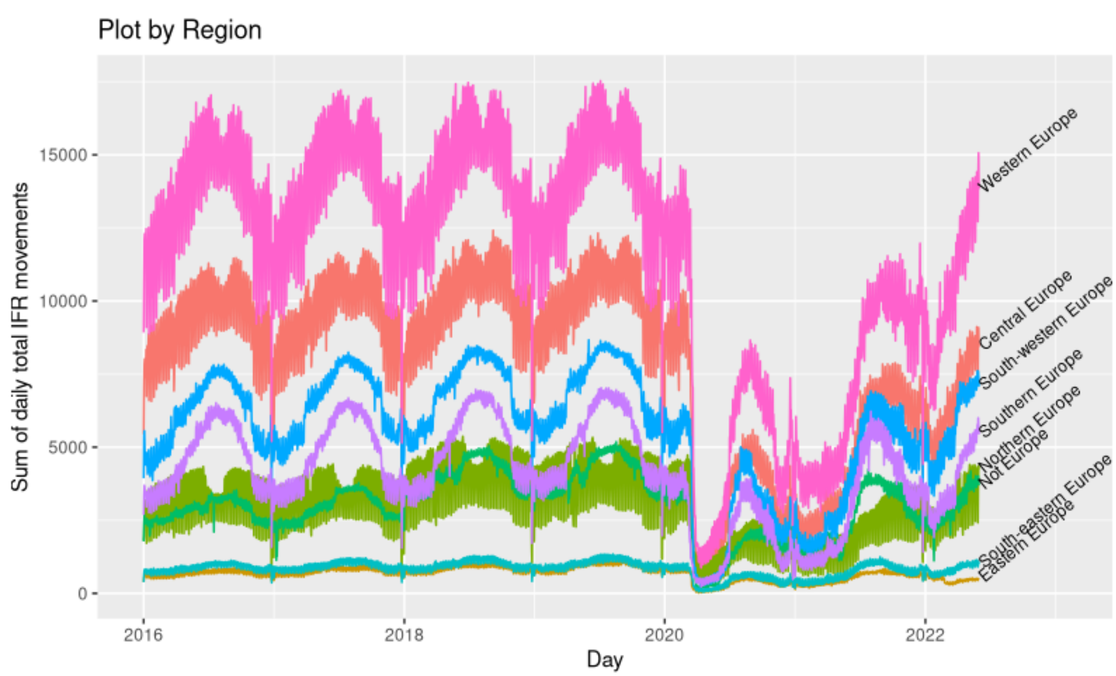
```r
assignRegion("Netherlands")
cleanState("Czech Republic")
```

```
[1] "Western Europe"
[1] "Czech Rep."
```

Explore

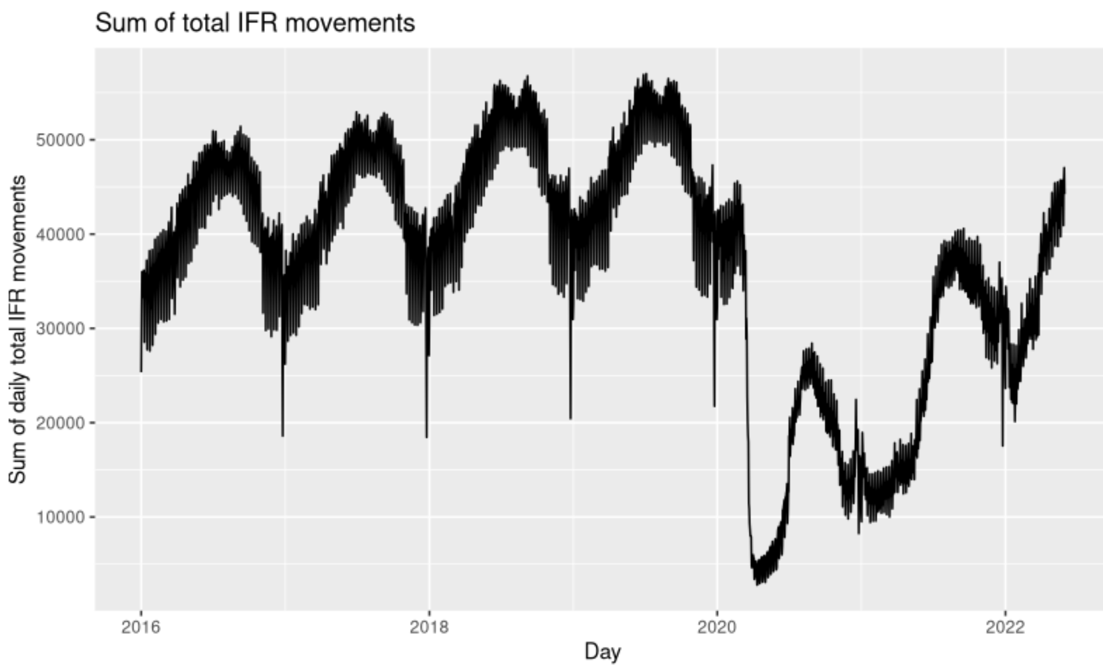
The huge dip in the beginning of 2020 is when the COVID-19 Pandemic lockdowns started to hit the world. Before that, we see a pretty consistent seasonality. After, there is still seasonality, but with significant growth trends. The order across regions also persists during the pandemic.

```
by_region <- flights %>%
  mutate(region = assignRegion(STATE_NAME)) %>%
  group_by(FLT_DATE, region) %>%
  summarise(tot = sum(FLT_TOT_1), .groups = "keep") %>%
  mutate(FLT_DATE = as.Date(FLT_DATE)) %>%
  as_tsibble(key = region, index = FLT_DATE)

## https://dcl-data-vis.stanford.edu/time-series.html#one-response-variable
by_region %>% autoplot(tot) +
  geom_text(aes(label = region),
            data = by_region %>% filter(FLT_DATE == "2022-05-31"),
            color = "black",
            hjust = 0,
            size = 3,
            nudge_x = 5,
            angle = 40) +
  xlab("Day") + ylab("Sum of daily total IFR movements") +
  ggtitle("Plot by Region") +
  scale_x_date(limits = as.Date(c("2016-01-01", "2023-02-01"))) +
  theme(legend.text=element_text(size=6),
        legend.position="none")
```



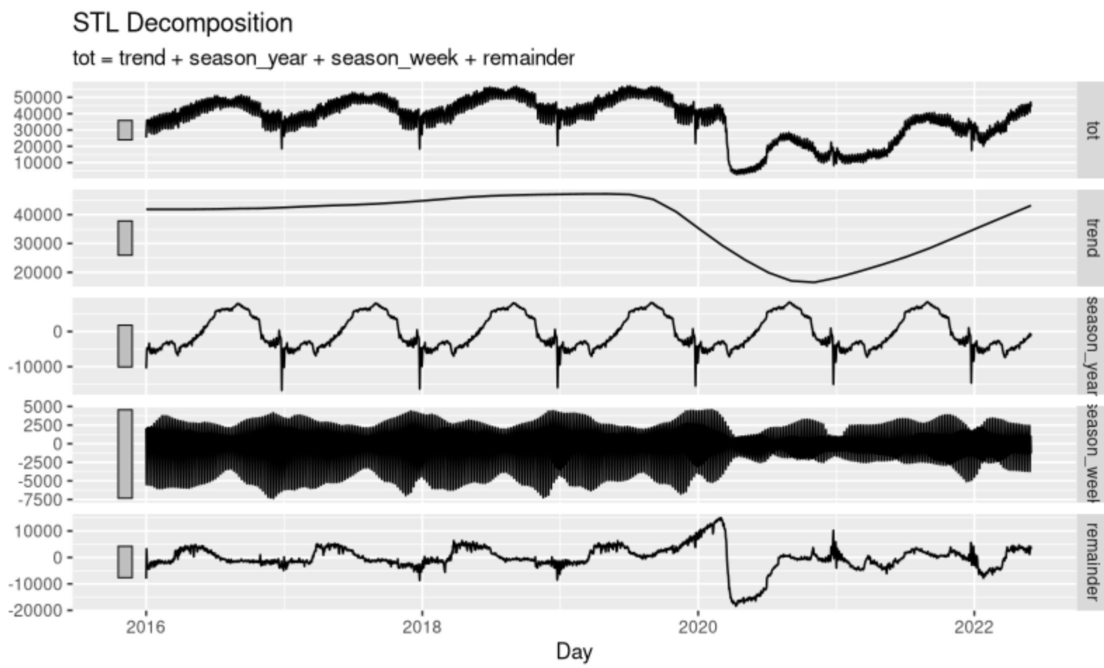
For clarity, we can view the same data, but summed over regions.

```
TOT_sum <- flights %>%
  group_by(FLT_DATE) %>%
  summarise(tot = sum(FLT_TOT_1)) %>%
  mutate(FLT_DATE = as.Date(FLT_DATE)) %>%
  as_tsibble(index = FLT_DATE)

TOT_sum %>% autoplot(tot) +
  xlab("Day") + ylab("Sum of daily total IFR movements") +
  ggtitle("Sum of total IFR movements")
```



STL Decomposition clearly shows that this data exhibits trend, year-seasonality, and week-seasonality. Non-patterns are caught in the *remainder*, especially the large dip during the start of COVID-19.

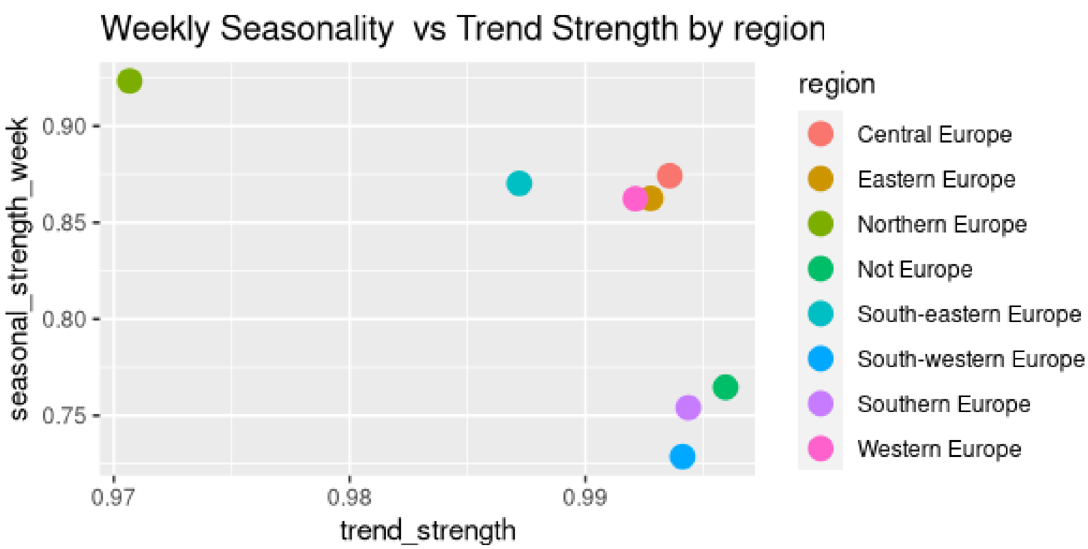
```
TOT_sum %>%
  model(stl = STL(tot)) %>%
  components() %>% autoplot() + xlab("Day") +
  ggtitle("STL Decomposition")
```



Inference

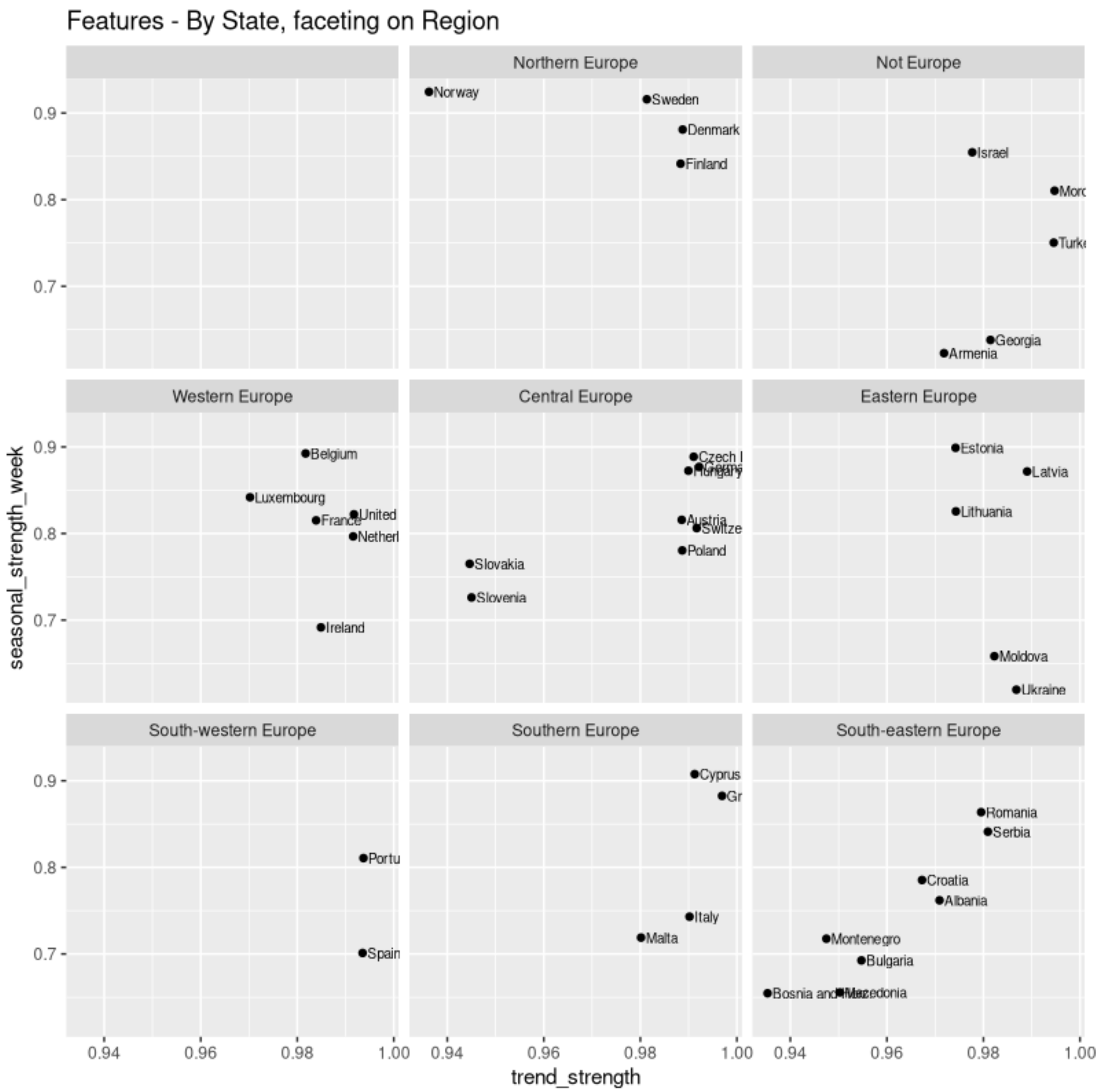
STL decomposition also allows us to look at behavior a time series exhibits, particularly seasonality and how strong it trends.

```
by_region %>%
  features(tot, feat_stl) %>%
  ggplot(aes(x = trend_strength,
             y = seasonal_strength_week,
             col = region)) +
  geom_point(size = 4) +
  ggtitle("Weekly Seasonality  vs Trend Strength by region")
```



This can also be applied to every state:

```
# https://stackoverflow.com/questions/30372368/adding-empty-graphs-to-facet-wrap-in-gg
↪   plot2
feats_state <- flights %>%
  group_by(FLT_DATE, STATE_NAME) %>%
  summarise(tot = sum(FLT_TOT_1), .groups = "keep") %>%
  mutate(FLT_DATE = as.Date(FLT_DATE)) %>%
  as_tsibble(key = STATE_NAME, index = FLT_DATE) %>%
  features(tot, feat_stl) %>%
  mutate(georegion = assignRegion(STATE_NAME)) %>%
  mutate(STATE_NAME = cleanState(STATE_NAME))

feats_state %>%
  ggplot(aes(x = trend_strength,
             y = seasonal_strength_week,
             label = STATE_NAME)) +
  ggtitle("Features - By State, faceting on Region") +
  geom_point() +
  geom_text(size = 2.5, hjust = 0, nudge_x = 0.001) +
  facet_wrap(.~factor(georegion,
                      # order levels to spatially arange facets
                      levels=c('', 'Northern Europe', 'Not Europe',
                               'Western Europe', 'Central Europe', 'Eastern Europe',
                               'South-western Europe', 'Southern Europe',
                               ↪   'South-eastern Europe')),
             drop=FALSE)
```



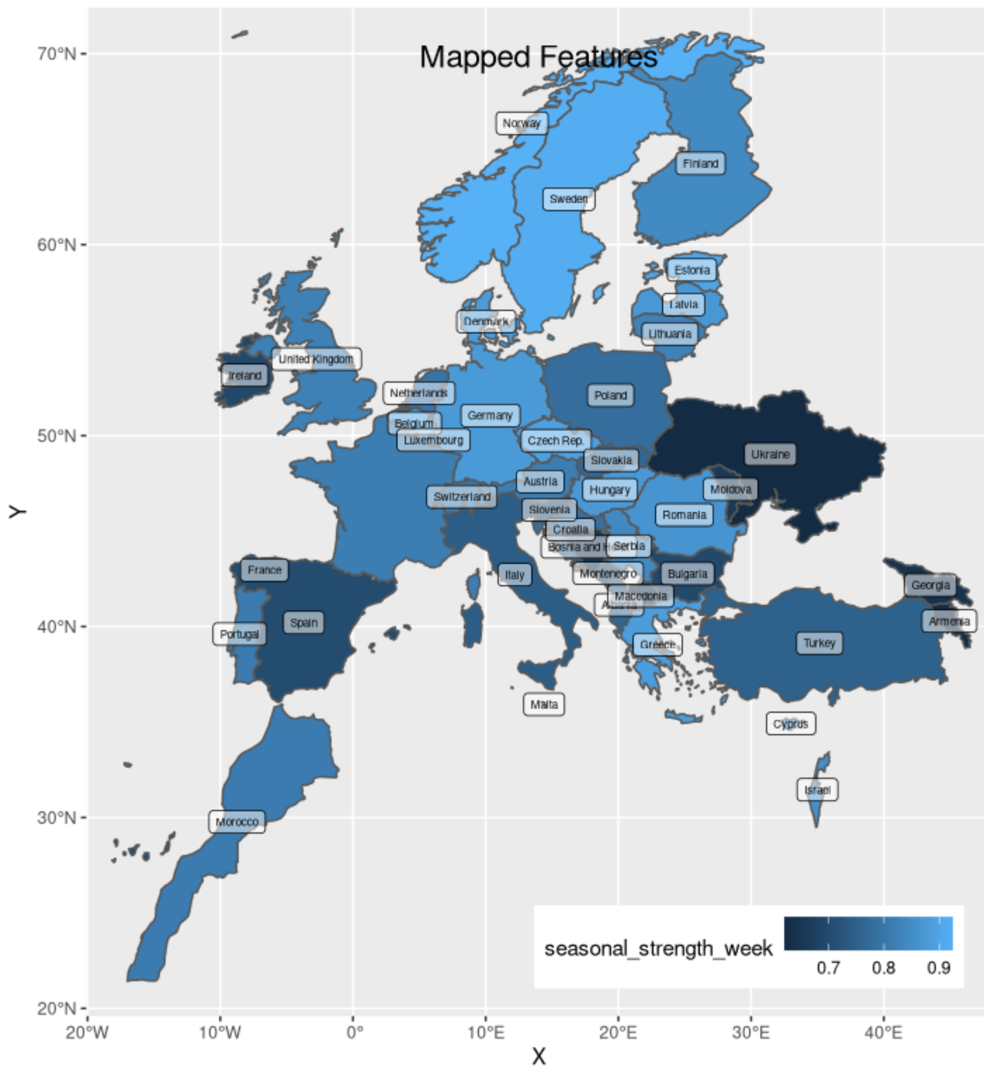
All states have a fairly high trend-strength, never less than `0.94`. On the other hand, Northern, Western, and Central Europe have high weekly seasonality.

GIS

The previous result can be visualized using some GIS libraries.

```
library("sf")
library("rnaturalearth")
library("rnaturalearthdata")
```

```
# "name" is rnaturalearth refers to a country
feats_state_gis <- feats_state %>% rename(name = STATE_NAME)

earth <- ne_countries(scale = "medium", returnclass = "sf")
eu <- right_join(earth, feats_state_gis, by = "name")
eu_coords = data.frame(name = eu$name, st_coordinates(st_centroid(eu)))

ggplot(eu) +
  # relevant: trend_strength, seasonal_strength_week, linearity, curvature
  geom_sf(aes(fill = seasonal_strength_week)) +
  geom_label(data = eu_coords, aes(x=X, y=Y, label = name), size = 2, alpha = 0.5) +
  coord_sf(xlim = c(-17, 45), ylim = c(22, 70)) + # 78 <> 70 to cut off top off Norway
  ↪  (Svalbard)
  ggtitle("Mapped Features") +
  theme(legend.position = c(1,0),
        legend.justification = c(1,0),
        legend.box.margin = margin(5, r = 5, b = 5, unit = "mm"),
        legend.direction = "horizontal",
        plot.title = element_text(vjust = -10, hjust = 0.5, size = 16)
        )
```



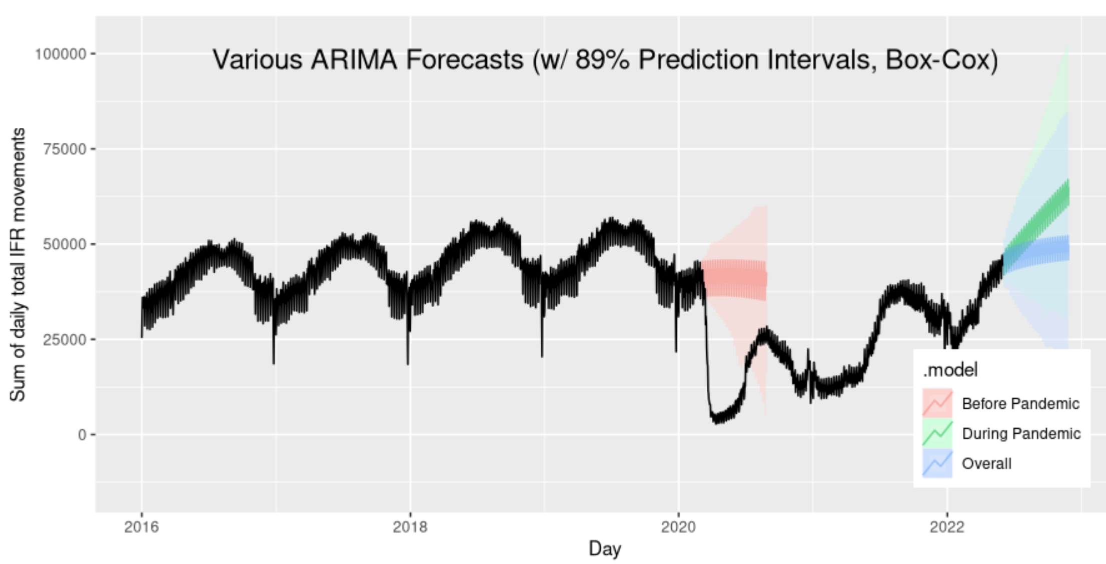
References

- Drawing beautiful maps programmatically with R, sf and ggplot2 — Part 1: Basics

Forecasting

For fun, we can build 3 *ARIMA* models, on data before the lockdowns (< March 2020), after/during (> April 2020), and overall. Box-Cox transformations will also be applied to stabilize results. Forecasts are calculated for half a year.

```
lambda_bef <- TOT_sum %>%
  filter_index(. ~ "2020-02") %>%
  features(tot, features = guerrero) %>% pull(lambda_guerrero)
lambda_dur <- TOT_sum %>%
  filter_index("2020-04" ~ .) %>%
  features(tot, features = guerrero) %>% pull(lambda_guerrero)
lambda_ovr <- TOT_sum %>%
  features(tot, features = guerrero) %>% pull(lambda_guerrero)

H <- 180

before <- TOT_sum %>% filter_index(. ~ "2020-02") %>%
  model("Before Pandemic" = ARIMA(box_cox(tot, lambda_bef))) %>% forecast(h = H)
during <- TOT_sum %>% filter_index("2020-04" ~ .) %>%
  model("During Pandemic" = ARIMA(box_cox(tot, lambda_dur))) %>% forecast(h = H)
overall <- TOT_sum %>%
  model("Overall" = ARIMA(box_cox(tot, lambda_ovr))) %>% forecast(h = H)

bind_rows(during, before, overall) %>%
  autoplot(TOT_sum, level = 89, alpha = 0.5) +
  xlab("Day") + ylab("Sum of daily total IFR movements") +
    ggtitle("Various ARIMA Forecasts (w/ 89% Prediction Intervals, Box-Cox)") +
  theme(legend.position = c(1,0),
        legend.justification = c(1, 0),
        legend.box.margin = margin(5, r = 5, b = 5, unit = "mm"),
        plot.title = element_text(vjust = -10, hjust = 0.5, size = 16)
        ) + guides(level = "none")
```



As expected, the `during` model has a high trend, due to the world bouncing back. Heuristically, `during` would not be a very suitable model as it overshoots the values before lockdowns. However, the future is uncertain, and even the prediction intervals on `overall` capture higher-than-before values.