Big Data Analytics

Lecture 8:

Visualization

Prof. Dr. Ulrich Matter

29/04/2021

Updates

Status

1. Introduction: Big Data, Data Economy. Walkowiak (2016): Chapter 1.
2. Computation and Memory in Applied Econometrics.
3. Computation and Memory in Applied Econometrics II.*
4. Advanced R Programming. Wickham (2019): Chapters 2, 3, 17,23, 24.
5. Advanced R Programming II. Wickham (2019): Chapters 2, 3, 17,23, 24.
6. Cleaning, Transformation, and Aggregation of Big Data. Walkowiak (2016): Chapter 3: p. 74-127.
7. Data Data Storage, Databases Interaction with R. Walkowiak (2016): Chapter 5.
8. **Data Visualization. Wickham et al.(2015); Schwabish (2014).**
9. Cloud Computing, Distributed Systems, Applied Econometrics with Spark. Walkowiak (2016): Chapter 4.
10. Project Presentations.
11. Project Presentations. Q&A, Feedback.

Project Presentations: 20.5.2021

- Team Algorithm
- SciencePro
- Patrons Association
- Team Send It
- Lord of the R
- Random Forest

Project Presentations: 27.5.2021

- Blue Data
- Team Data Digger
- Team Significant
- Team Nordic
- Diamond Hands

(Big) Data Visualization

ggplot2

- ‘Grammar of Graphics’
- Build plots layer-by-layer
- Here: Usefull tool for explorative visualization
- In-memory operations
    - Works well with 1 million obs.

Data import

```
# load packages
library(data.table)

# import data into RAM (needs around 200MB)
taxi <- fread("../data/tlc_trips.csv",
              nrows = 1000000)
```

Data preparation

We prepare/clean the data as in the `ff`-approach above.

```
# first, we remove the empty vars V8 and V9
taxi$V8 <- NULL
taxi$V9 <- NULL


# set covariate names according to the data dictionary
# see https://www1.nyc.gov/assets/tlc/downloads/pdf/data_dictionary_trip_records_yellow.pdf
# note instead of taxizonne ids, long/lat are provided

varnames <- c("vendor_id",
              "pickup_time",
              "dropoff_time",
              "passenger_count",
              "trip_distance",
              "start_lat",
              "start_long",
              "dest_lat",
              "dest_long",
              "payment_type",
              "fare_amount",
              "extra",
              "mta_tax",
              "tip_amount",
              "tolls_amount",
              "total_amount")
```

Exploration: what determines tip amounts?

Set up the canvas…

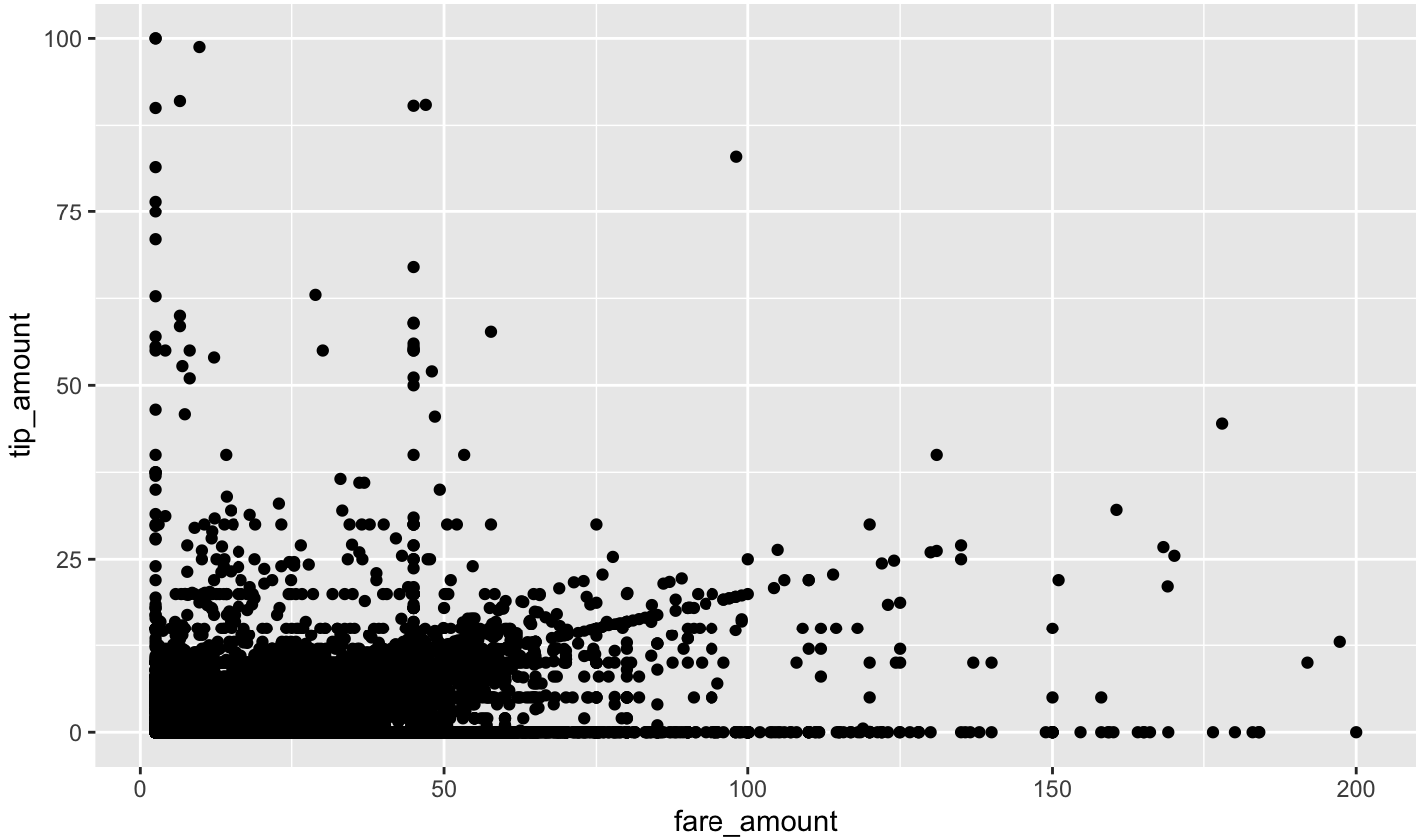
```
# load packages
library(ggplot2)

# set up the canvas
taxiplot <- ggplot(taxi, aes(y=tip_amount, x= fare_amount))
taxiplot
```

Exploration: what determines tip amounts?

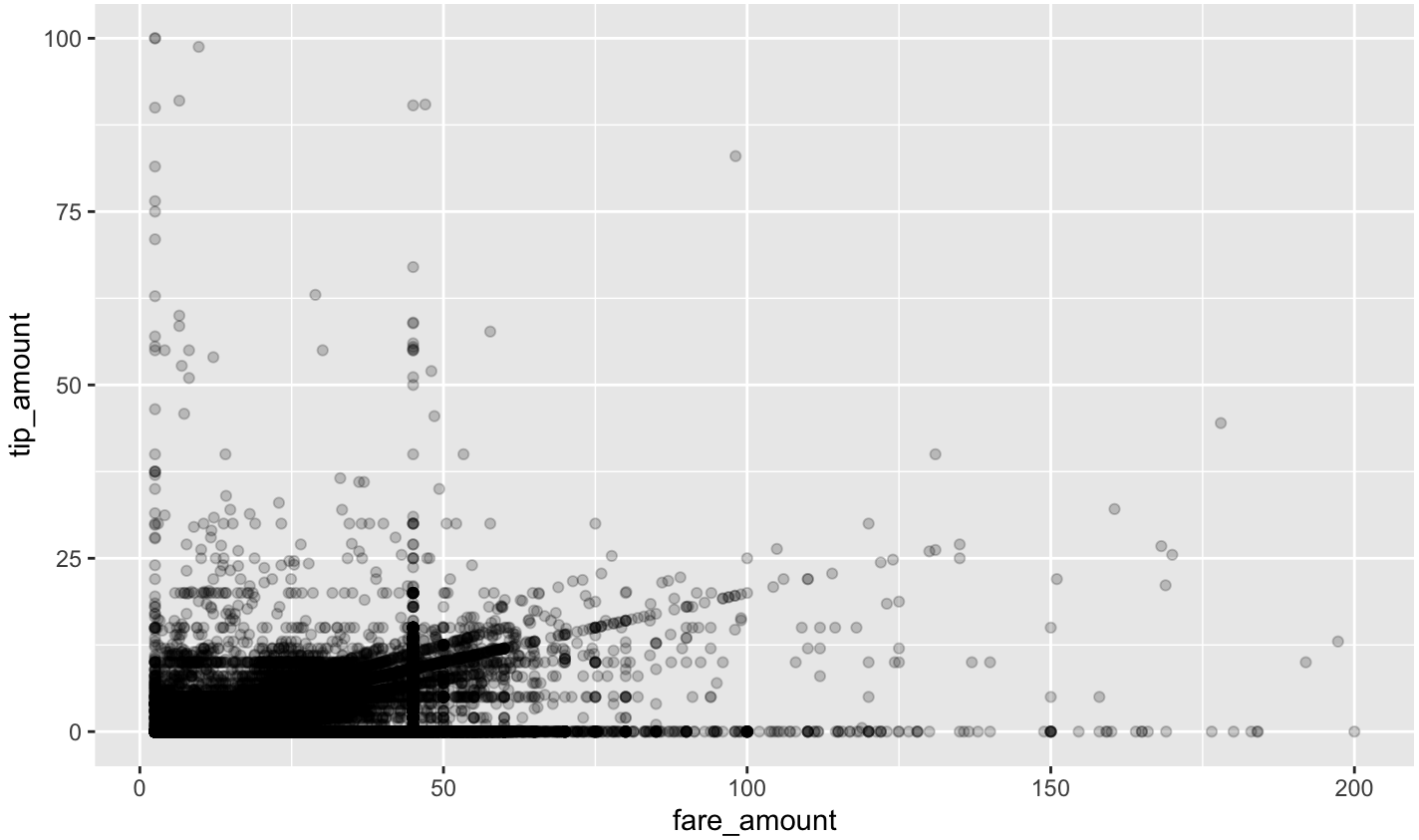
Visualize the co-distribution of the two variables with a simple scatter-plot.

```
# simple x/y plot
taxiplot +
     geom_point()
```



Problem: too many points

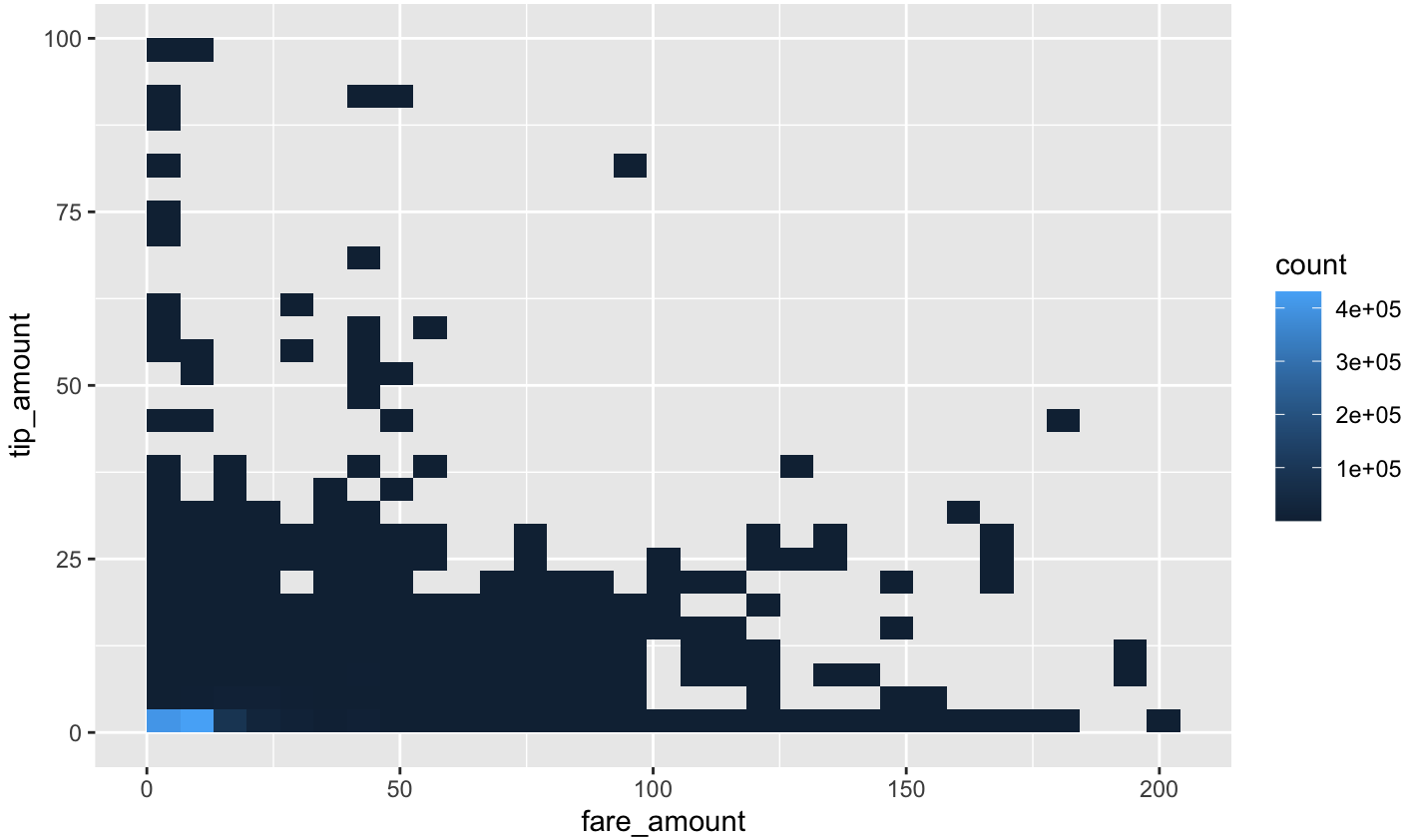
```
# simple x/y plot
taxiplot +
     geom_point(alpha=0.2)
```



2-D bins

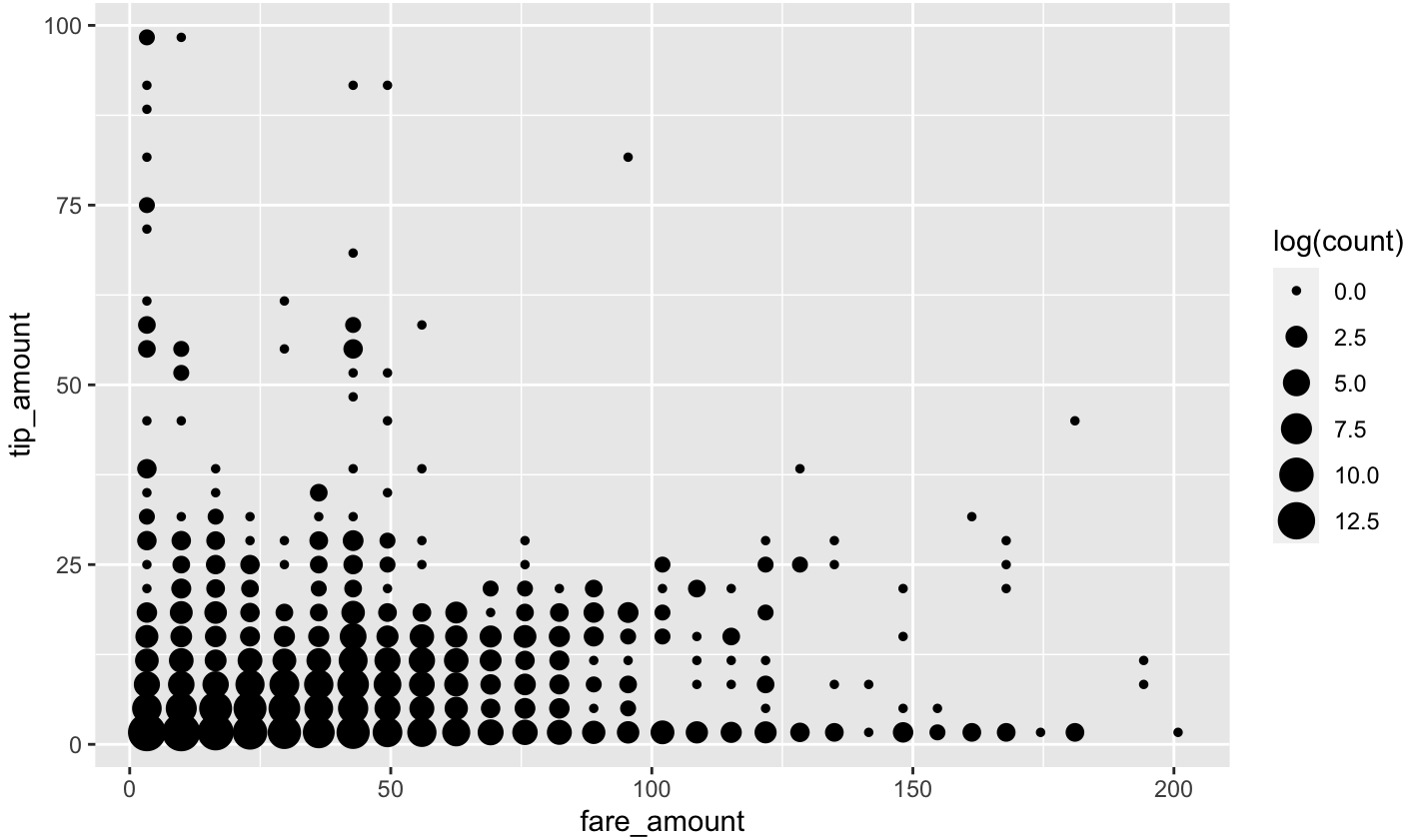
Where are most observations located?

```
# 2-dimensional bins
taxiplot +
     geom_bin2d()
```



2-D bins: ln of count

```
# 2-dimensional bins
taxiplot +
     stat_bin_2d(geom="point",
                 mapping= aes(size = log(..count..))) +
     guides(fill = FALSE)
```

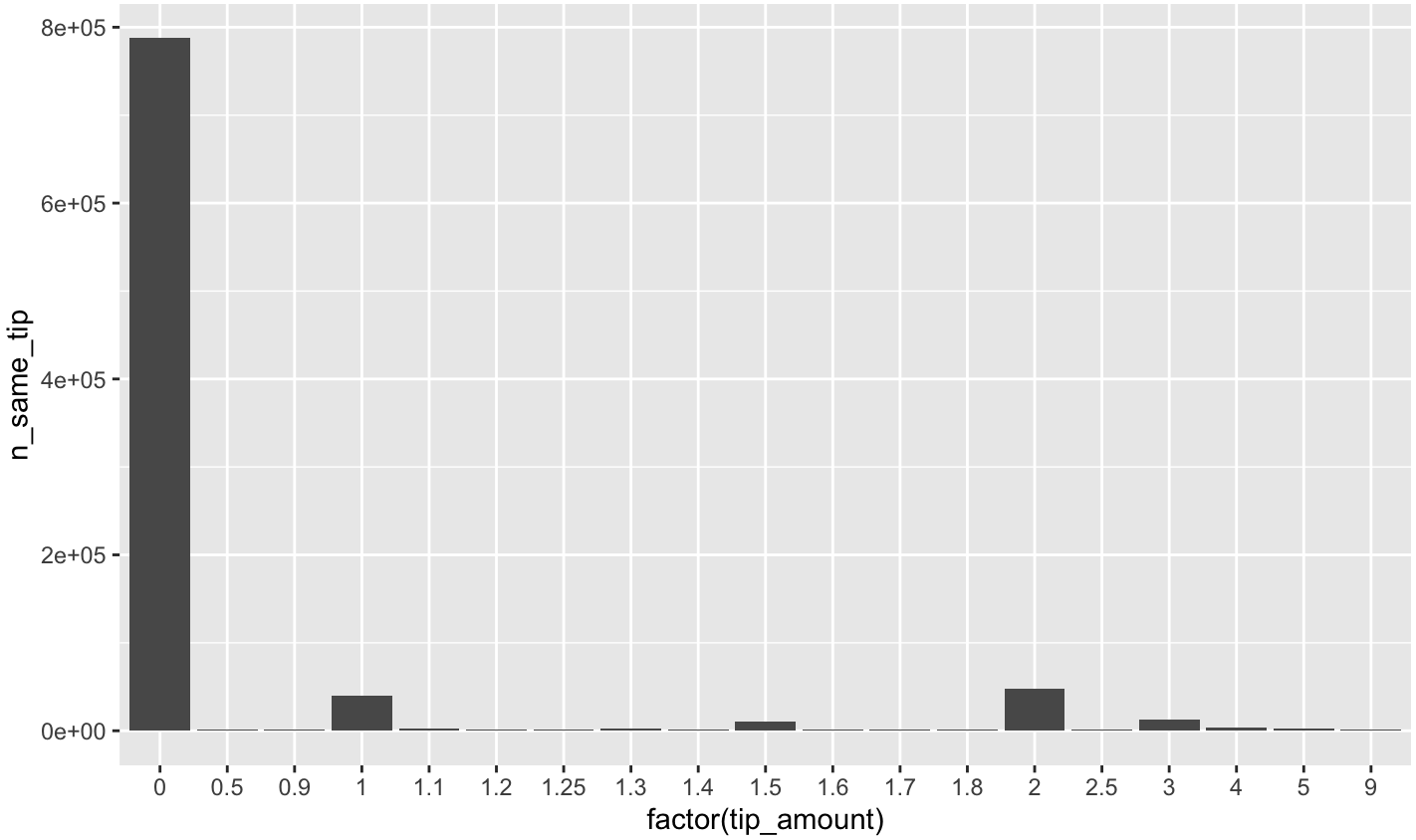


Frequencies

```
# compute frequency of per tip amount and payment method
taxi[, n_same_tip:= .N, by= c("tip_amount", "payment_type")]
frequencies <- unique(taxi[payment_type %in% c("credit", "cash"),
                           c("n_same_tip", "tip_amount", "payment_type")][order(n_same_tip, decreasi
```

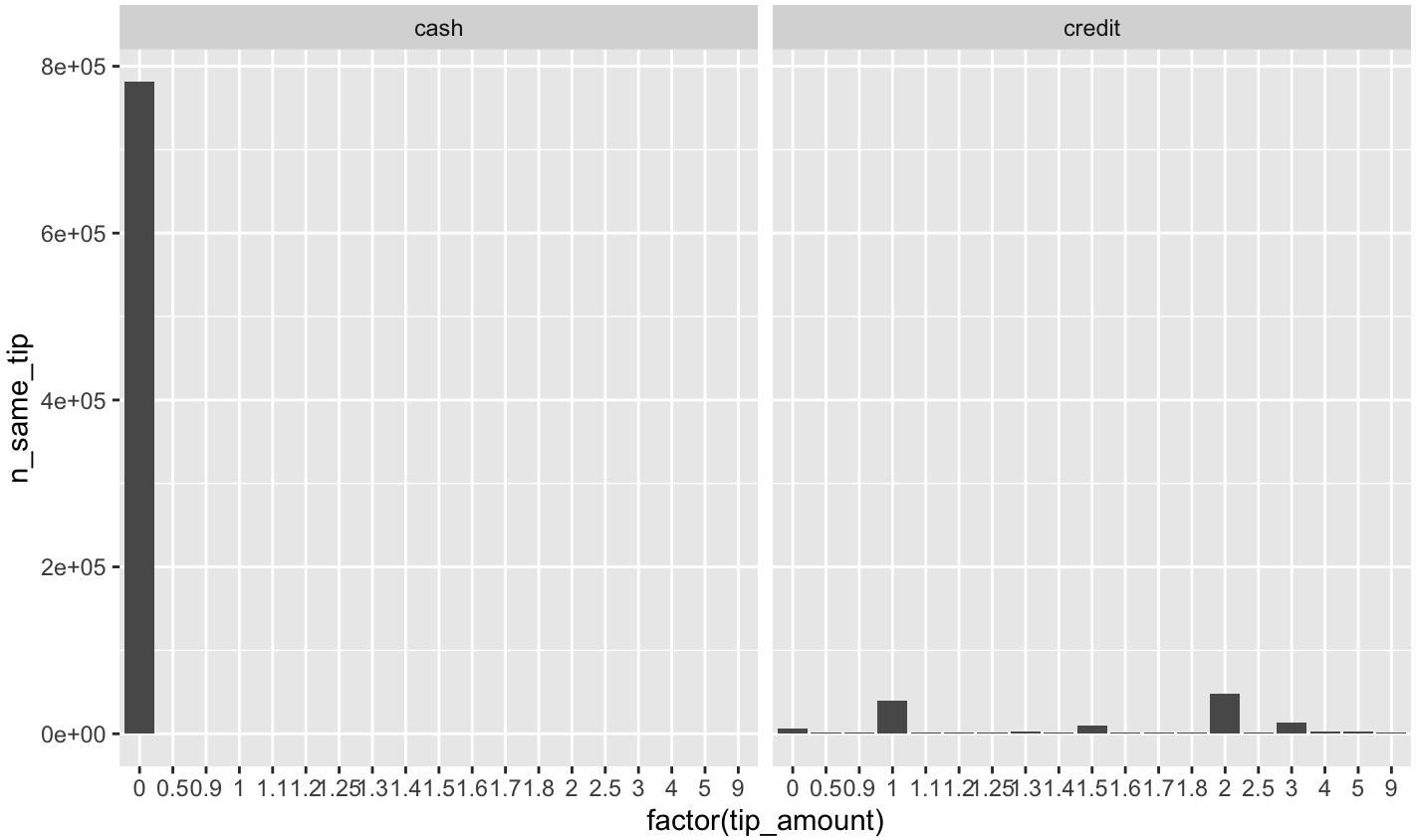
Frequencies

```
# plot top 20 frequent tip amounts
fare <- ggplot(data = frequencies[1:20], aes(x = factor(tip_amount), y = n_same_tip))
fare + geom_bar(stat = "identity")
```



Split by payment type

```
fare + geom_bar(stat = "identity") +
     facet_wrap("payment_type")
```



Payment habits?

Fractions of dollars due to loose change as tip?

```
# indicate natural numbers
taxi[, dollar_paid := ifelse(tip_amount == round(tip_amount,0), "Full", "Fraction"),]


# extended x/y plot
taxiplot +
     geom_point(alpha=0.2, aes(color=payment_type)) +
     facet_wrap("dollar_paid")
```

Payment habits?

Rounding up?

```
taxi[, rounded_up := ifelse(fare_amount + tip_amount == round(fare_amount + tip_amount, 0),
                            "Rounded up",
                            "Not rounded")]
# extended x/y plot
taxiplot +
     geom_point(data= taxi[payment_type == "credit"],
                alpha=0.2, aes(color=rounded_up)) +
     facet_wrap("dollar_paid")
```

Modelling of payment habits

‘X% tip rule’?

```
modelplot <- ggplot(data= taxi[payment_type == "credit" & dollar_paid == "Fraction" & 0 < tip_amount
                    aes(x = fare_amount, y = tip_amount))
modelplot +
     geom_point(alpha=0.2, colour="darkgreen") +
     geom_smooth(method = "lm", colour = "black")


## `geom_smooth()` using formula 'y ~ x'
```

Prepare the plot for reporting

```
modelplot <- ggplot(data= taxi[payment_type == "credit" & dollar_paid == "Fraction" & 0 < tip_amount
                    aes(x = fare_amount, y = tip_amount))
modelplot +
     geom_point(alpha=0.2, colour="darkgreen") +
     geom_smooth(method = "lm", colour = "black") +
     ylab("Amount of tip paid (in USD)") +
     xlab("Amount of fare paid (in USD)") +
     theme_bw(base_size = 18, base_family = "serif")
```

```
## `geom_smooth()` using formula 'y ~ x'
```

Data Visualization Part II

Visualization of spatial data with `ggplot2`

- Data source: NYC Taxi & Limousine Commission (TLC).
- Data on all trip records including **pick-up and drop-off times/locations**.

Preparations

- Load packages for GIS data/operations

```
# load GIS packages
library(rgdal)
library(rgeos)
```

Download map data

```
# download the zipped shapefile to a temporary file, unzip
URL <- "https://www1.nyc.gov/assets/planning/download/zip/data-maps/open-data/nycd_19a.zip"
tmp_file <- tempfile()
download.file(URL, tmp_file)
file_path <- unzip(tmp_file, exdir= "../data")
# delete the temporary file
unlink(tmp_file)
```

Import map data

```
# read GIS data
nyc_map <- readOGR(file_path[1], verbose = FALSE)

# have a look at the polygons that constitute the map
summary(nyc_map)
```

```
## Object of class SpatialPolygonsDataFrame
## Coordinates:
##        min       max
## x 913175.1 1067382.5
## y 120121.9  272844.3
## Is projected: TRUE
## proj4string :
## [+proj=lcc +lat_1=41.03333333333333 +lat_2=40.66666666666666 +lat_0=40.16666666666666
## +lon_0=-74 +x_0=300000.0000000001 +y_0=0 +ellps=GRS80 +towgs84=0,0,0,0,0,0,0 +units=us-ft
## +no_defs]
## Data attributes:
##      BoroCD        Shape_Leng       Shape_Area
##  Min.   :101.0   Min.   : 23963   Min.   : 24293239
##  1st Qu.:205.5   1st Qu.: 36611   1st Qu.: 48407357
##  Median :308.0   Median : 52246   Median : 82702417
##  Mean   :297.2   Mean   : 74890   Mean   :118724012
##  3rd Qu.:405.5   3rd Qu.: 85711   3rd Qu.:136615357
##  Max.   :595.0   Max.   :270660   Max.   :599062130
```

Change map projection

```
# transform the projection
nyc_map <- spTransform(nyc_map,
                       CRS("+proj=longlat +datum=WGS84 +no_defs +ellps=WGS84 +towgs84=0,0,0"))
# check result
summary(nyc_map)
```

```
## Object of class SpatialPolygonsDataFrame
## Coordinates:
##         min       max
## x -74.25559 -73.70001
## y  40.49612  40.91553
## Is projected: FALSE
## proj4string :
## [+proj=longlat +datum=WGS84 +no_defs +ellps=WGS84 +towgs84=0,0,0]
## Data attributes:
##      BoroCD        Shape_Leng        Shape_Area
##  Min.   :101.0   Min.   : 23963   Min.   : 24293239
##  1st Qu.:205.5   1st Qu.: 36611   1st Qu.: 48407357
##  Median :308.0   Median : 52246   Median : 82702417
##  Mean   :297.2   Mean   : 74890   Mean   :118724012
##  3rd Qu.:405.5   3rd Qu.: 85711   3rd Qu.:136615357
##  Max.   :595.0   Max.   :270660   Max.   :599062130
```

Prepare map for plotting with `ggplot2`

```
nyc_map <- fortify(nyc_map)
```

Prepare pick-up and drop-off data

```
# taxi trips plot data
taxi_trips <- taxi[start_long <= max(nyc_map$long) &
                       start_long >= min(nyc_map$long) &
                       dest_long <= max(nyc_map$long) &
                       dest_long >= min(nyc_map$long) &
                       start_lat <= max(nyc_map$lat) &
                       start_lat >= min(nyc_map$lat) &
                       dest_lat <= max(nyc_map$lat) &
                       dest_lat >= min(nyc_map$lat)
                       ]
taxi_trips <- taxi_trips[sample(nrow(taxi_trips), 50000)]
```

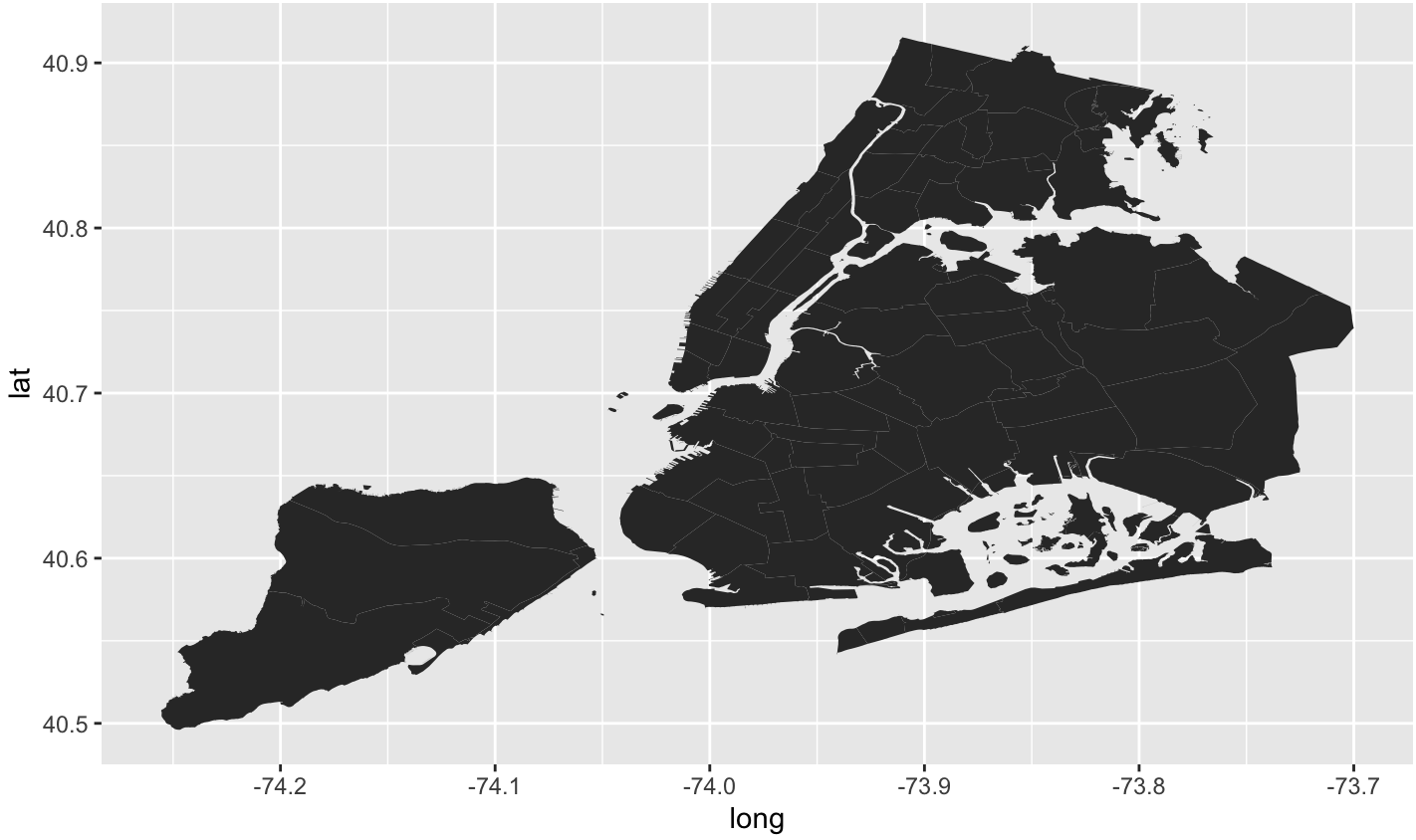
Code time dimension(s)

```
taxi_trips$start_time <- hour(taxi_trips$pickup_time)


# define new variable for facets
taxi_trips$time_of_day <- "Morning"
taxi_trips[start_time > 12 & start_time < 17]$time_of_day <- "Afternoon"
taxi_trips[start_time %in% c(17:24, 0:5)]$time_of_day <- "Evening/Night"
taxi_trips$time_of_day  <- factor(taxi_trips$time_of_day, levels = c("Morning", "Afternoon", "Evenin
```

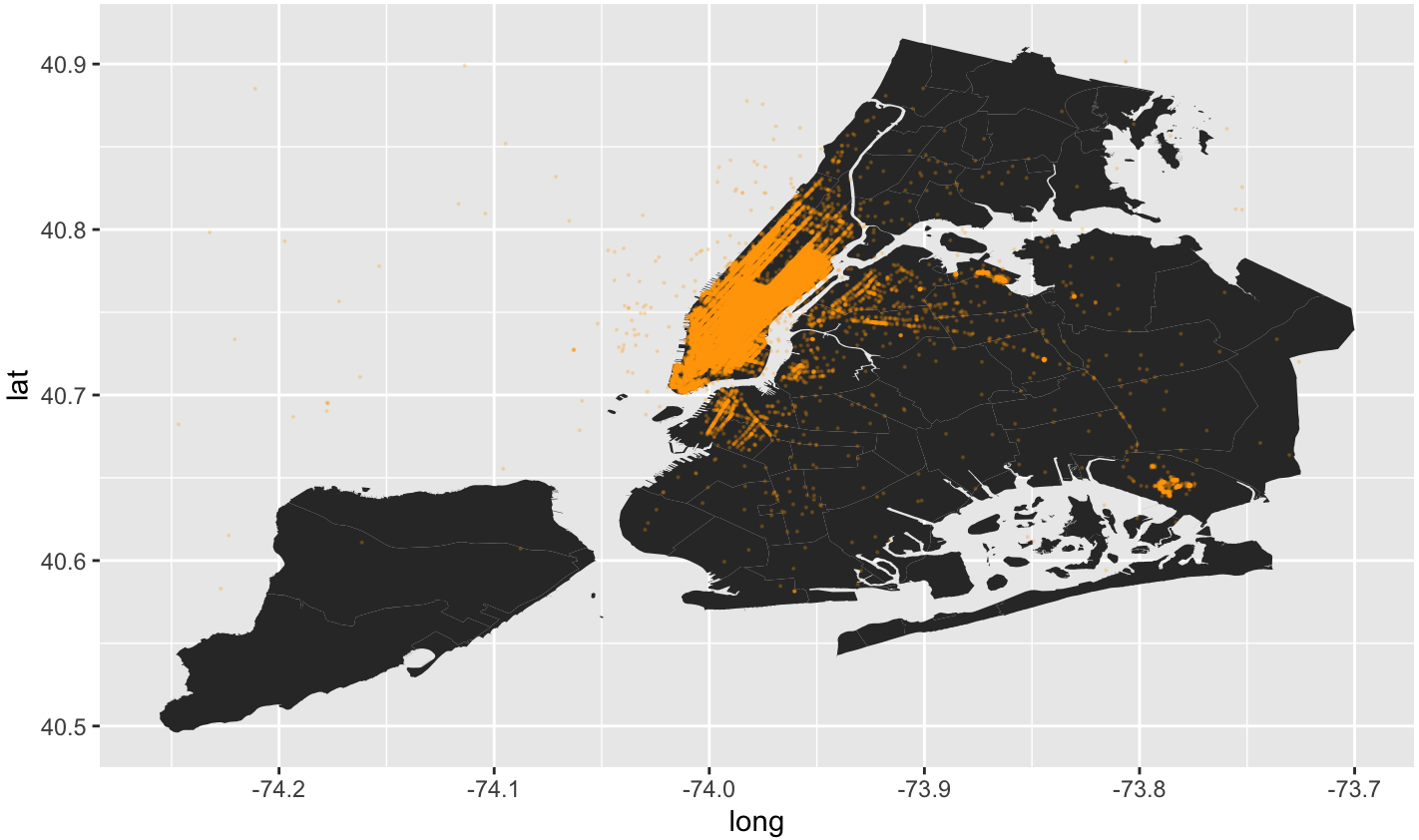
Base plot: Map of NYC

```
# set up the canvas
locations <- ggplot(taxi_trips, aes(x=long, y=lat))
# add the map geometry
locations <- locations + geom_map(data = nyc_map,
                                  map = nyc_map,
                                  aes(map_id = id))
locations
```



Add pick-up locations

```
# add pick-up locations to plot
locations +
     geom_point(aes(x=start_long, y=start_lat),
                color="orange",
                size = 0.1,
                alpha = 0.2)
```

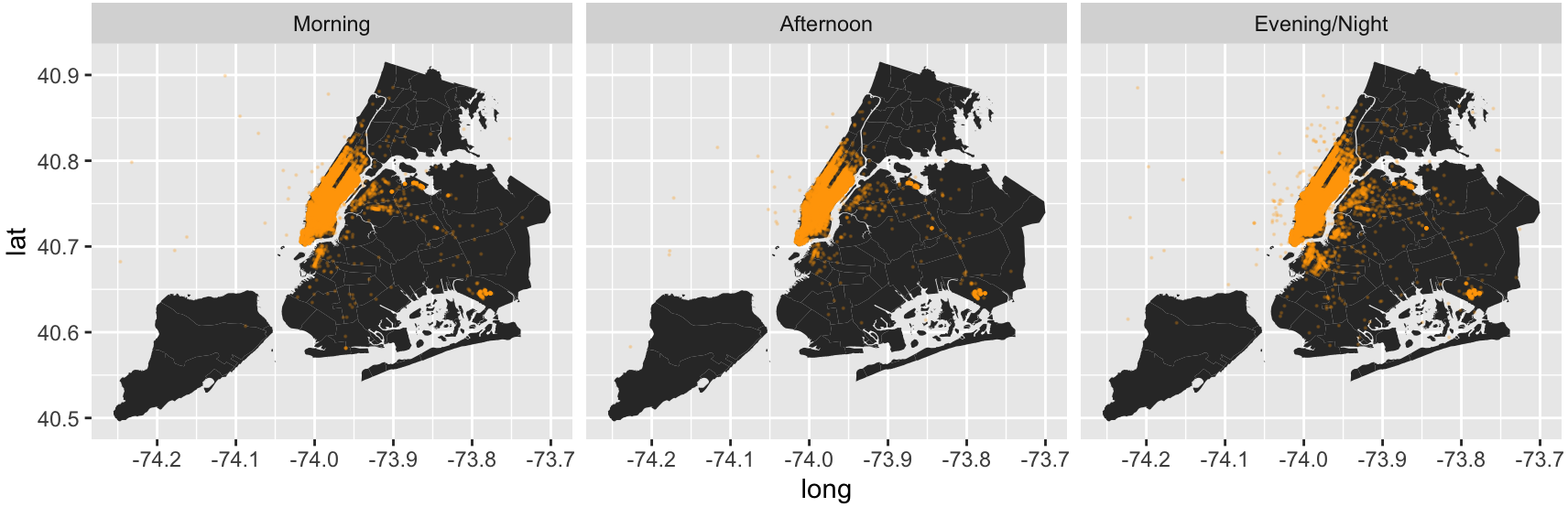


Add drop-off locations

```
# add pick-up locations to plot
locations +
     geom_point(aes(x=dest_long, y=dest_lat),
                color="steelblue",
                size = 0.1,
                alpha = 0.2) +
     geom_point(aes(x=start_long, y=start_lat),
                color="orange",
                size = 0.1,
                alpha = 0.2)
```

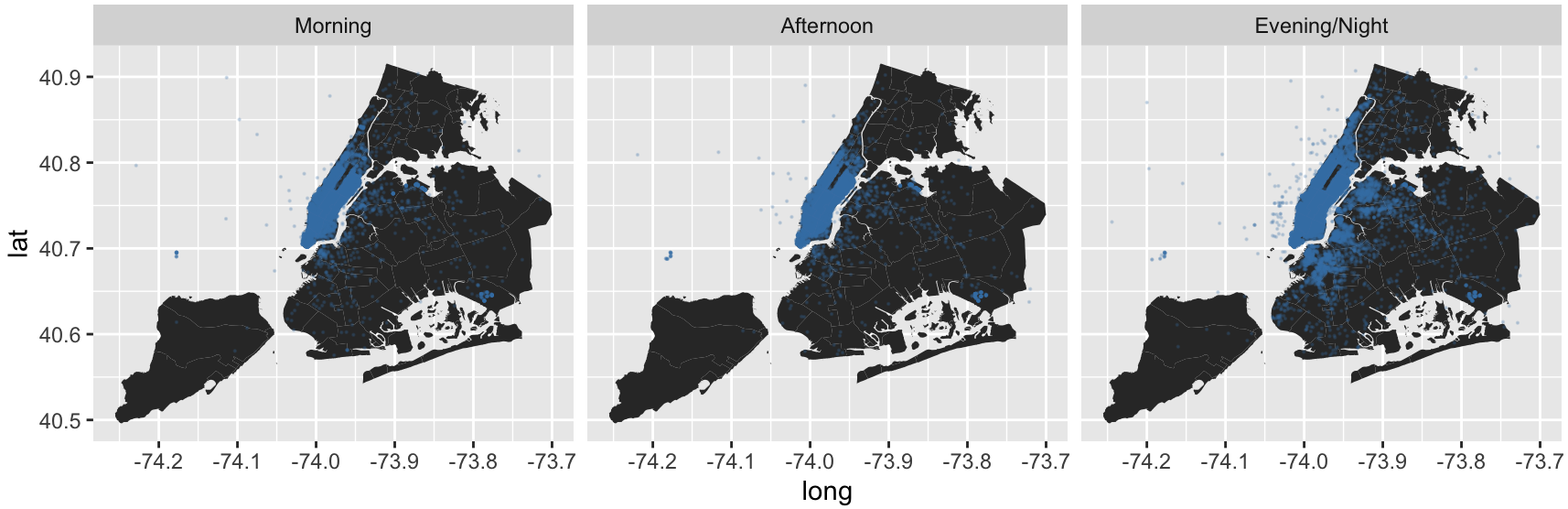
Taxi traffic over the course of a day

```
# pick-up locations
locations +
     geom_point(aes(x=start_long, y=start_lat),
                color="orange",
                size = 0.1,
                alpha = 0.2) +
     facet_wrap(vars(time_of_day))
```



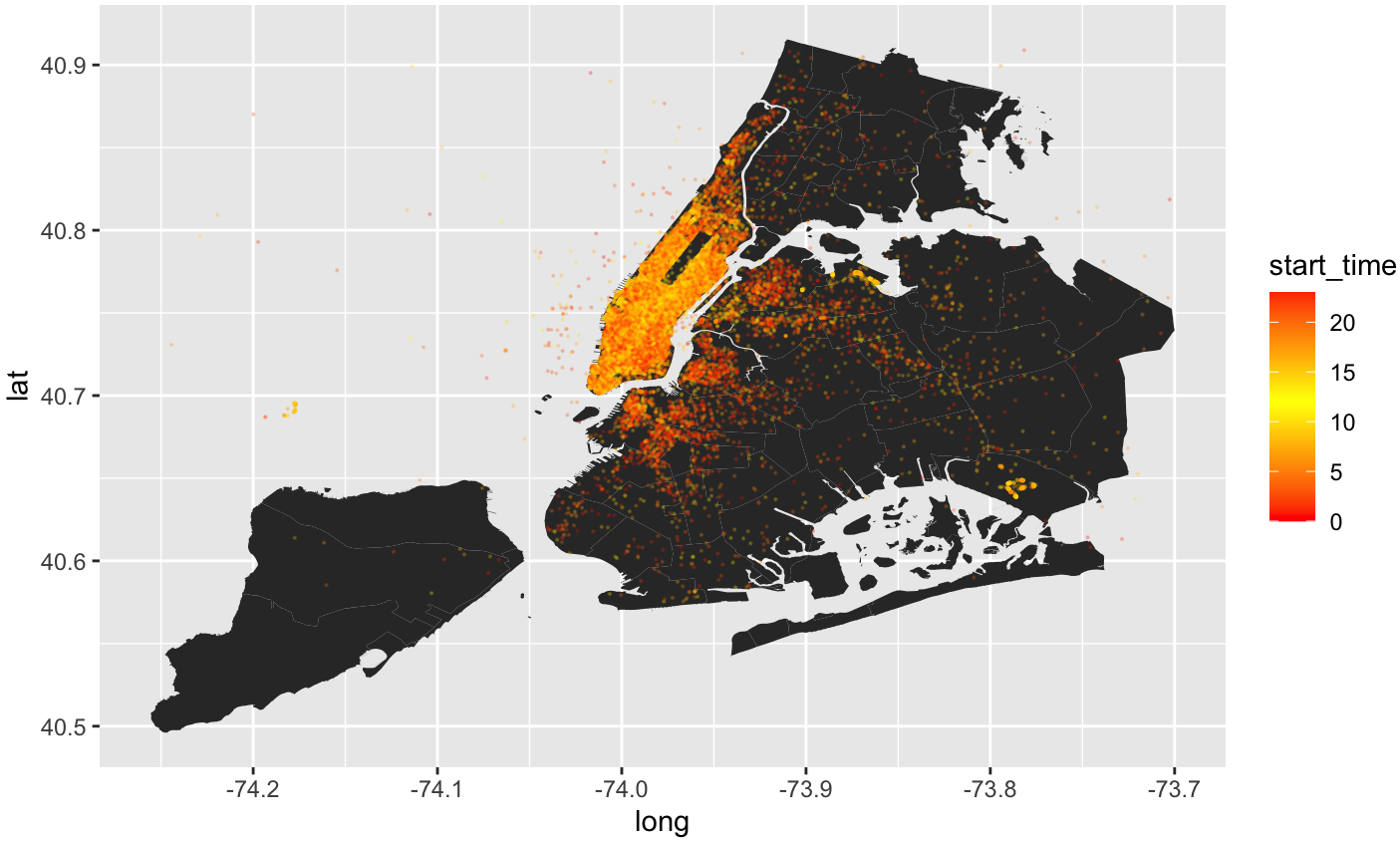
Taxi traffic over the course of a day

```
# drop-off locations
locations +
     geom_point(aes(x=dest_long, y=dest_lat),
                color="steelblue",
                size = 0.1,
                alpha = 0.2) +
     facet_wrap(vars(time_of_day))
```



Taxi traffic over the course of a day

```
# drop-off locations
locations +
     geom_point(aes(x=dest_long, y=dest_lat, color = start_time ),
                size = 0.1,
                alpha = 0.2) +
     scale_colour_gradient2( low = "red", mid = "yellow", high = "red",
                             midpoint = 12)
```



References