Lab Session 3: Time series modeling

Don Fejfar

3/16/2021

Set working directory

We will first set a working directory. Your working directory should be where your plan to save your R code and also where the example datasets have been stored. In the "Files" tab, navigate to this folder. Once you are there, select the "More" dropdown and select "Set As Working Directory".

Install and load R packages

You should have already installed the tidyverse package. Now, you will need to load the package into R. This will allow you to use the functionality of the package.

```
library(tidyverse)
library(lubridate)
library(car)
```

STEP 1. Choose relevant indicator and visualize data

Load the monthly number of acute respiratory infections ".rds" file and save it as a data frame called "facility".

```
facility <- readRDS("session3_data/example_facility_ari.rds")
```

View the first six observations in the facility dataset.

```
head(facility)
```

What is the date range in the dataset?

```
facility %>%
  summarize(min(date),
            max(date))
```

Create a scatterplot of the date and actue respiratory infection count.

```
ggplot(facility,aes(x=date,y=ari_count)) +
  geom_point() +
  geom_line() +
  theme_bw()
```

STEP 2. Choose the baseline and evaluation period

The evaluation period will start on January 1, 2020. Include a vertical line on the scatter plot.

```
ggplot(facility,aes(x=date,y=ari_count)) +
  geom_point() +
  geom_line() +
  theme_bw() +
  geom_vline(xintercept=as.Date("2020-01-01"))
```

STEP 3. Fit time series model to baseline period

Modify the facility dataset to include: time (month number), year, one cosine, and one sine term.

```
facility %>%
  arrange(date) %>%
  mutate(time=1:n(),
         year=year(date)) %>%
  mutate(cos1=cos(2*pi*time/12),
         sin1=sin(2*pi*time/12)) -> facility_new
```

Fit a model with only a linear time term on the baseline period data. Call this *fit1*.

```
facility_new_baseline <- facility_new %>% filter(date < as.Date("2020-01-01"))

fit1 <- lm(ari_count ~ time,data=facility_new_baseline)

summary(fit1)
```

STEP 4. Using the model from Step 3, calculate deviations from expected in the evaluation period.

Calculate the difference between observed and expected in the evaluation period. Do you notice a pattern?

```
facility_new %>%
  mutate(deviation=ari_count-fit1$fitted.values) %>%
  filter(date >= as.Date("2020-01-01")) %>%
  dplyr::select(date,deviation)
```

In order to compare the deviations across each month, standardize by the predicted values.

```
facility_new %>%
  mutate(deviation=(ari_count-fit1$fitted.values)/fit1$fitted.values) %>%
  filter(date >= as.Date("2020-01-01")) %>%
  dplyr::select(date,deviation)
```

Step 5. Produce interpretable visualizations.

Calculate the 95% prediction intervals.

```
predicted_values <- predict(fit1, facility_new, interval="predict")
```

Plot the observed values (points), predicted line, and 95% prediction intervals. **Hint:** You will need to pull the date and observed counts into the prediction interval data frame.

```
predicted_values %>%
  data.frame() %>%
  mutate(date=facility_new$date,
         observed=facility_new$ari_count) -> predicted_values_new

ggplot(predicted_values_new,aes(x=date,y=observed)) +
  geom_point() +
  geom_ribbon(aes(ymin=lwr,ymax=upr),fill = "grey70",alpha=.4) +
  geom_line(aes(x=date,y=fit)) +
  geom_vline(xintercept=as.Date("2020-01-01")) +
  theme_bw() +
  theme(legend.position="none")
```

ACTIVITY

1. Repeat Steps 3-5 above with a model including the following terms: year, cosine, and sine term (created above). Call this *fit2*.

```
# Fit the model
facility_new_baseline <- facility_new %>% filter(date < as.Date("2020-01-01"))

fit2 <- lm(ari_count ~ year + cos1 + sin1,data=facility_new_baseline)

summary(fit2)

# Deviations
facility_new %>%
  mutate(deviation=ari_count-fit2$fitted.values) %>%
  filter(date >= as.Date("2020-01-01")) %>%
  dplyr::select(date,deviation)

# 95% Predicted intervals
predicted_values2 <- predict(fit2, facility_new, interval="predict")

predicted_values2 %>%
  data.frame() %>%
  mutate(date=facility_new$date,
         observed=facility_new$ari_count) -> predicted_values_new2

ggplot(predicted_values_new2,aes(x=date,y=observed)) +
  geom_point() +
  geom_ribbon(aes(ymin=lwr,ymax=upr),fill = "grey70",alpha=.4) +
  geom_line(aes(x=date,y=fit)) +
  geom_vline(xintercept=as.Date("2020-01-01")) +
  theme_bw() +
  theme(legend.position="none")
```

2. Compare the ACF plots between fit1 and fit2. Which looks better?

```
acf(fit1$residuals,lag.max=48)

acf(fit2$residuals,lag.max=48)
```