SEJ Analytix: Smoothed Averages for the Daily State Traffic Volume in 2020

Create time series for plotting and analyzing

Emanuela Ene

#Clear the space

```
rm(list = ls())
```

Introduction and Data

We will be converting the Daily Traffic Volume at the FTS stations in the US cumulative per state in 2020 to smoothed averages.

The state cumulative (imputed) daily traffic was computed by SEJ Analytix from the original data at https://www.fhwa.dot.gov/policyinformation/tables/tmasdata/ (https://www.fhwa.dot.gov/policyinformation/tables/tmasdata/).
The COVID19 state policies were extracted from https://en.wikipedia.org/wiki/U.S._state_and_local_government_responses_to_the_COVID19_pandemic (https://en.wikipedia.org/wiki/U.S._state_and_local_government_responses_to_the_COVID19_pandemic)
and https://www.nytimes.com/interactive/2020/us/states-reopen-map-coronavirus.html (https://www.nytimes.com/interactive/2020/us/states-reopen-map-coronavirus.html)   The COVID19 state policies were coded numerically.

The structure of the imputed state cumulative daily traffic data is

X - control index
date - full calendar date
FIPS.State.Code - numeric code for state, as in https://www.bls.gov/respondents/mwr/electronic-data-interchange/appendix-d-usps-state-abbreviations-and-fips-codes.htm (https://www.bls.gov/respondents/mwr/electronic-data-interchange/appendix-d-usps-state-abbreviations-and-fips-codes.htm) Year.of.Data - two digit code for year
Month.of.Data - numeric month of the year
Day.of.Data - numeric day of the month
stateTraffic - state cumulative daily traffic, million of vehicles   Day.of.Week - numeric day of the week, Sunday=1
Week.of.Year - numeric week of year

```
setwd("C:/Users/Mama/Desktop/Customer_cases/CovidOnTransportation")
```

Load packages

```
require(tidyverse)
```

```
## Loading required package: tidyverse
```

```
## Warning: package 'tidyverse' was built under R version 4.3.3
```

```
## Warning: package 'ggplot2' was built under R version 4.3.3
```

```
## Warning: package 'tibble' was built under R version 4.3.3
```

```
## Warning: package 'tidyr' was built under R version 4.3.3
```

```
## Warning: package 'readr' was built under R version 4.3.3
```

```
## Warning: package 'purrr' was built under R version 4.3.3
```

```
## Warning: package 'dplyr' was built under R version 4.3.3
```

```
## Warning: package 'stringr' was built under R version 4.3.3
```

```
## Warning: package 'forcats' was built under R version 4.3.3
```

```
## Warning: package 'lubridate' was built under R version 4.3.3
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✔ dplyr     1.1.4     ✔ readr     2.1.5
## ✔ forcats   1.0.0     ✔ stringr   1.5.1
## ✔ ggplot2   3.5.1     ✔ tibble    3.2.1
## ✔ lubridate 1.9.3     ✔ tidyr     1.3.1
## ✔ purrr     1.0.2
```

```
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becom
e errors
```

```
require(astsa)
```

```
## Loading required package: astsa
```

Read the *.csv file for traffic volume.

```
timelineCovid<-read.csv("C:/Users/Mama/Desktop/Customer_cases/CovidOnTransportation/categorical&
numerical_Covid & imputed_TrafficDaily.csv", header=T, all=T)
stateList<-read.csv("C:/Users/Mama/Desktop/Customer_cases/CovidOnTransportation/USPS_State_Abbre
viations.csv", header=T, all=T)
```

Extract columns of interest for the Shiny app.

```
numericalCovid<-timelineCovid%>%select(-emer, -lock,-trv,-cls)
numericalCovid <- numericalCovid%>%mutate(date = ymd(date))

#create a list of all states in the data base
allStateCodes<-numericalCovid$FIPS.State.Code
save(allStateCodes, file="allStateCodes.R")
```

Store state info

```
stateInfo=list()

for ( n in allStateCodes) {
  codeFIPS<-n
  name <- (stateList %>% dplyr::filter(FIPS.State.Code == codeFIPS))$stateName
  abbr<- (stateList %>% dplyr::filter(FIPS.State.Code == codeFIPS))$stateCode
  stateInfo[[codeFIPS]]<-c(codeFIPS=codeFIPS, name=name, abbr=abbr)
}
save(stateInfo, file="stateInfo.R")
```

All traffic 2020 time series

```
library(lubridate)
library(zoo)
```

```
## Warning: package 'zoo' was built under R version 4.3.3
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
traffic_ts=list()

for ( n in allStateCodes) {

traffic20<-numericalCovid%>% filter(FIPS.State.Code == n)%>%select(date, stateTraffic)%>% mutate(traffic20 = rollmean(stateTraffic, k = 7, fill = NA, align = "center"))%>%select(traffic20)
traffic_ts[[n]]<-traffic20
}
save(traffic_ts, file="traffic_ts.R")
```

All prior traffic time series

```
library(lubridate)
library(zoo)

prior_ts=list()

for ( n in allStateCodes) {

prior<-numericalCovid%>% filter(FIPS.State.Code == n)%>%select(date, priorTraffic)%>% mutate(prior = rollmean(priorTraffic, k = 7, fill = NA, align = "center"))%>%select(prior)
prior_ts[[n]]<-prior
}

save(prior_ts, file="prior_ts.R")
```

All emergency time series

```
library(lubridate)
library(zoo)

emer_ts=list()

for ( n in allStateCodes) {

emer<-numericalCovid%>% filter(FIPS.State.Code == n)%>%select(date, emergency)%>% mutate(emer = rollmean(emergency, k = 7, fill = NA, align = "center"))%>%select(emer)
emer_ts[[n]]<-emer
}

save(emer_ts, file="emer_ts.R")
```

All lockDown time series

```
library(lubridate)
library(zoo)

lock_ts=list()

for ( n in allStateCodes) {

lock<-numericalCovid%>% filter(FIPS.State.Code == n)%>%select(date, lockDown)%>% mutate(lock = rollmean(lockDown, k = 7, fill = NA, align = "center"))%>%select(lock)
lock_ts[[n]]<-lock
}

save(lock_ts, file="lock_ts.R")
```