# BIKE SHARING USAGE IN HAMBURG

### THE DATA

The map shows the bike sharing usage of StadtRAD, the bike sharing system in Hamburg – Germany. The data is available on the open data platform from Deutsche Bahn, the public railway company in Germany. The last new StadtRAD station was put into operation in May 2016, that is why a have chosen to display the usage of June 2016. The brighter the lines, the more bikes have been cycled along that street.

### THE PROCESSING

From data processing and spatial analysis to visualization the whole project was done in R. I have used the leaflet and shiny package to display the data interactively. The bikes themselves don't have GPS, so the routes are estimated on a fastest route basis using the awesome CycleStreets API. The biggest challenge has been the aggregation of overlapping routes. I found the overline function from the stplanr package very helpful. It converts a series of overlaying lines and aggregates their values for overlapping segments.

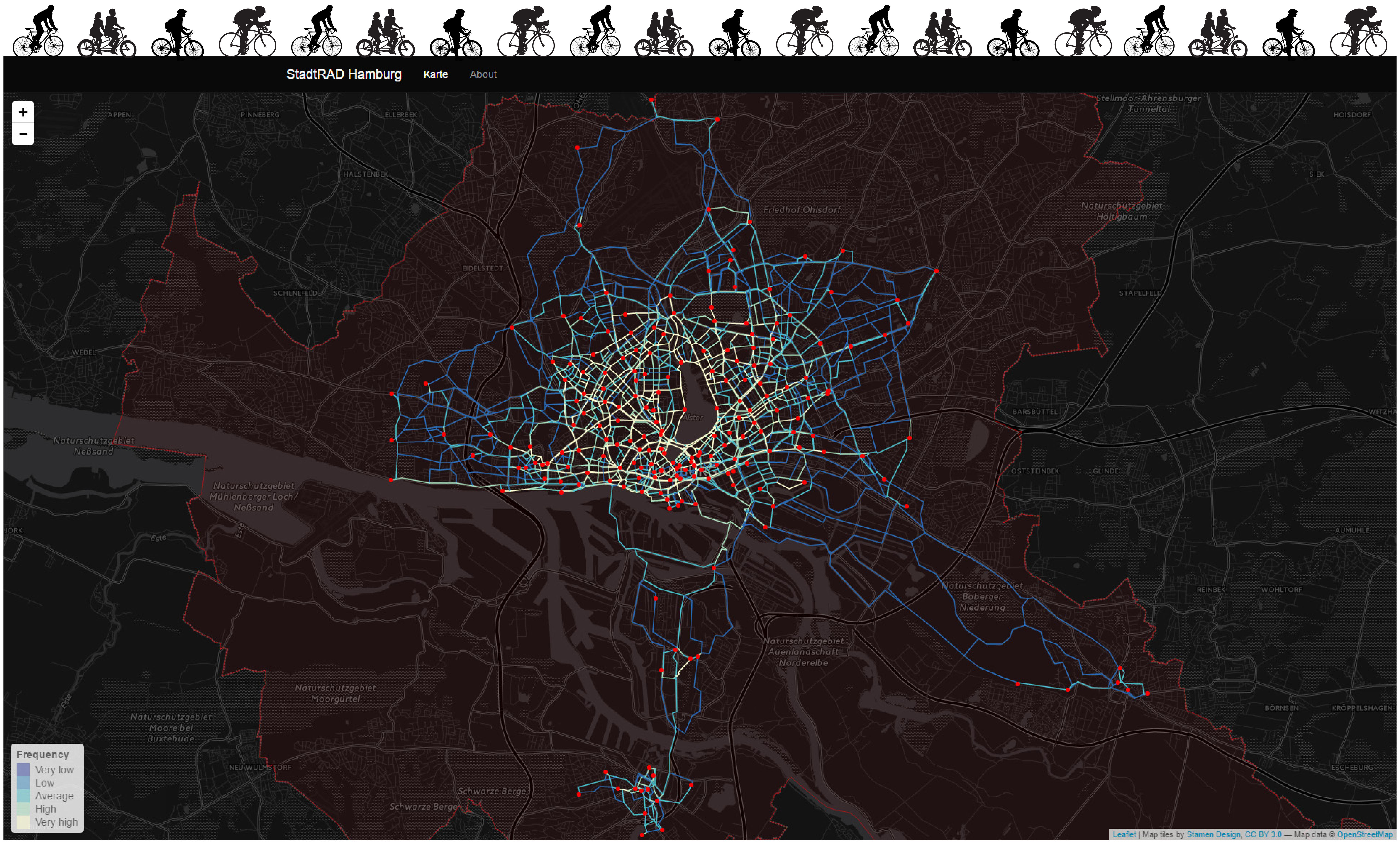
The raw data file from Deutsche Bahn is quite huge so I struggled to import the data into R to be able to process it. In the end the read.csv.sql function from the sqldf package did the job. This way I did not need to import the whole file and just could filter out the bike rides for Hamburg.

The code could easily be used to map other spatial data, for example the car sharing data from car2go which is available via their API. This might be a future project.

### THE MAP

As a cycling ethusiast and Hamburg native I have been riding the streets of Hamburg for a long time now. Over the years I found my favorite cycle routes throughout the city but also know the tight and problematic corners of Hamburg, where missing or overcrowded cycling paths bring you too close to other bikers, cars or pedastrians. It is amazing to see that the data set can proof some of my hypothesis about the current state of the bicylce infrastructure in Hamburg and even bring up new questions I have not even thought about before.

When you look at the map you can see a widely spread bike sharing network over big parts of the city but also notice some enclaving processes where missing stations disconnect bike riders from the high frequented and well connected city center. As the Elbe river seperates Hamburg in a northern and southern part it seems like bike sharing became a well accepted means of transportation to keep both parts of the city connected.



Leaflet | Map tiles by Stamen Design, CC BY 3.0 — Map data © OpenStreetMap

```
##########################################################################################
# IMPORT AND PROCESS DATA
##########################################################################################

# load packages
x = c("sqldf","dplyr","data.table","sp","rgdal","stplanr","reshape2","rmapshaper","leaflet","RColorBrewer")
lapply(x, require, character.only = T)

# import bike rentals
mydata = read.csv.sql("HACKATHON_BOOKING_CALL_A_BIKE.csv",
                      sql = "select * from file where CITY_RENTAL_ZONE = 'Hamburg' ", sep = ";")

# filter on time period
mydata$DATE_FROM = as.POSIXct(strptime(mydata$DATE_FROM, "%Y-%m-%d %H:%M:%S"))
mydata = filter(mydata, DATE_FROM >= "2016-06-01 00:00:00" & DATE_FROM <= "2016-06-30 23:59:59")

# aggregate doubles
mydata = transform(mydata, min = pmin(as.character(START_RENTAL_ZONE_GROUP), as.character(END_RENTAL_ZONE_GROUP)))
mydata = transform(mydata, max = pmax(as.character(START_RENTAL_ZONE_GROUP), as.character(END_RENTAL_ZONE_GROUP)))

# get lan/lat from stations
station = read.csv("HACKATHON_RENTAL_ZONE_CALL_A_BIKE.csv", sep = ";")
station = filter(station, CITY == "Hamburg")

# merge station coordinates with bike rentals
mydata = merge(mydata, station, by.x = "min", by.y = "RENTAL_ZONE_GROUP", all.x = T)
mydata = merge(mydata, station, by.x = "max", by.y = "RENTAL_ZONE_GROUP", all.x = T)

# count bike rides for each route (combine lat/lon)
mydata$start = paste(mydata$RENTAL_ZONE_Y_COORDINATE.x, mydata$RENTAL_ZONE_X_COORDINATE.x, sep = " ")
mydata$dest = paste(mydata$RENTAL_ZONE_Y_COORDINATE.y, mydata$RENTAL_ZONE_X_COORDINATE.y, sep = " ")
mydata = mydata %>% group_by(start, dest) %>% summarise(count = n())

# split lat/lon into two columns
mydata$id = rownames(mydata)
mydata = melt(mydata, id.vars = c("id","count"))
test = data.frame(do.call('rbind', strsplit(as.character(mydata$value),' ', fixed=T)))
mydata = cbind(mydata, test)
mydata = select(mydata, X1, X2, id, count)
colnames(mydata) = c("lat","lon","id","count")
```

```
##########################################################################################
# GET ROUTES FROM CYCLESTREETS API
##########################################################################################

# transform to SPDF
dt = as.data.table(mydata)
lst_lines = lapply(unique(dt$id), function(x){
  Lines(Line(dt[id == x, .(lon, lat)]), ID = x)
})

spl_lst = SpatialLines(lst_lines)
spl_df = SpatialLinesDataFrame(spl_lst, data.frame(mydata$count))

# get routes from cyclestreet API
spl_df = line2route(spl_df, "route_cyclestreet", plan = "fastest")

# add ride count for each route
spl_df@data$count = mydata$count

# simplify SLDF
spl_df = ms_simplify(input = spl_df, keep = 0.01)

# aggregate ride counts on overlapping routes
spl_df = overline(spl_df, attrib = "count", fun = sum)

##########################################################################################
# CREATE LEAFLET MAP
##########################################################################################

# load hamburg shape for the map
hhshape = readOGR(dsn = ".", layer = "HH_ALKIS_Landesgrenze")

# create color palette
qpal = colorQuantile(rev(brewer.pal(4, "YlGnBu")), NULL, n = 4)

# create leaflet map
leaflet(spf_df) %>%
  addPolygons(data = hhshape, stroke = T, smoothFactor = 0.05, fillOpacity = 0.05, color = "red", weight = 1) %>%
  addPolylines(color = qpal(spf_df@data$count),opacity = 1, weight = 1.5) %>%
  addCircleMarkers(lng = station$RENTAL_ZONE_X_COORDINATE, lat = station$RENTAL_ZONE_Y_COORDINATE,
                   fillOpacity = 100, color = "red", stroke = F, radius = 3) %>%
  addLegend(position = 'bottomleft',colors =  rev(brewer.pal(5, "YlGnBu")),
            labels = c("Very low","Low","Average","High","Very high"), title = 'Frequency')
```

github/kruse-alex

@krusealex2013