In this blog post you will learn how to:

* get free city district polygons data from OpenStreetMaps API,
* get free street lines (polygons) coordinates from OpenStreetMaps API,
* visualize polygons and street lines with ggplot2 and ggmap,
* merge spatial data with violation data represented as time series,
* animate spatial data combined with violation time series with gganimate.

**Bratislava Open-Data**

The ultimate goal is to show where and when are the most dangerous places in the capital of Slovakia – Bratislava.  
For this task, I will use open-data that cover violations gathered from the city police with locations (city district and street name) and time-stamp.

Firstly, load all the needed packages.

library(data.table) # handling data

library(lubridate) # handling timestamps

library(stringi) # text processing

library(osmdata) # OSM data API

# library(nominatim) # alternative OSM data API

library(sf) # handling spatial data

library(ggplot2) # visualisations

library(ggsci) # color pallets

library(ggmap) # map vis

library(gganimate) # animations

library(magick) # image handling

Then, let’s download the violations data from [opendata.bratislava.sk](https://opendata.bratislava.sk/) webpage and translate Slovak column names to English.

data\_violation\_17 <- fread("<https://opendata.bratislava.sk/dataset/download/852>",

encoding = "Latin-1")

data\_violation\_18 <- fread("<https://opendata.bratislava.sk/dataset/download/407>",

encoding = "Latin-1")

data\_violation\_19 <- fread("<https://opendata.bratislava.sk/dataset/download/911>",

encoding = "Latin-1")

# colnames to English

setnames(data\_violation\_17, colnames(data\_violation\_17), c("Date\_Time", "Group\_vio",

"Type\_vio", "Street", "Place"))

setnames(data\_violation\_18, colnames(data\_violation\_18), c("Date\_Time", "Group\_vio",

"Type\_vio", "Street", "Place"))

setnames(data\_violation\_19, colnames(data\_violation\_19), c("Date\_Time", "Group\_vio",

"Type\_vio", "Street", "Place"))

Let’s bind all the data together.

data\_violation <- rbindlist(list(data\_violation\_17, data\_violation\_18, data\_violation\_19),

use.names = T)

Next, I will transform Date\_Time to POCIXct format and generate time aggregation features – Year and Month – *Year\_M*.

data\_violation[, Date\_Time := dmy\_hm(Date\_Time)]

data\_violation[, Date := date(Date\_Time)]

# Aggregation features

data\_violation[, Month := month(Date\_Time)]

data\_violation[, Year := year(Date\_Time)]

data\_violation[, Year\_M := Year + Month/100]

Let’s see how many violations have each *Place* (city district) for whole available period (2017-2019.09):

data\_violation[, .N, by = .(Place)]

## Place N

## 1: Stare Mesto 68714

## 2: Karlova Ves 11462

## 3: Petrzalka 17062

## 4: Dubravka 6307

## 5: Ruzinov 21352

## 6: Vajnory 1194

## 7: Pod. Biskupice 1590

## 8: Vrakuna 1262

## 9: Nove Mesto 17964

## 10: Devin 269

## 11: Raca 2390

## 12: Devinska Nova Ves 1406

## 13: Lamac 1455

## 14: Jarovce 66

## 15: Zah. Bystrica 928

## 16: Rusovce 753

## 17: Cunovo 569

We can see that Old-town (Stare Mesto) rocks in this statistic..obviously – e.g. lot of tourists. There are also some misunderstand Slavic letters. We should get rid of them – in the blog post there we be lot of handling of these special Slavic (Slovak) symbols.

data\_violation[, Place := stri\_trans\_general(Place, "Latin-ASCII")]

data\_violation[.("Eunovo"), on = .(Place), Place := "Cunovo"]

data\_violation[.("Raea"), on = .(Place), Place := "Raca"]

data\_violation[.("Vrakuoa"), on = .(Place), Place := "Vrakuna"]

data\_violation[.("Lamae"), on = .(Place), Place := "Lamac"]

I will extract only types of violations that relate to the bad behavior of people, like harassment, using alcoholic beverages on public space and related things. So, I will not extract traffic violations like bad car parking, etc. For this task, I need to extract codes of violations:

data\_violation[, Group\_vio\_ID := as.numeric(substr(Group\_vio, 1, 4))]

# our codes are: 4836, 5000, 4834, 4700, 4838, 4828, 4839,

# 4813, 9803, 4840, 4841, 9809, 3000, 9806, 4900

**Polygons of city districts**

I want to show a number of crimes per city district and street (so both place information) on a map, so I need to get coordinates of city districts and streets.  
Let’s get polygons of city districts first using OpenStreetMap API.

poly\_sm <- getbb(place\_name = "Stare Mesto Bratislava", format\_out = "polygon")

poly\_nm <- getbb(place\_name = "Nove Mesto Bratislava", format\_out = "polygon")

poly\_ruz <- getbb(place\_name = "Ruzinov Bratislava", format\_out = "polygon")

poly\_raca <- getbb(place\_name = "Raca Bratislava", format\_out = "polygon")

poly\_petr <- getbb(place\_name = "Petrzalka Bratislava", format\_out = "polygon")

poly\_kv <- getbb(place\_name = "Karlova Ves Bratislava", format\_out = "polygon")

poly\_dubr <- getbb(place\_name = "Dubravka Bratislava", format\_out = "polygon")

poly\_vajn <- getbb(place\_name = "Vajnory Bratislava", format\_out = "polygon")

poly\_lamac <- getbb(place\_name = "Lamac Bratislava", format\_out = "polygon")

poly\_vrak <- getbb(place\_name = "Vrakuna Bratislava", format\_out = "polygon")

poly\_pd <- getbb(place\_name = "Podunajske Biskupice Bratislava", format\_out = "polygon")

poly\_jar <- getbb(place\_name = "Jarovce Bratislava", format\_out = "polygon")

poly\_dnv <- getbb(place\_name = "Devinska Nova Ves Bratislava", format\_out = "polygon")

poly\_rus <- getbb(place\_name = "Rusovce Bratislava", format\_out = "polygon")

poly\_zb <- getbb(place\_name = "Zahorska Bystrica Bratislava", format\_out = "polygon")

poly\_devin <- getbb(place\_name = "Devin Bratislava", format\_out = "polygon")

poly\_cun <- getbb(place\_name = "Cunovo Bratislava", format\_out = "polygon")

Let’s bind all the districts data of Bratislava.

data\_poly\_ba <- rbindlist(list(

as.data.table(poly\_sm)[, Place := "Stare Mesto"],

as.data.table(poly\_nm[[1]])[, Place := "Nove Mesto"],

as.data.table(poly\_ruz[[1]])[, Place := "Ruzinov"],

as.data.table(poly\_raca)[, Place := "Raca"],

as.data.table(poly\_petr)[, Place := "Petrzalka"],

as.data.table(poly\_kv[[1]])[, Place := "Karlova Ves"],

as.data.table(poly\_dubr[[1]])[, Place := "Dubravka"],

as.data.table(poly\_vajn[[1]])[, Place := "Vajnory"],

as.data.table(poly\_lamac[[1]])[, Place := "Lamac"],

as.data.table(poly\_vrak[[1]])[, Place := "Vrakuna"],

as.data.table(poly\_pd[[1]])[, Place := "Pod. Biskupice"],

as.data.table(poly\_jar[[1]])[, Place := "Jarovce"],

as.data.table(poly\_dnv[[1]])[, Place := "Devinska Nova Ves"],

as.data.table(poly\_rus[[1]])[, Place := "Rusovce"],

as.data.table(poly\_zb[[1]])[, Place := "Zah. Bystrica"],

as.data.table(poly\_devin[[1]])[, Place := "Devin"],

as.data.table(poly\_cun[[1]])[, Place := "Cunovo"]

))

setnames(data\_poly\_ba, c("V1", "V2"), c("lon", "lat"))

data\_poly\_ba[, Place := factor(Place)]

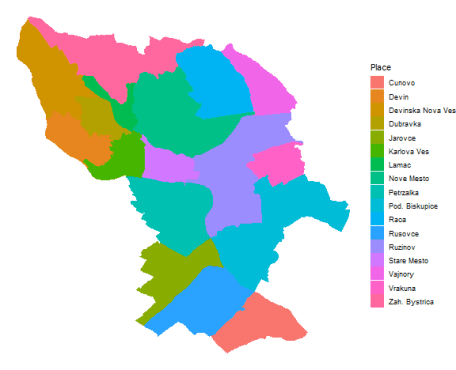
Let’s visualize simply the districts.

ggplot(data\_poly\_ba) +

geom\_polygon(aes(x = lon, y = lat,

fill = Place, color = Place)) +

theme\_void()



For animation and visualization purposes, I need to aggregate violations data by districts (*Place*) and Year+Month columns (*Year\_M*).

data\_violation\_agg\_place <- copy(data\_violation[.(c(4836, 5000, 4834, 4700, 4838,

4828, 4839, 4813, 9803, 4840,

4841, 9809, 3000, 9806, 4900)),

on = .(Group\_vio\_ID),

.(N = .N),

by = .(Place, Year\_M)])

The next step is to merge polygon data with aggregated violation data:

data\_places <- merge(data\_poly\_ba[, .(Place, lon, lat)],

data\_violation\_agg\_place[, .(Place, N\_violation = N, Year\_M)],

by = "Place",

all.y = T,

allow.cartesian = T)

Let’s also compute mean coordinates for every district for showing theirs names on a graph.

data\_poly\_ba\_mean\_place <- copy(data\_poly\_ba[, .(lon = mean(lon),

lat = mean(lat)),

by = .(Place)])

Let’s test visualization of violations on June 2019:

ggplot() +

geom\_polygon(data = data\_places[.(2019.06), on = .(Year\_M)],

aes(x = lon, y = lat,

fill = N\_violation,

group = Place),

color = "grey40") +

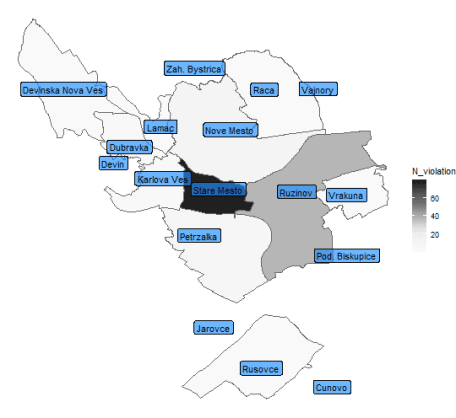
scale\_fill\_material("grey") +

geom\_label(data = data\_poly\_ba\_mean\_place,

aes(x = lon, y = lat, label = Place),

color = "black", fill = "dodgerblue1", alpha = 0.65) +

theme\_void()



**Street lines coordinates**

The next step is to download street coordinates from OpenStreetMaps.

Firstly, we have to extract unique street names from violation data, and handle Slovak letters and other punctuation for easier matching by street name.

dt\_uni\_streets <- copy(data\_violation[, .(Place = data.table::first(Place)),

by = .(Street)])

dt\_uni\_streets[, Street\_edit := gsub(pattern = "Ãˆ", replacement = "C", x = Street)]

dt\_uni\_streets[, Street\_edit := gsub(pattern = "Ã²", replacement = "n", x = Street\_edit)]

dt\_uni\_streets[, Street\_edit := gsub(pattern = "Ã¨", replacement = "c", x = Street\_edit)]

dt\_uni\_streets[, Street\_edit := gsub(pattern = "\u009d", replacement = "t", x = Street\_edit)]

dt\_uni\_streets[, Street\_edit := gsub(pattern = "Â¾", replacement = "l", x = Street\_edit)]

dt\_uni\_streets[, Street\_edit := gsub(pattern = "Â¼", replacement = "L", x = Street\_edit)]

dt\_uni\_streets[, Street\_edit := gsub(pattern = "Ã²", replacement = "n", x = Street\_edit)]

dt\_uni\_streets[, Street\_edit := gsub(pattern = "\_", replacement = "", x = Street\_edit)]

dt\_uni\_streets[, Street\_edit := gsub(pattern = "Ã¯", replacement = "d", x = Street\_edit)]

dt\_uni\_streets[, Street\_edit := gsub(pattern = "Ã ", replacement = "r", x = Street\_edit)]

dt\_uni\_streets[, Street\_edit := stri\_trans\_general(Street\_edit, "Latin-ASCII")]

dt\_uni\_streets[, Street\_edit := sub(" - MC.\*", "", Street\_edit)]

dt\_uni\_streets[, Street\_edit := gsub(pattern = "Nam.", replacement = "Namestie ",

x = Street\_edit)]

dt\_uni\_streets[, Street\_query := gsub(pattern = " ", replacement = "+", x = Street\_edit)]

dt\_uni\_streets[, Street\_query := paste0(Street\_query, "+Bratislava")]

Let’s get all streets coordinates of Bratislava just by one (powerful) command, again using OSM API:

ba\_bb <- getbb(place\_name = "Bratislava")

bratislava <- opq(bbox = ba\_bb) %>%

add\_osm\_feature(key = 'highway') %>%

osmdata\_sf() %>%

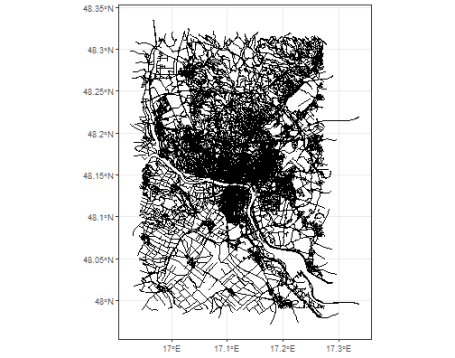
osm\_poly2line()

Let’s plot it:

ggplot(data = bratislava$osm\_lines) +

geom\_sf() +

theme\_bw()



Pretty nice web.

Now, we need to handle again street names downloaded from OSM. I will extract only available streets from violation data:

street\_names <- data.table(Street = unique(bratislava$osm\_lines$name))

street\_names[, Street\_edit := gsub(pattern = "Ãƒ", replacement = "i", x = Street)]

street\_names[, Street\_edit := gsub(pattern = "iÂ©", replacement = "e", x = Street\_edit)]

street\_names[, Street\_edit := gsub(pattern = "iÂ¡", replacement = "a", x = Street\_edit)]

street\_names[, Street\_edit := gsub(pattern = "iÂº", replacement = "u", x = Street\_edit)]

street\_names[, Street\_edit := gsub(pattern = "Ã…Â¾", replacement = "z", x = Street\_edit)]

street\_names[, Street\_edit := gsub(pattern = "Ã„Â¾", replacement = "l", x = Street\_edit)]

street\_names[, Street\_edit := gsub(pattern = "Ã„\u008d", replacement = "c", x = Street\_edit)]

street\_names[, Street\_edit := gsub(pattern = "Ã…Â¡", replacement = "s", x = Street\_edit)]

street\_names[, Street\_edit := gsub(pattern = "Ã„Â½", replacement = "L", x = Street\_edit)]

street\_names[, Street\_edit := gsub(pattern = "Ã…Ë†", replacement = "n", x = Street\_edit)]

street\_names[, Street\_edit := gsub(pattern = "iÂ½", replacement = "y", x = Street\_edit)]

street\_names[, Street\_edit := gsub(pattern = "Ã…â€¢", replacement = "r", x = Street\_edit)]

street\_names[, Street\_edit := gsub(pattern = "iÂ³", replacement = "o", x = Street\_edit)]

street\_names[, Street\_edit := gsub(pattern = "iÂ¤", replacement = "a", x = Street\_edit)]

street\_names[, Street\_edit := gsub(pattern = "Ã…Â½", replacement = "Z", x = Street\_edit)]

street\_names[, Street\_edit := gsub(pattern = "Ã… ", replacement = "S", x = Street\_edit)]

street\_names[, Street\_edit := gsub(pattern = "Ã…Â¥", replacement = "t", x = Street\_edit)]

street\_names[, Street\_edit := gsub(pattern = "Ã„Å’", replacement = "C", x = Street\_edit)]

street\_names[, Street\_edit := gsub(pattern = "Ã„\u008f", replacement = "d", x = Street\_edit)]

street\_names[, Street\_edit := gsub(pattern = "iÂ¶", replacement = "o", x = Street\_edit)]

street\_names[, Street\_edit := gsub(pattern = "Ã…â€˜", replacement = "o", x = Street\_edit)]

street\_names[, Street\_edit := gsub(pattern = "Ã„â€º", replacement = "e", x = Street\_edit)]

street\_names[, Street\_edit := gsub(pattern = "Ã„â€º", replacement = "e", x = Street\_edit)]

street\_names[, Street\_edit := stri\_trans\_general(Street\_edit, "Latin-ASCII")]

street\_names[, Street\_edit := gsub(pattern = "i-", replacement = "i", x = Street\_edit)]

street\_names[, Street\_edit := gsub(pattern = "A ", replacement = "S", x = Street\_edit)]

street\_names[dt\_uni\_streets, on = .(Street\_edit)]

street\_names\_vio <- copy(street\_names[dt\_uni\_streets, on = .(Street\_edit),

Street\_orig := i.Street])[![is.na](http://is.na)(Street\_orig)]

We lost some street data from the dataset by not exact matched street names.

Let’s subset the street data.

ba\_streets\_vio <- bratislava$osm\_lines

ba\_streets\_vio <- ba\_streets\_vio[ba\_streets\_vio$name %in% street\_names\_vio$Street,]

ba\_streets\_vio <- ba\_streets\_vio[![is.na](http://is.na)(ba\_streets\_vio$name),]

Now, I will transform data to standard lon/lat matrix (data.table class) format instead of sf object (I highly recommend this for next ggplot usage).

data\_streets\_st <- data.table::data.table(sf::st\_coordinates(ba\_streets\_vio$geometry))

Next, I will bound streets by existing polygons of Bratislava and add merging column – Street\_edit:

ba\_streets\_vio$L1 <- 1:nrow(ba\_streets\_vio)

data\_streets\_st <- merge(data\_streets\_st, as.data.table(ba\_streets\_vio)[, .(L1, name)],

by = "L1")

data\_streets\_st[street\_names\_vio[, .(name, Street\_edit)], on = .(name),

Street\_edit := i.Street\_edit]

setnames(data\_streets\_st, c("X", "Y"), c("lon", "lat"))

data\_streets\_st <- data\_streets\_st[!lon > data\_poly\_ba[, max(lon)]]

data\_streets\_st <- data\_streets\_st[!lon < data\_poly\_ba[, min(lon)]]

data\_streets\_st <- data\_streets\_st[!lat > data\_poly\_ba[, max(lat)]]

Let’s also aggregate violation data by street names and Year + Month (*Year\_M*):

data\_violation\_agg\_street <- copy(data\_violation[.(c(4836, 5000, 4834, 4700, 4838,

4828, 4839, 4813, 9803, 4840,

4841, 9809, 3000, 9806, 4900)),

on = .(Group\_vio\_ID),

.(N = .N), by = .(Street, Year\_M)])

setorder(data\_violation\_agg\_street, Year\_M, -N)

Let’s add look-up column *Street\_edit* to aggregated data and merge spatial street data with violation data.

data\_violation\_agg\_street[dt\_uni\_streets, on = .(Street), Street\_edit := i.Street\_edit]

data\_streets\_st <- merge(data\_streets\_st,

data\_violation\_agg\_street[, .(Street\_edit, N\_violations = N, Year\_M)],

by = "Street\_edit",

all.y = T,

allow.cartesian = TRUE)

Now, I will transform integers of the number of violations to reasonable factors segments:

data\_streets\_st[N\_violations >= 11, N\_violation\_type := "> 11"]

data\_streets\_st[N\_violations >= 6 & N\_violations < 11, N\_violation\_type := "(6, 10)"]

data\_streets\_st[N\_violations >= 3 & N\_violations < 6, N\_violation\_type := "(3, 5)"]

data\_streets\_st[N\_violations <= 2, N\_violation\_type := "< 2"]

data\_streets\_st[, N\_violations\_per\_street := factor(N\_violation\_type,

levels = c("< 2", "(3, 5)",

"(6, 10)", "> 11"))]

levels(data\_streets\_st$N\_violations\_per\_street)

## [1] "< 2" "(3, 5)" "(6, 10)" "> 11"

Let’s see what we extracted so far for streets in one example year-month…

ggplot(data\_streets\_st[![is.na](http://is.na)(L1)][.(2019.06),

on = .(Year\_M)]) +

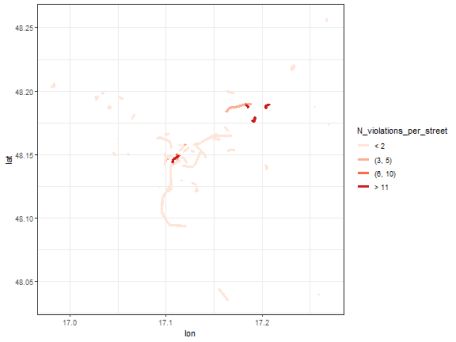
geom\_line(aes(lon, lat, group = L1,

color = N\_violations\_per\_street),

size = 1.4) +

scale\_color\_brewer(palette = "Reds") +

theme\_bw()



**ggmap**

Since, we will use ggmap for visualizations, we need to extract map image of Bratislava:

bbox <- make\_bbox(lon, lat, data = data\_poly\_ba, f = .01) # used our polygons

map\_ba <- get\_map(location = bbox, source = 'stamen', maptype = "watercolor")

Now, let’s make it altogether to one visualization – so violations by city districts and also streets for one time-stamp (*Year\_M*).

ggmap(map\_ba, extent = "device") +

geom\_polygon(data = data\_places[.(2019.06),

on = .(Year\_M)],

aes(x = lon, y = lat,

fill = N\_violation,

group = Place),

color = "grey40") +

scale\_fill\_material("grey", alpha = 0.75) +

geom\_line(data = data\_streets\_st[![is.na](http://is.na)(L1)][.(2019.06),

on = .(Year\_M)],

aes(lon, lat, group = L1, color = N\_violations\_per\_street),

size = 1.4) +

scale\_color\_brewer(palette = "Reds") +

geom\_label(data = data\_poly\_ba\_mean\_place,

aes(x = lon, y = lat, label = Place,

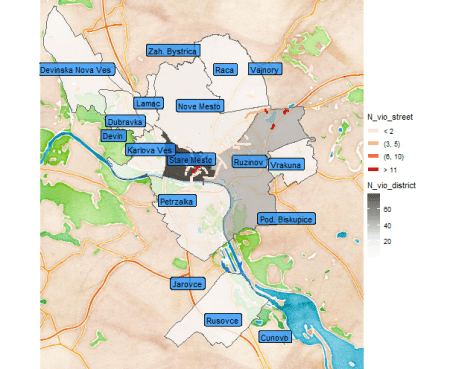
fill = NA\_real\_),

color = "black", fill = "dodgerblue1", alpha = 0.75) +

labs(fill = "N\_vio\_district", color = "N\_vio\_street") +

guides(color = guide\_legend(override.aes = list(size = 2))) +

theme\_void()



I will also extract the most violated streets by Year+Month for labeling purposes.

data\_streets\_st\_max <- copy(data\_streets\_st[![is.na](http://is.na)(L1),

.SD[N\_violations == max(N\_violations, na.rm = T)],

by = .(Year\_M)])

data\_streets\_st\_max[, unique(Street\_edit)] # the unique month "winners"

data\_streets\_st\_max <- copy(data\_streets\_st\_max[, .(lon = mean(lon),

lat = mean(lat),

Street\_edit = first(Street\_edit),

N\_violations = first(N\_violations),

N\_violations\_per\_street =

first(N\_violations\_per\_street)

),

by = .(Year\_M)])

As we noticed so far, the most violated city district is Old-town, so I will create also zoomed visualization for this city district like this (using coord\_map function):

ggmap(map\_ba) +

geom\_polygon(data = data\_places[.(2019.06), on = .(Year\_M)],

aes(x = lon, y = lat,

fill = N\_violation,

group = Place

),

color = "grey40") +

scale\_fill\_material("grey", alpha = 0.75) +

geom\_line(data = data\_streets\_st[![is.na](http://is.na)(L1)][.(2019.06), on = .(Year\_M)],

aes(lon, lat, group = L1, color = N\_violations\_per\_street),

size = 1.4) +

geom\_label(data = data\_streets\_st\_max[.(2019.06), on = .(Year\_M)],

aes(x = lon, y = lat,

label = paste0(Street\_edit, " - ", N\_violations)

),

color = "black",

fill = "firebrick2",

size = 4.5,

alpha = 0.75

) +

scale\_color\_brewer(palette = "Reds") +

geom\_label(data = data\_poly\_ba\_mean\_place,

aes(x = lon, y = lat, label = Place,

fill = NA\_real\_),

color = "black", fill = "dodgerblue1",

alpha = 0.75, size = 4.5) +

coord\_map(

xlim=c(data\_poly\_ba[.("Stare Mesto"), on = .(Place), min(lon)],

data\_poly\_ba[.("Stare Mesto"), on = .(Place), max(lon)]),

ylim= c(data\_poly\_ba[.("Stare Mesto"), on = .(Place), min(lat)],

data\_poly\_ba[.("Stare Mesto"), on = .(Place), max(lat)])

) +

labs(fill = "N\_vio\_district", color = "N\_vio\_street") +

guides(color = guide\_legend(ncol = 1, override.aes = list(size = 2),

title.position = "top"),

fill = guide\_legend(nrow = 2, title.position = "top")) +

theme\_bw() +

theme(axis.text = element\_blank(),

axis.line = element\_blank(),

axis.title = element\_blank(),

axis.ticks = element\_blank(),

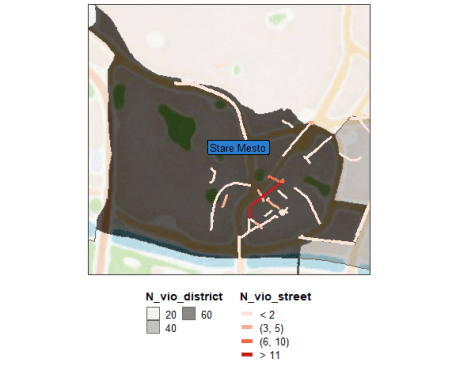
legend.text = element\_text(size = 13),

legend.title = element\_text(size = 14, face = "bold"),

legend.background = element\_rect(fill = "white"),

legend.key = element\_rect(fill = "white"),

legend.position = "bottom")



**Using ggnanimate**

We have all prepared to create some interesting animated map.  
For this purpose, I will use the gganimate package that simply uses the transition\_states function that combines our previously prepared ggplot2 and ggmap plot and time feature *Year\_M*.  
Let’s firstly animate the whole city picture.

gg\_ba\_anim\_lines <- ggmap(map\_ba, extent = "device") +

geom\_polygon(data = data\_places,

aes(x = lon, y = lat,

fill = N\_violation,

group = Place),

color = "grey40") +

scale\_fill\_material("grey", alpha = 0.7) +

geom\_line(data = data\_streets\_st[![is.na](http://is.na)(L1)],

aes(lon, lat,

group = L1,

color = N\_violations\_per\_street

),

size = 1.4

) +

scale\_color\_brewer(palette = "Reds") +

labs(fill = "N\_vio\_district", color = "N\_vio\_street") +

geom\_label(data = data\_poly\_ba\_mean\_place,

aes(x = lon, y = lat, label = Place,

fill = NA\_real\_),

color = "black", fill = "dodgerblue1",

alpha = 0.75, size = 4.5) +

transition\_states(reorder(Year\_M, Year\_M), transition\_length = 2,

state\_length = 1) +

labs(title = "Year.Month - {closest\_state}") +

guides(color = FALSE, fill = FALSE) +

theme\_void() +

theme(plot.title = element\_text(size = 16, face = "bold")

)

# save animation to object

anim\_whole <- animate(gg\_ba\_anim\_lines, duration = 46, width = 700, height = 750)

Then, let’s animate zoomed map of Old-town district.

gg\_ba\_anim\_lines\_zoom <- ggmap(map\_ba, extent = "device") +

geom\_polygon(data = data\_places,

aes(x = lon, y = lat,

fill = N\_violation,

group = Place),

color = "grey40") +

scale\_fill\_material("grey", alpha = 0.7) +

geom\_line(data = data\_streets\_st[![is.na](http://is.na)(L1)],

aes(lon, lat,

group = L1,

color = N\_violations\_per\_street

),

size = 1.4

) +

scale\_color\_brewer(palette = "Reds") +

geom\_label(data = data\_streets\_st\_max,

aes(x = lon, y = lat,

label = paste0(Street\_edit, " - ", N\_violations)

),

color = "black", fill = "firebrick2",

alpha = 0.75, size = 4.5

) +

geom\_label(data = data\_poly\_ba\_mean\_place,

aes(x = lon, y = lat, label = Place,

fill = NA\_real\_),

color = "black", fill = "dodgerblue1",

alpha = 0.75, size = 4.5

) +

coord\_map(

xlim=c(data\_poly\_ba[.("Stare Mesto"), on = .(Place), min(lon)],

data\_poly\_ba[.("Stare Mesto"), on = .(Place), max(lon)]),

ylim= c(data\_poly\_ba[.("Stare Mesto"), on = .(Place), min(lat)],

data\_poly\_ba[.("Stare Mesto"), on = .(Place), max(lat)])

) +

transition\_states(reorder(Year\_M, Year\_M), transition\_length = 2,

state\_length = 1) +

labs(fill = "N\_vio\_district", color = "N\_vio\_street") +

guides(color = guide\_legend(ncol = 1, override.aes = list(size = 2),

title.position = "top"),

fill = guide\_legend(nrow = 2, title.position = "top")) +

theme\_bw() +

theme(axis.text = element\_blank(),

axis.line = element\_blank(),

axis.title = element\_blank(),

axis.ticks = element\_blank(),

legend.text = element\_text(size = 13),

legend.title = element\_text(size = 14, face = "bold"),

legend.background = element\_rect(fill = "white"),

legend.key = element\_rect(fill = "white"),

legend.position = "bottom")

# save animation to object

anim\_zoom <- animate(gg\_ba\_anim\_lines\_zoom, duration = 46, width = 700, height = 750)

**GIFS binding with magick package**

Now, we have to bind these two animations into one GIF. I will do it by functions from the magick package.

a\_mgif <- image\_read(anim\_whole)

b\_mgif <- image\_read(anim\_zoom)

new\_gif <- image\_append(c(a\_mgif[1],

b\_mgif[1]))

for(i in 2:100){

combined <- image\_append(c(a\_mgif[i],

b\_mgif[i]))

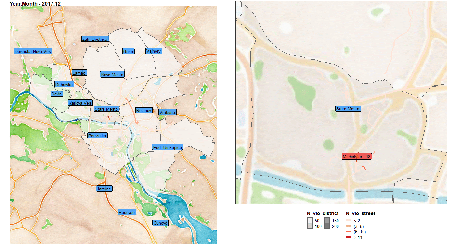
new\_gif <- c(new\_gif, combined)

}

# final!

new\_gif

Voilaaa:



For full resolution – right click on the gif and click on the view image option.

We can see that at the end of the observed period, so the summer of 2019, the Old-town district has even-more violations than usual. That can be caused by multiple factors…

Also notice, that peripheral areas of districts as Ruzinov and Vrakuna have streets with repeating multiple violations.

**Streets with the most violations**

Let’s see some of the most violated streets as time series as we seen in the animation.

times <- data.table(Year\_M = data\_violation\_agg\_street[, sort(unique(Year\_M))])

times[, Date := as.Date("2017-01-01")]

times[, row\_id := 1:.N]

times[, Date := Date + row\_id\*30.4]

data\_vio\_most\_street <- copy(data\_violation\_agg\_street[

.(c("Michalska", "Postova",

"Rybarska brana", "Namestie slobody")),

on = .(Street\_edit)])

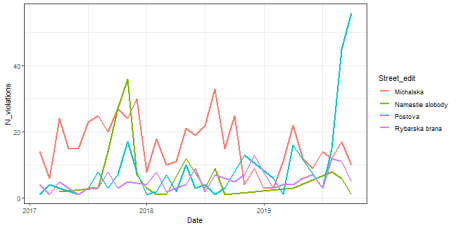
data\_vio\_most\_street[times, on = .(Year\_M), Date := i.Date]

setnames(data\_vio\_most\_street, "N", "N\_violations")

ggplot(data\_vio\_most\_street) +

geom\_line(aes(Date, N\_violations, color = Street\_edit), size = 0.8) +

theme\_bw()



The most of the time, Michalska street in the historical center of the Old-town was the most violated street, but the last three months is Postova the “winner of the most violated street”…this is maybe because of the new city-police station nearby.