Analysis of the Triboro line

Requirements

Libraries

```
library(tidyverse)
library(igraph)
library(sf)
```

Files

```
nys_od <- readr::read_csv('data/ny_od_main_JT00_2019.csv')
nyc_nta_borders <- sf::st_read('data/nyc_2010_nta_borders.geojson')
```

```
## Reading layer `nyc_2010_nta_borders' from data source
##   `/home/miller/GeoI/fall-2022/gtech_705-spatial_anlysis/triboro-line/brooklyn-lodes/data/nyc_2010_n
##   using driver `GeoJSON'
## Simple feature collection with 195 features and 8 fields
## Geometry type: MULTIPOLYGON
## Dimension:     XY
## Bounding box:  xmin: -74.25559 ymin: 40.49614 xmax: -73.70001 ymax: 40.91554
## Geodetic CRS:  WGS 84
```

```
nyc_nta_tract_equiv <- readxl::read_xlsx('data/nyc_2010_census_tract_nta_equiv.xlsx')

subway_lines <- readr::read_csv('data/nta-subway-lines.csv')
subway_times <- readr::read_csv('data/nta-subway-times.csv')
driving_times <- readr::read_csv('data/nta-driving-times.csv')
walking_times <- readr::read_csv('data/nta-walking-times.csv')
```

Data

```
bk_name <- "Brooklyn"
bk_county_code <- "047"
bk_parks <- "BK99"

bk_nta_border <- nyc_nta_borders %>%
  dplyr::filter(BoroName == bk_name)  %>%
  dplyr::filter(NTACode != bk_parks) %>%
  dplyr::select("NTACode")

bk_nta_tract_equiv <- nyc_nta_tract_equiv %>%
  dplyr::filter(borough_name == bk_name) %>%
  dplyr::filter(nta_code != bk_parks) %>%
  dplyr::rename(tract = census_tract) %>%
  dplyr::select("tract", "nta_code")

od <- nys_od %>%
  dplyr::filter(
    stringr::str_sub(as.character(w_geocode), 3, 5) == bk_county_code &
    stringr::str_sub(as.character(h_geocode), 3, 5) == bk_county_code
  ) %>%
  dplyr::mutate(
```

```
   w_tract = stringr::str_sub(as.character(w_geocode), 6, 11)
  ) %>%
  dplyr::mutate(
   h_tract = stringr::str_sub(as.character(h_geocode), 6, 11)
  ) %>%
  dplyr::select("w_tract", "h_tract", "S000") %>%
  dplyr::left_join(bk_nta_tract_equiv, c("w_tract" = "tract")) %>%
  dplyr::rename(w_nta_code = nta_code) %>%
  dplyr::left_join(bk_nta_tract_equiv, c("h_tract" = "tract")) %>%
  dplyr::rename(h_nta_code = nta_code) %>%
  dplyr::select("h_nta_code", "w_nta_code", "S000") %>%
  dplyr::filter(w_nta_code != bk_parks & h_nta_code != bk_parks)
```

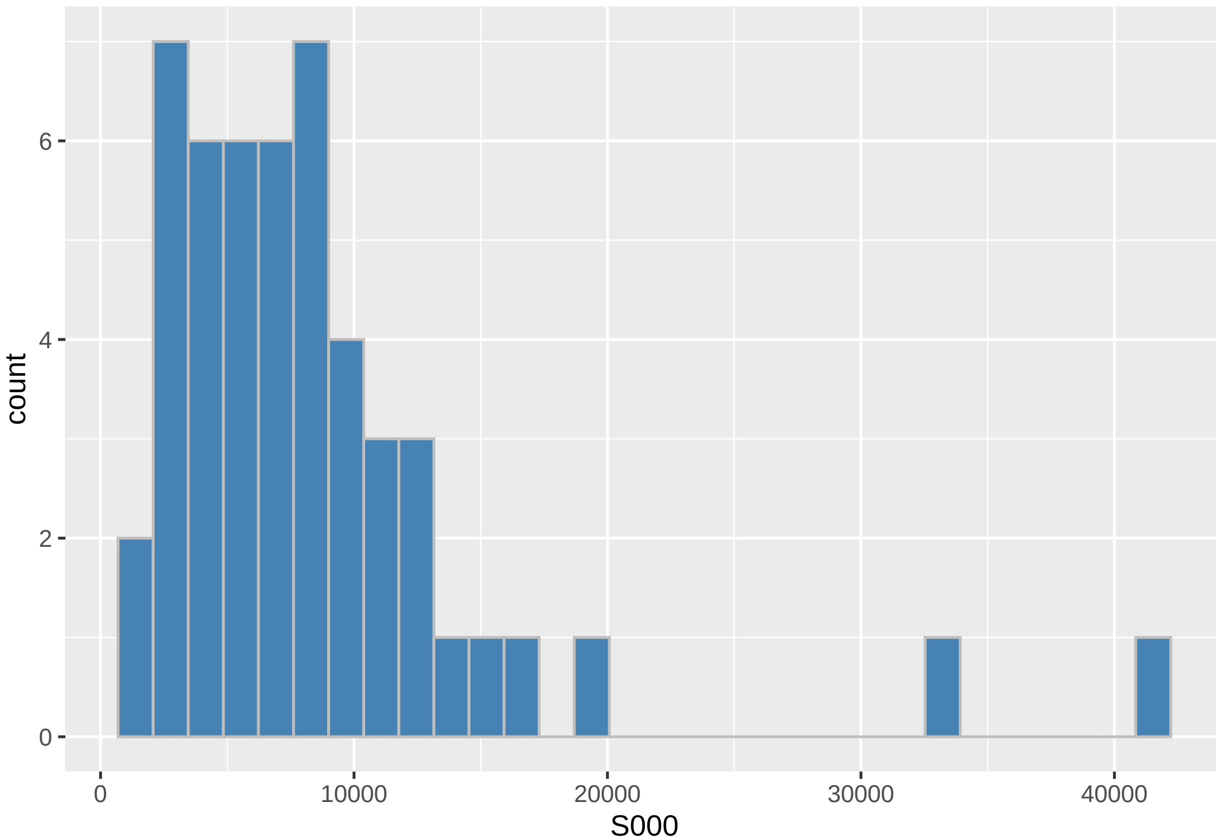
Exploratory data analysis

Distribution of job counts

```
job_counts <- od %>%
  dplyr::group_by(w_nta_code) %>%
  dplyr::summarise(
    S000 = sum(S000)
  ) %>%
  unique() %>%
  mutate(
    log_S000 = log(S000)
  )
```

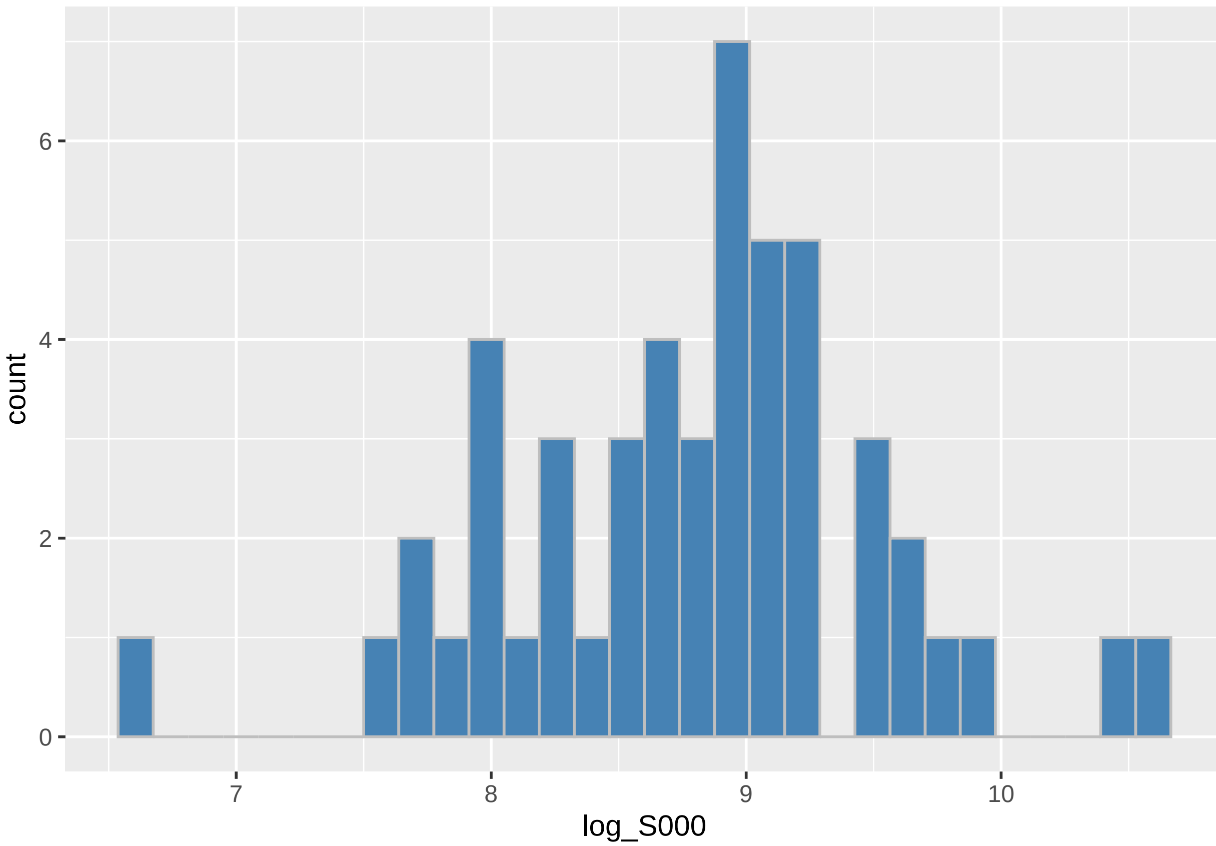
Original

```
ggplot(job_counts) +
  geom_histogram(aes(x = S000), fill = "steelblue", color = "grey", bins = "30")
```



Natural log

```
ggplot(job_counts) +
  geom_histogram(aes(x = log_S000), fill = "steelblue", color = "grey", bins = "30")
```

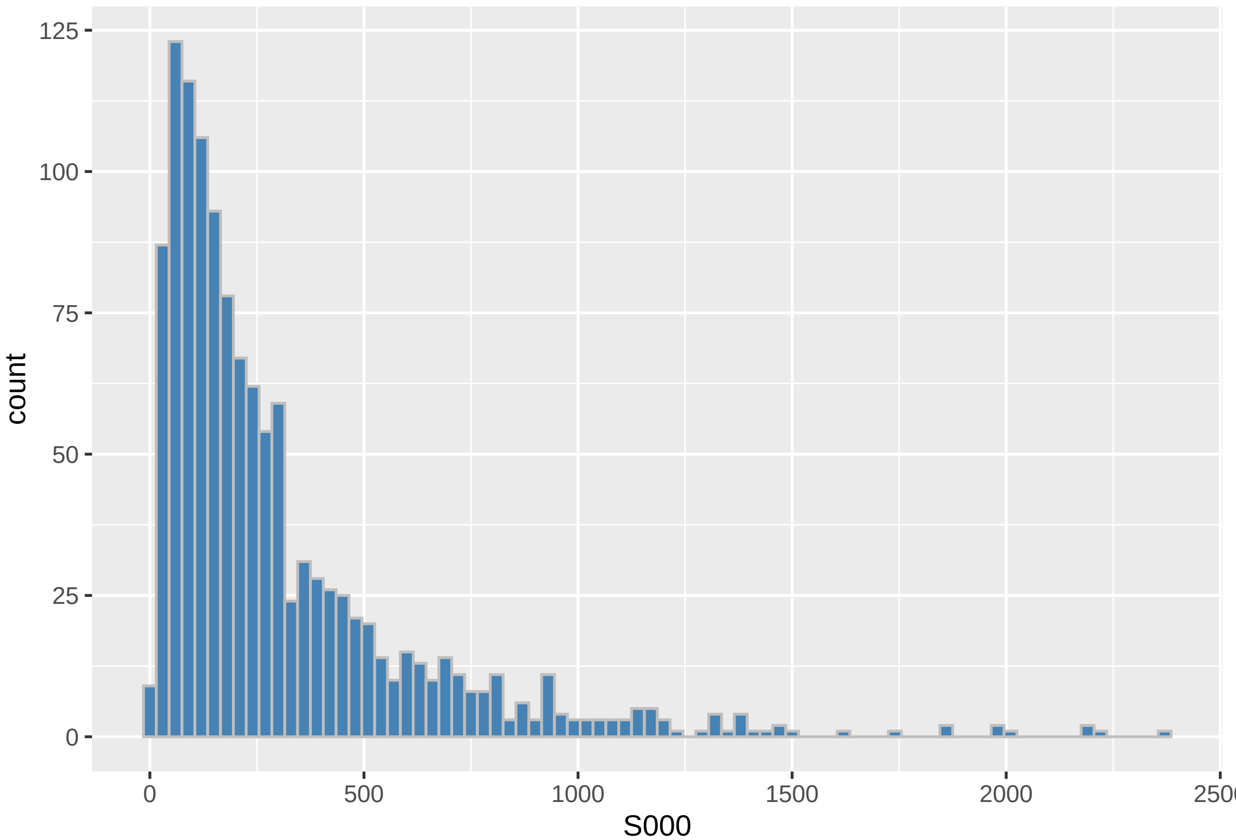


Distribution of commute counts

```
commute_counts <- subway_lines %>%
  dplyr::select(trip, S000) %>%
  dplyr::mutate(
    log_S000 = log(S000)
  )
```

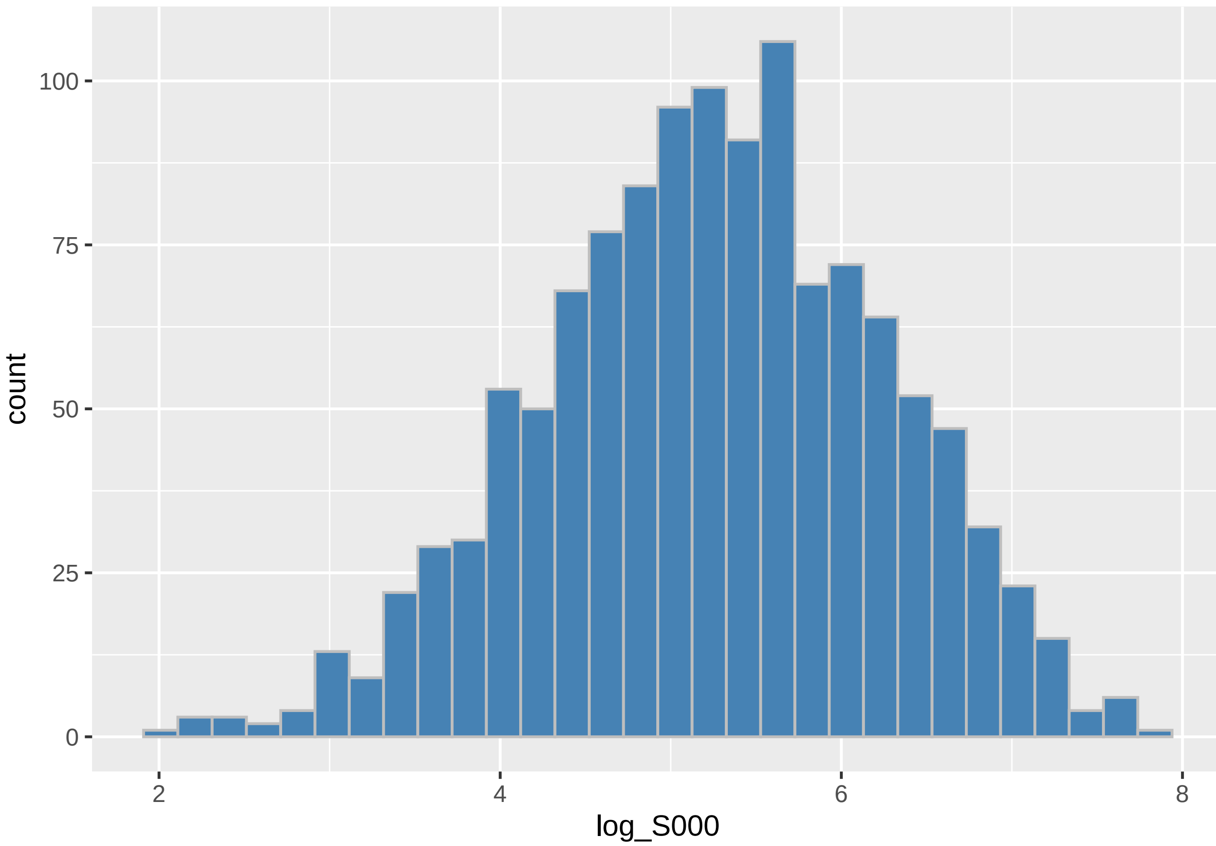
Original

```
ggplot(commute_counts) +
  geom_histogram(aes(x = S000), fill = "steelblue", color = "grey", binwidth = 30)
```



Natural log

```
ggplot(data = commute_counts) +
  geom_histogram(aes(x = log_S000), bins = 30, fill = "steelblue", color = "grey")
```

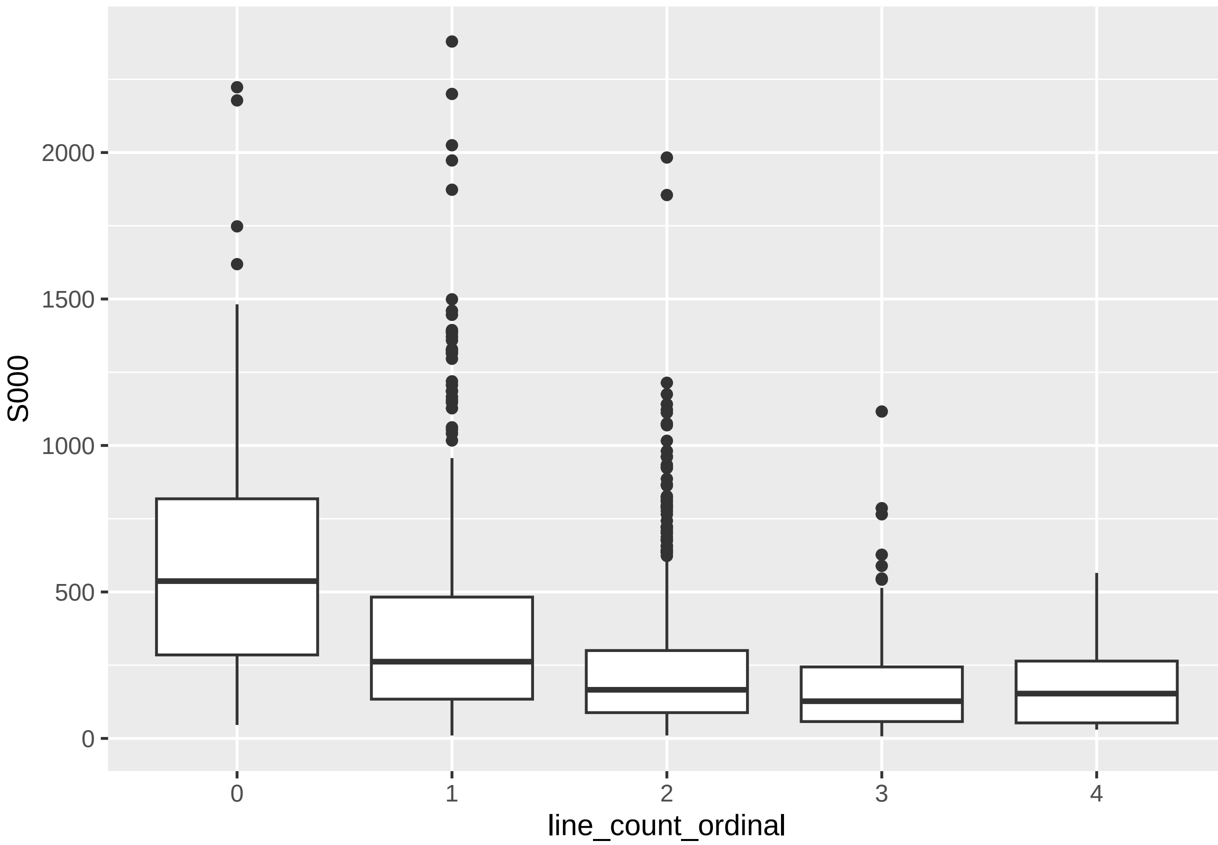


Number of subway lines and commute count

```
subway_lines <- subway_lines %>%
  dplyr::mutate(
    line_count_ordinal = as.character(line_count),
    log_S000 = log(S000)
  )
```

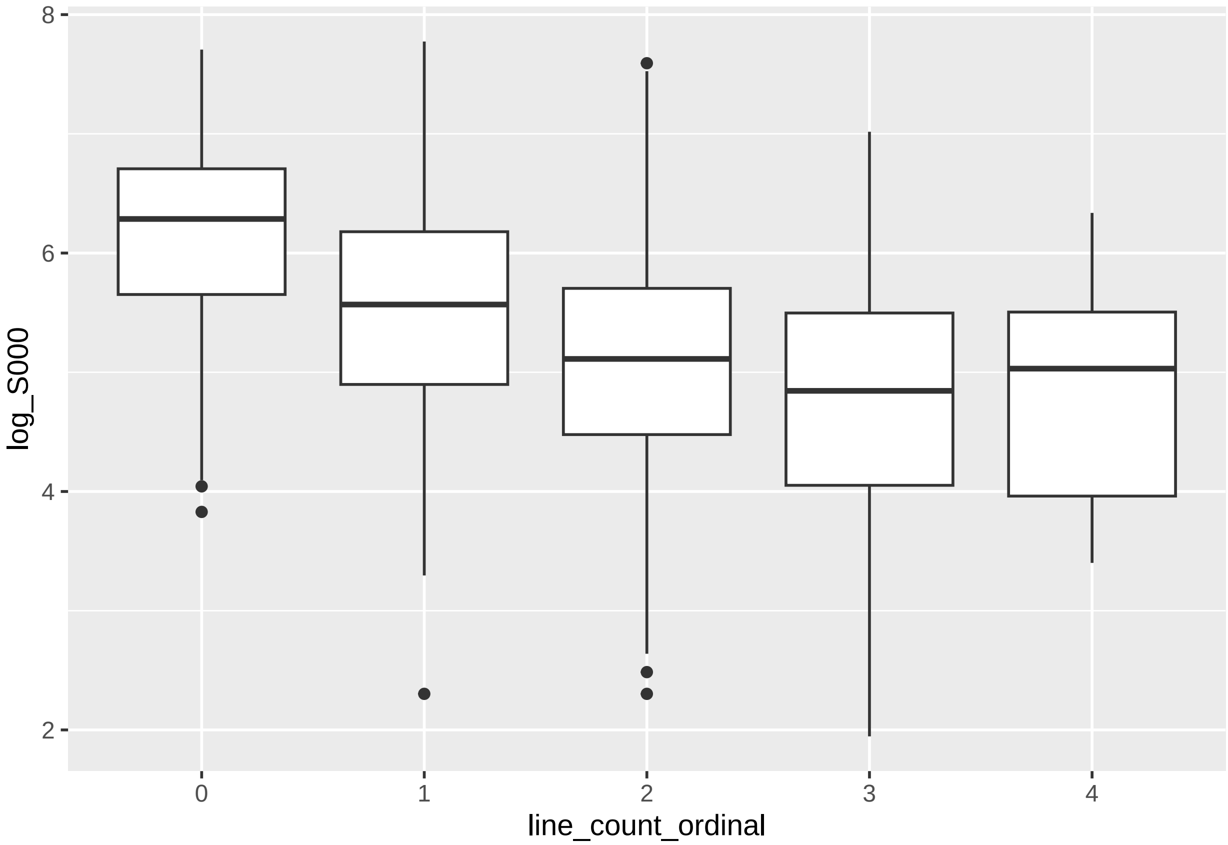
Original

```
ggplot(data = subway_lines, aes(x = line_count_ordinal, y = S000)) +
  geom_boxplot()
```



Transformed

```
ggplot(data = subway_lines, aes(x  = line_count_ordinal, y = log_S000)) +
  geom_boxplot()
```

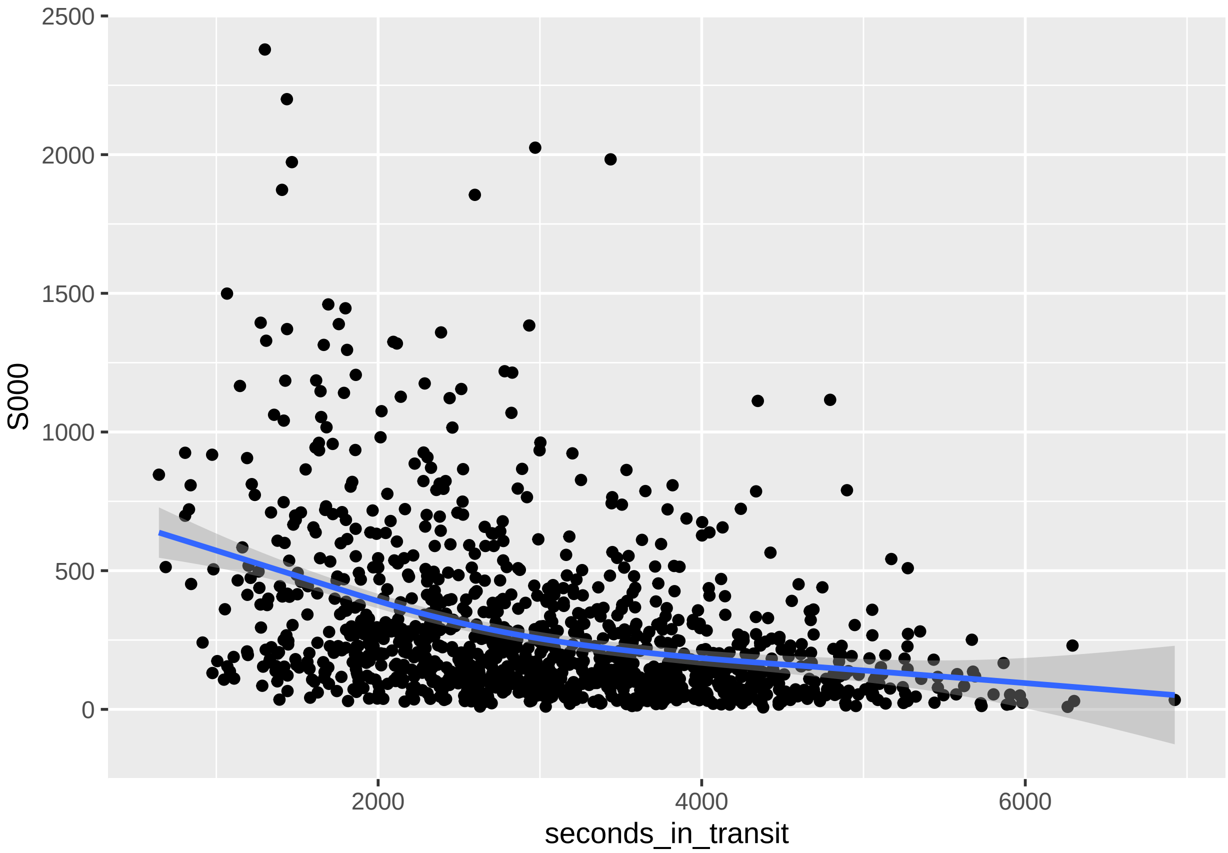


Subway Transit time and commute count

```
subway_times <- subway_times %>%
  dplyr::mutate(
    log_S000 = log(S000),
    i_seconds_in_transit = 1 / seconds_in_transit^(1/8)
  )
subway_times_connected <- subway_times %>%
  dplyr::filter(
    subway_lines$line_count > 0
  )
```

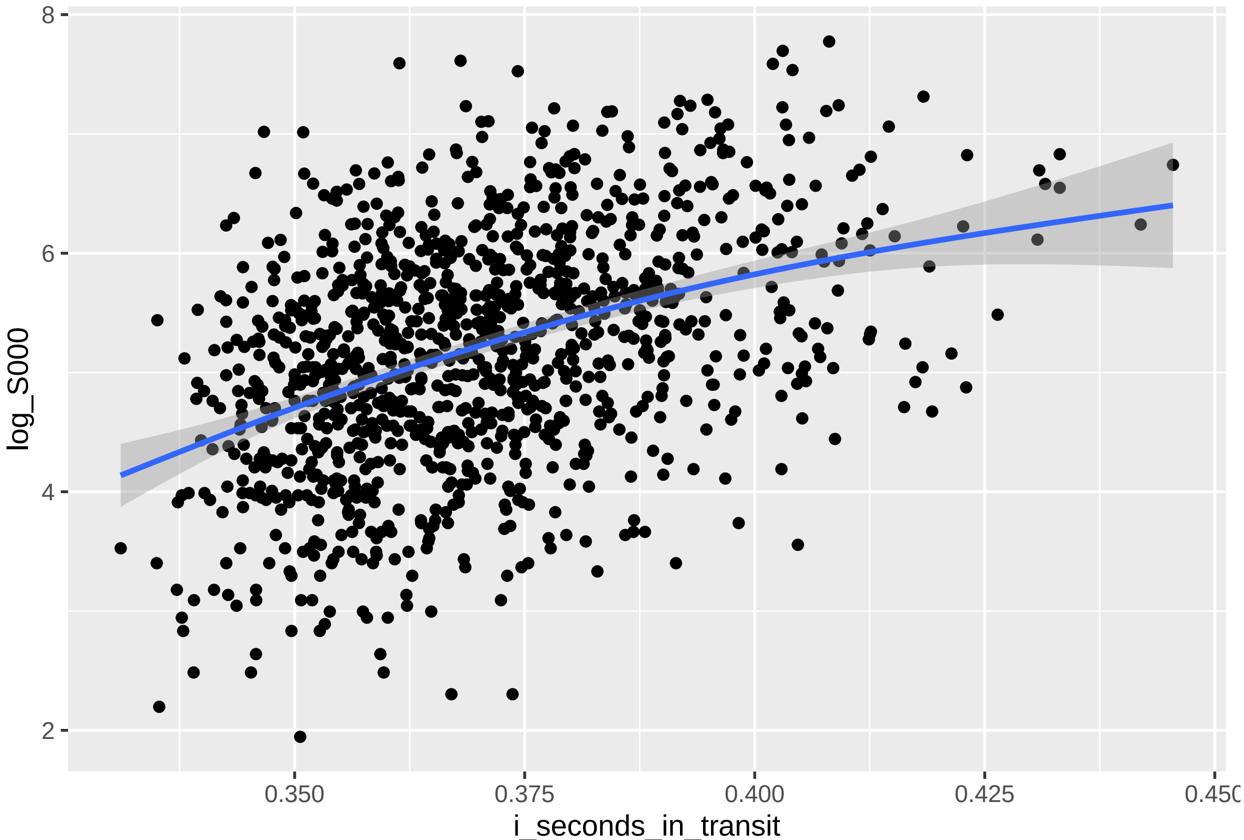
Original

```
ggplot(data = subway_times_connected, aes(x = seconds_in_transit, y = S000)) +
  geom_point() +
  stat_smooth()
```



Transformed

```
ggplot(data = subway_times_connected, aes(x = i_seconds_in_transit, y = log_S000)) +
  geom_point() +
  stat_smooth()
```

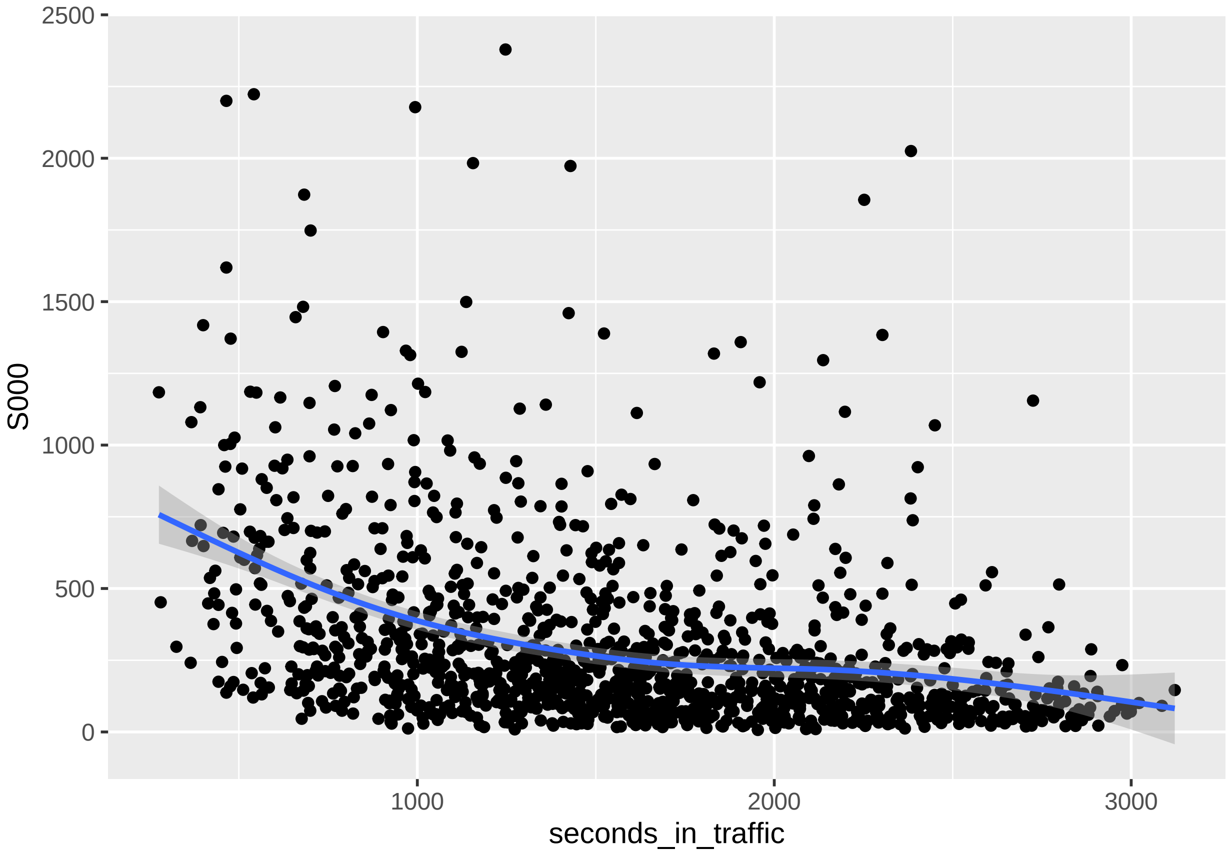


Driving in traffic time and commute count

```
driving_times <- driving_times %>%
  dplyr::mutate(
    log_S000 = log(S000),
    i_seconds_in_traffic = 1 / seconds_in_traffic^(1/8)
  )
```

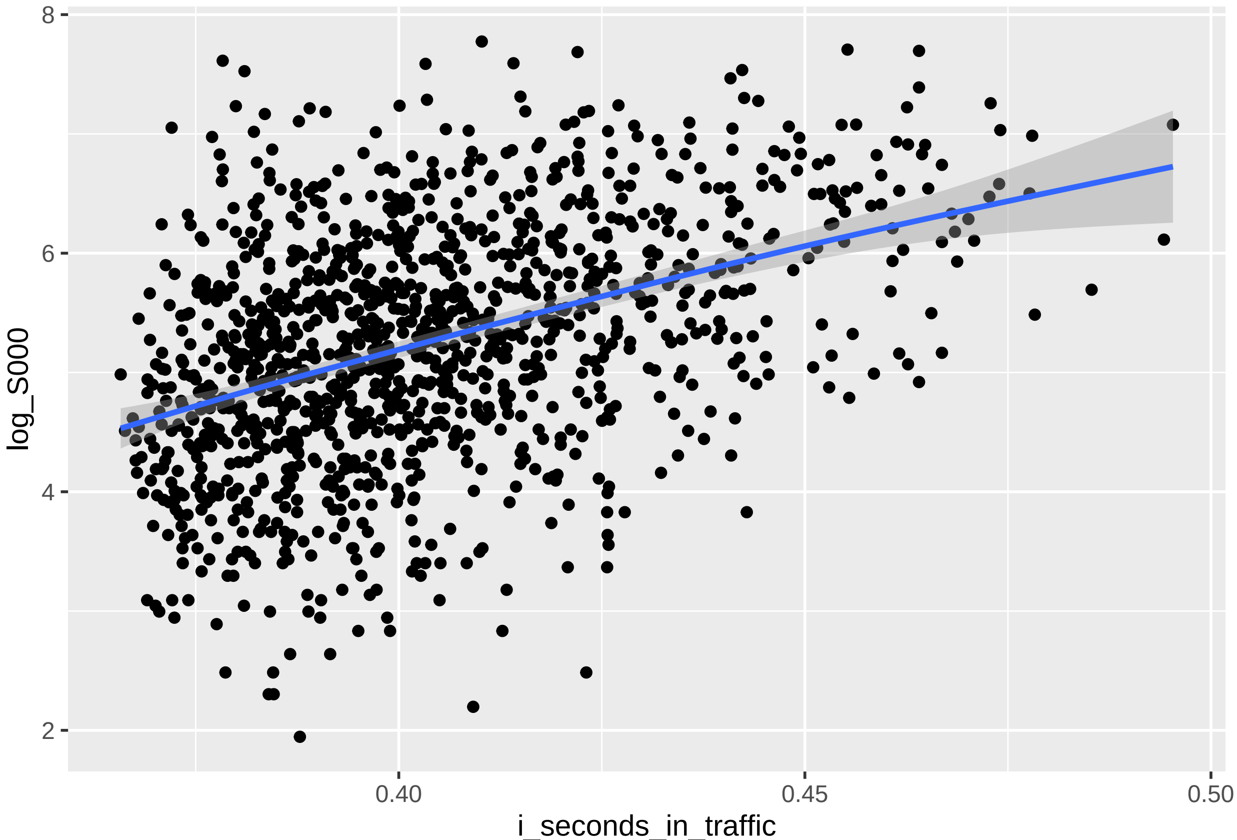
Original

```
ggplot(data = driving_times, aes(x = seconds_in_traffic, y = S000)) +
  geom_point() +
  stat_smooth()
```



Transformed

```
ggplot(data = driving_times, aes(x = i_seconds_in_traffic, y = log_S000)) +
  geom_point() +
  stat_smooth()
```

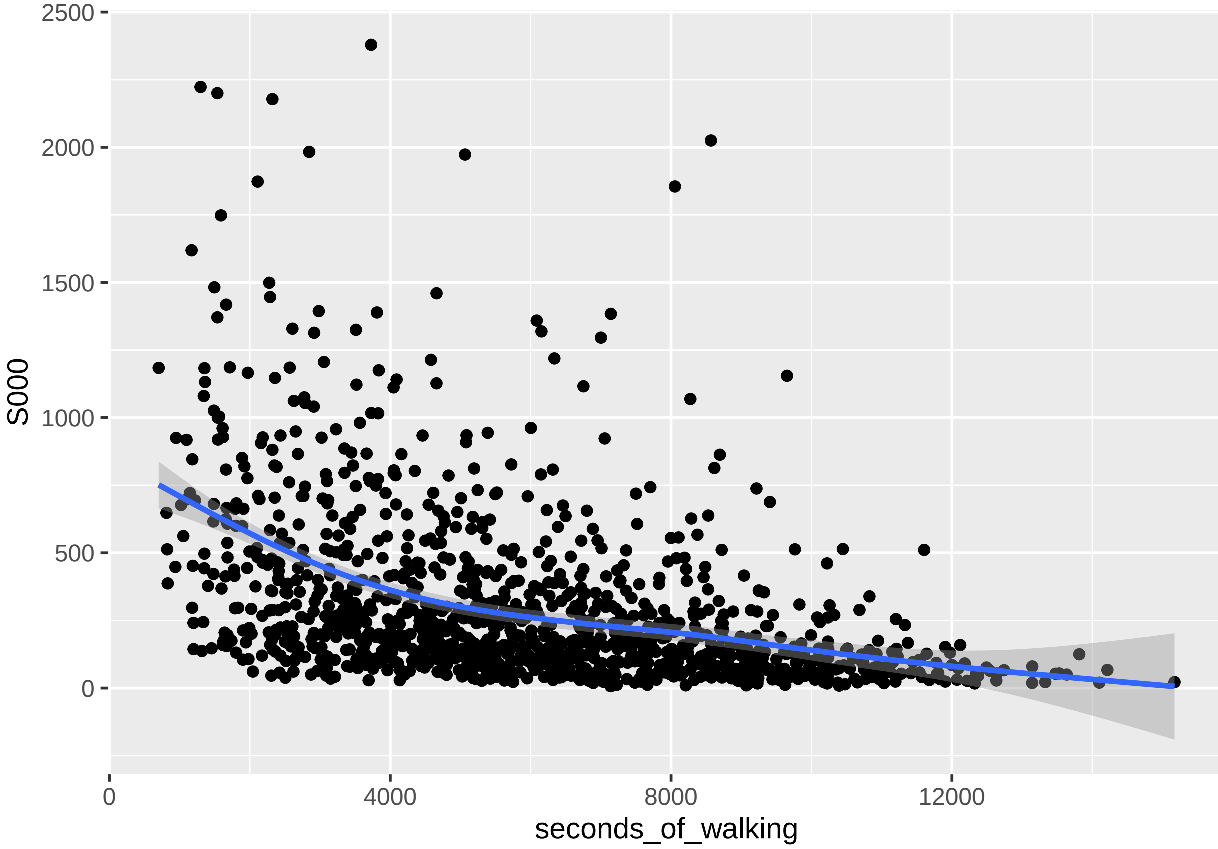


Walking time and commute count

```
walking_times <- walking_times %>%
  dplyr::mutate(
    log_S000 = log(S000),
    i_seconds_of_walking = 1 / seconds_of_walking^(1/8)
  )
```

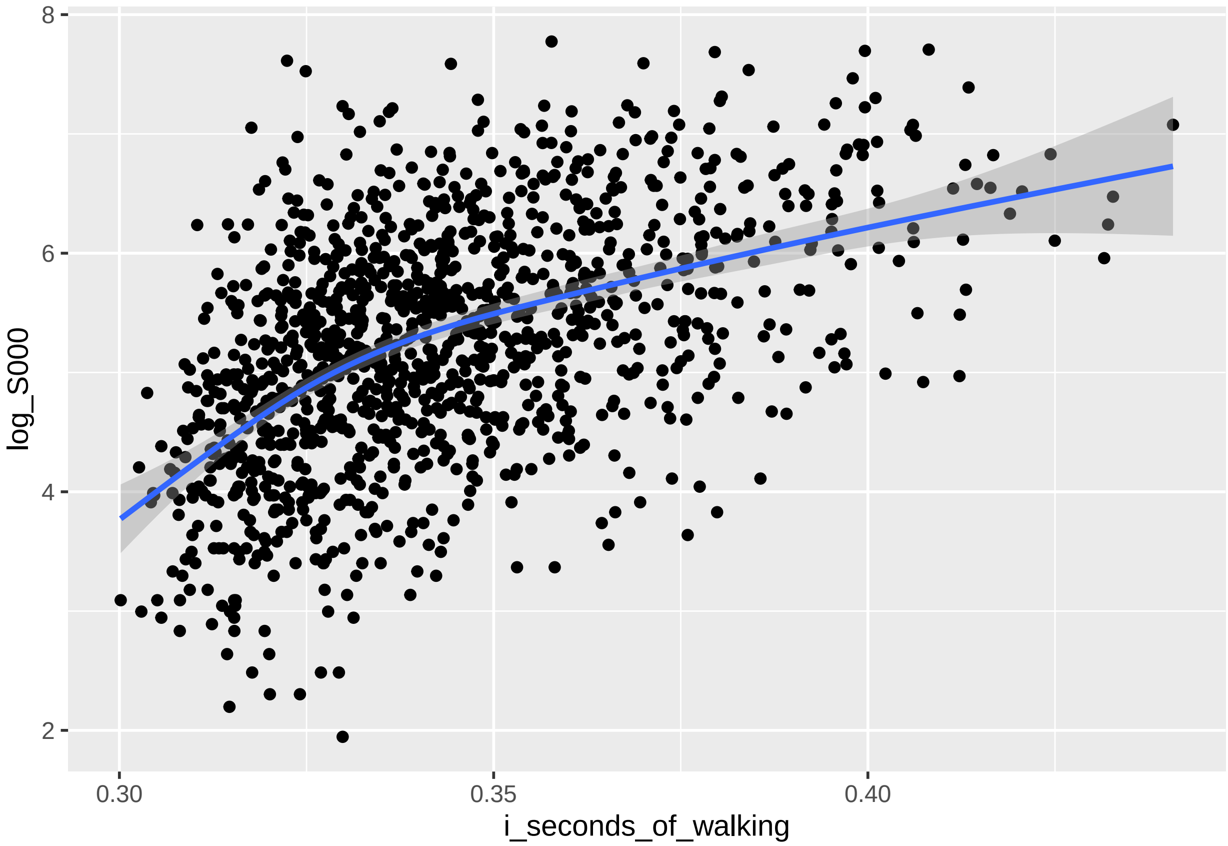
Original

```
ggplot(data = walking_times, aes(x = seconds_of_walking, y = S000)) +
  geom_point() +
  stat_smooth()
```



Transformed

```
ggplot(data = walking_times, aes(x = i_seconds_of_walking, y = log_S000)) +
  geom_point() +
  stat_smooth()
```



Regression of Subway, Driving, and walking

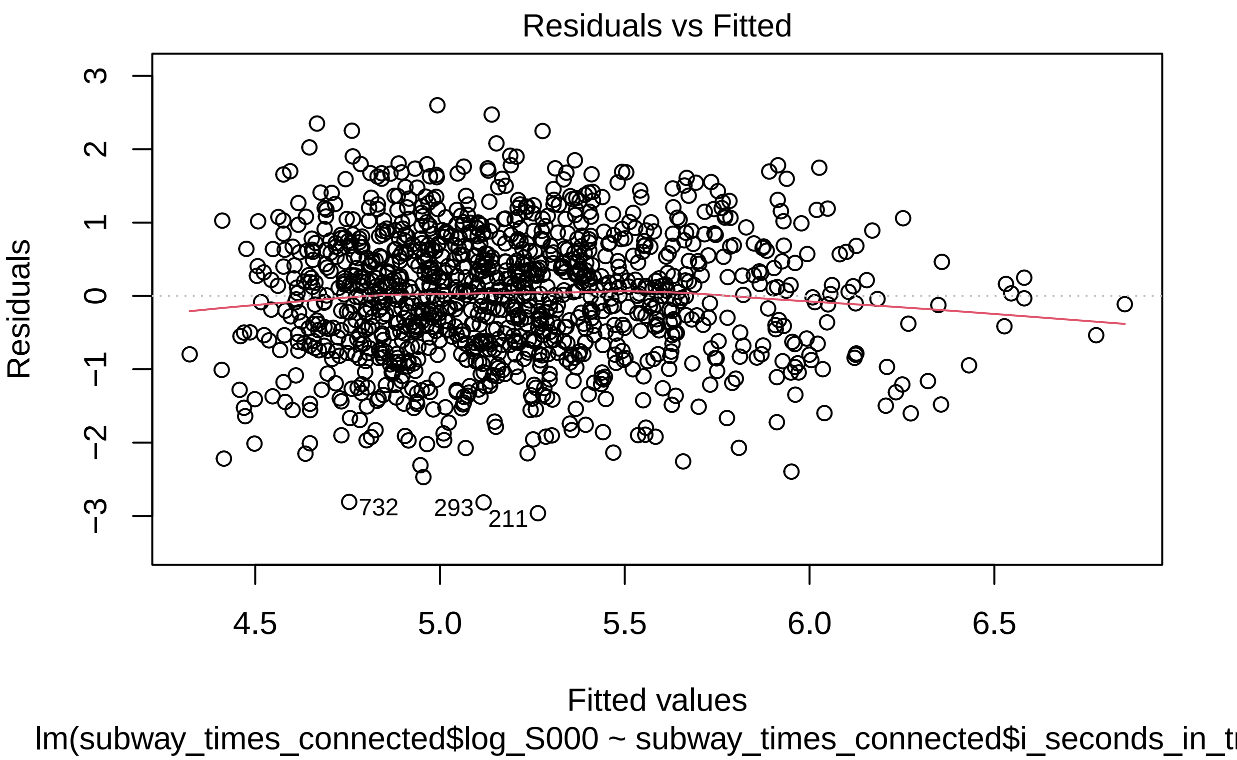
Subway model

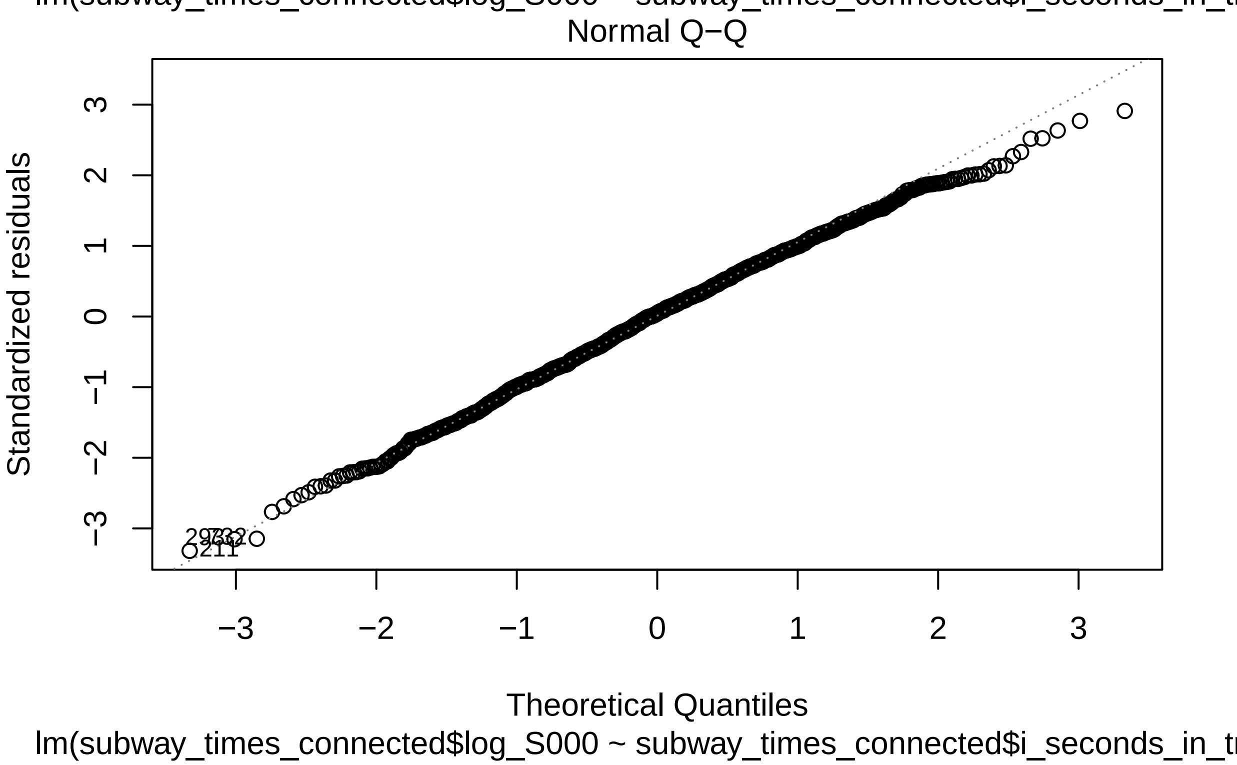
```
subway_connected_model <- lm(subway_times_connected$log_S000 ~ subway_times_connected$i_seconds_in_trans
summary(subway_connected_model)
```

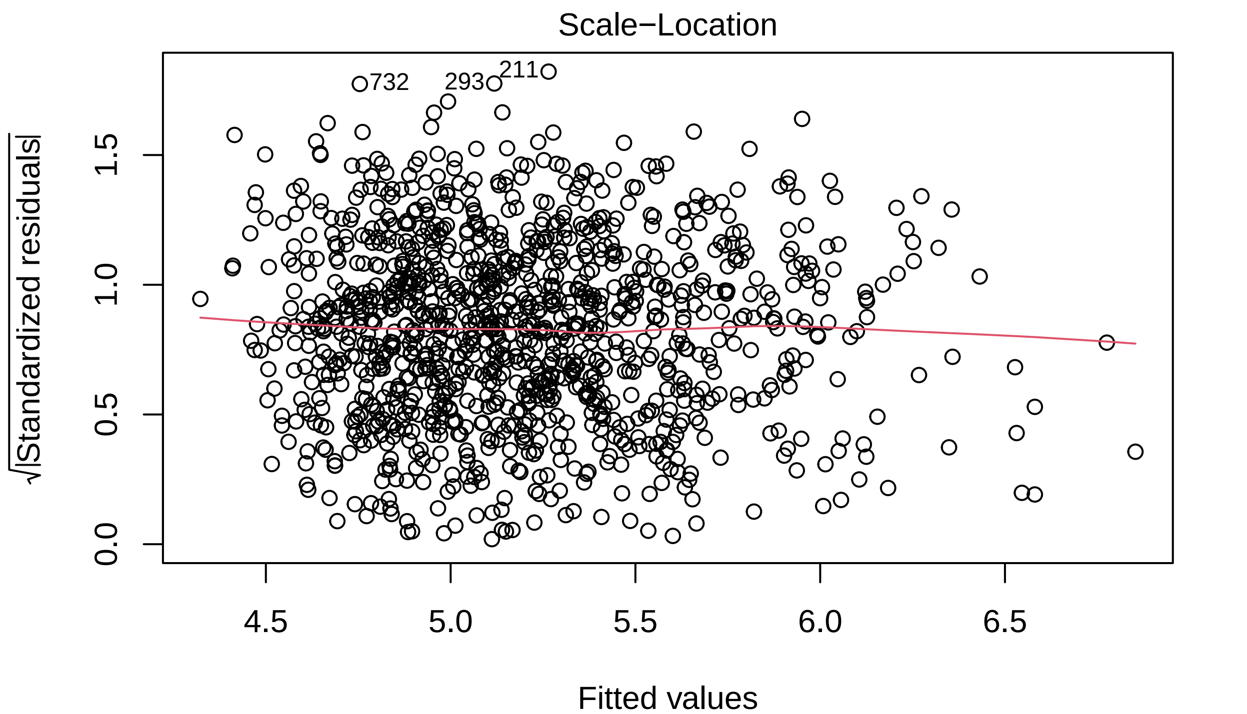
```
##
## Call:
## lm(formula = subway_times_connected$log_S000 ~ subway_times_connected$i_seconds_in_transit)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.96244 -0.61708  0.03841  0.63741  2.59935
##
## Coefficients:
##                                             Estimate Std. Error t value
## (Intercept)                                  -3.0033     0.5352  -5.612
## subway_times_connected$i_seconds_in_transit  22.1264     1.4434  15.330
##                                             Pr(>|t|)
## (Intercept)                                 2.51e-08 ***
## subway_times_connected$i_seconds_in_transit  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8933 on 1146 degrees of freedom
## Multiple R-squared:  0.1702, Adjusted R-squared:  0.1694
```

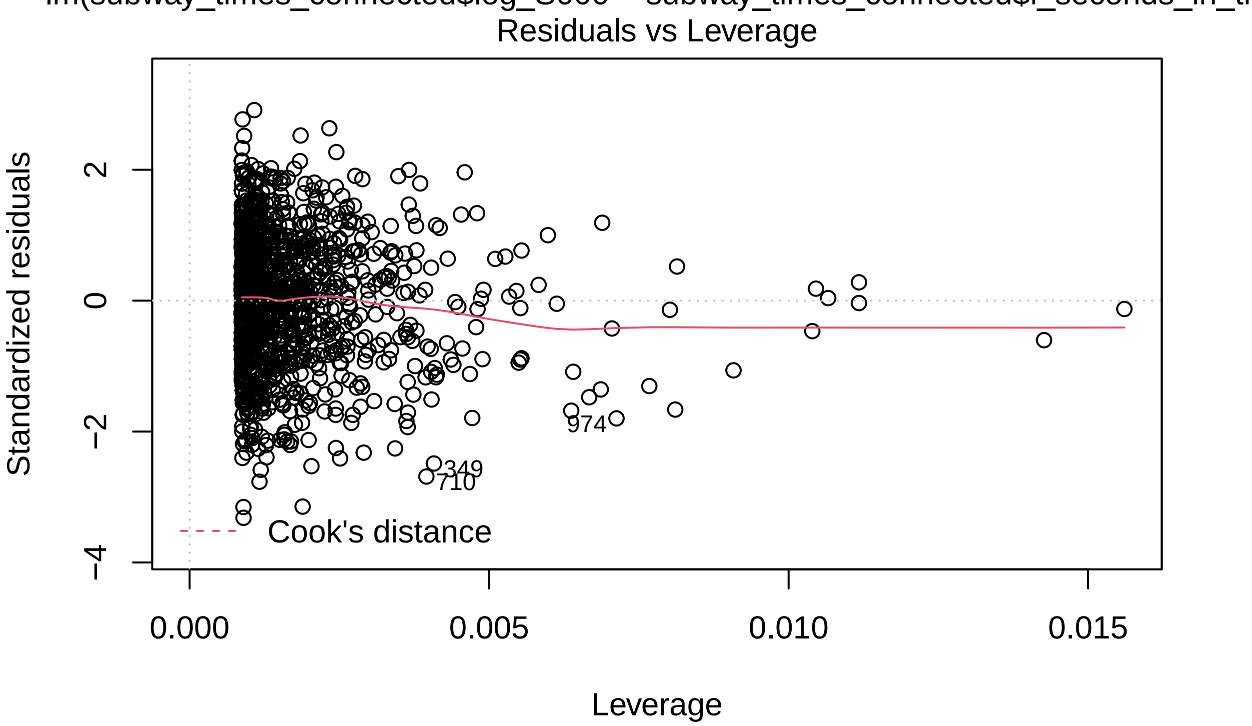
```
## F-statistic:   235 on 1 and 1146 DF,  p-value: < 2.2e-16
```

```
plot(subway_connected_model)
```









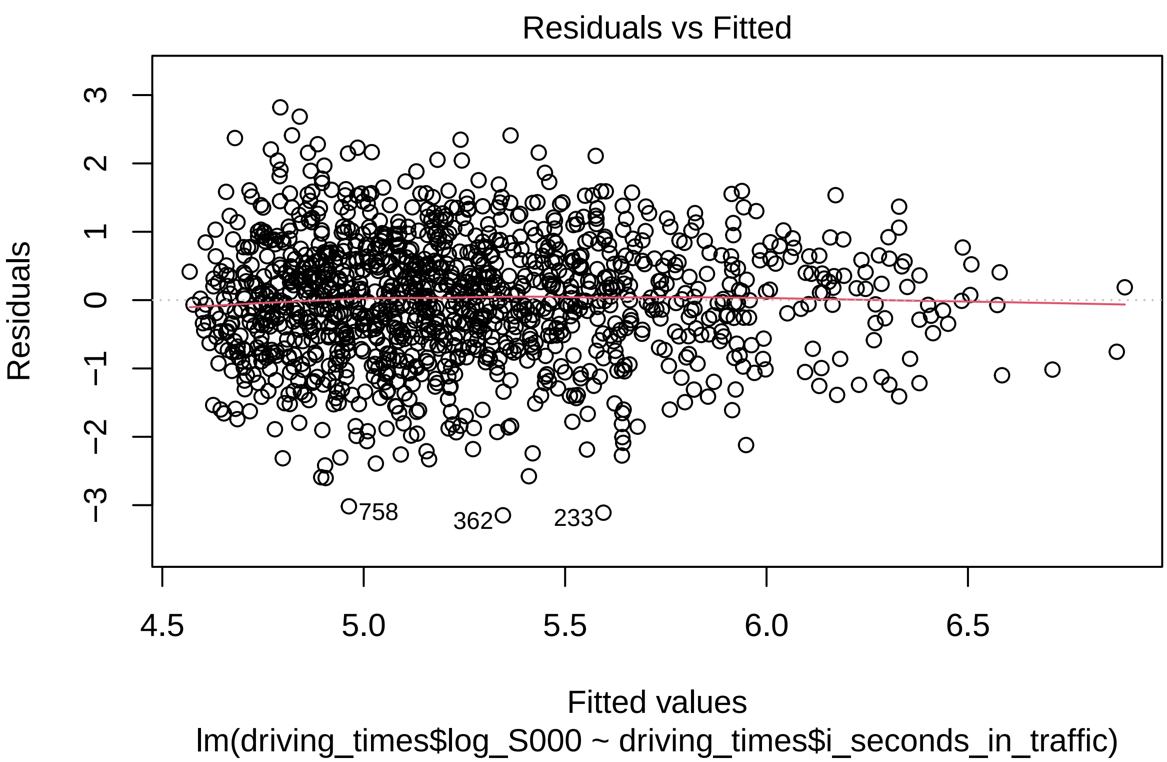
Driving

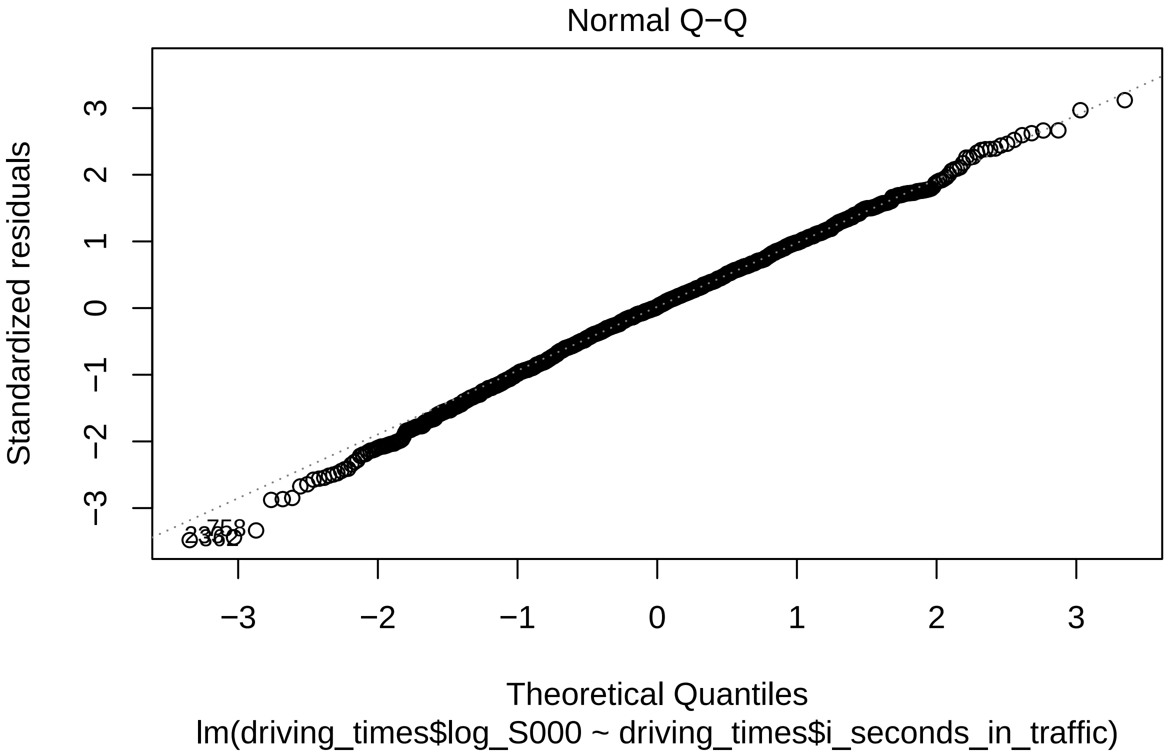
```
driving_model <- lm(driving_times$log_S000 ~ driving_times$i_seconds_in_traffic)
summary(driving_model)
```

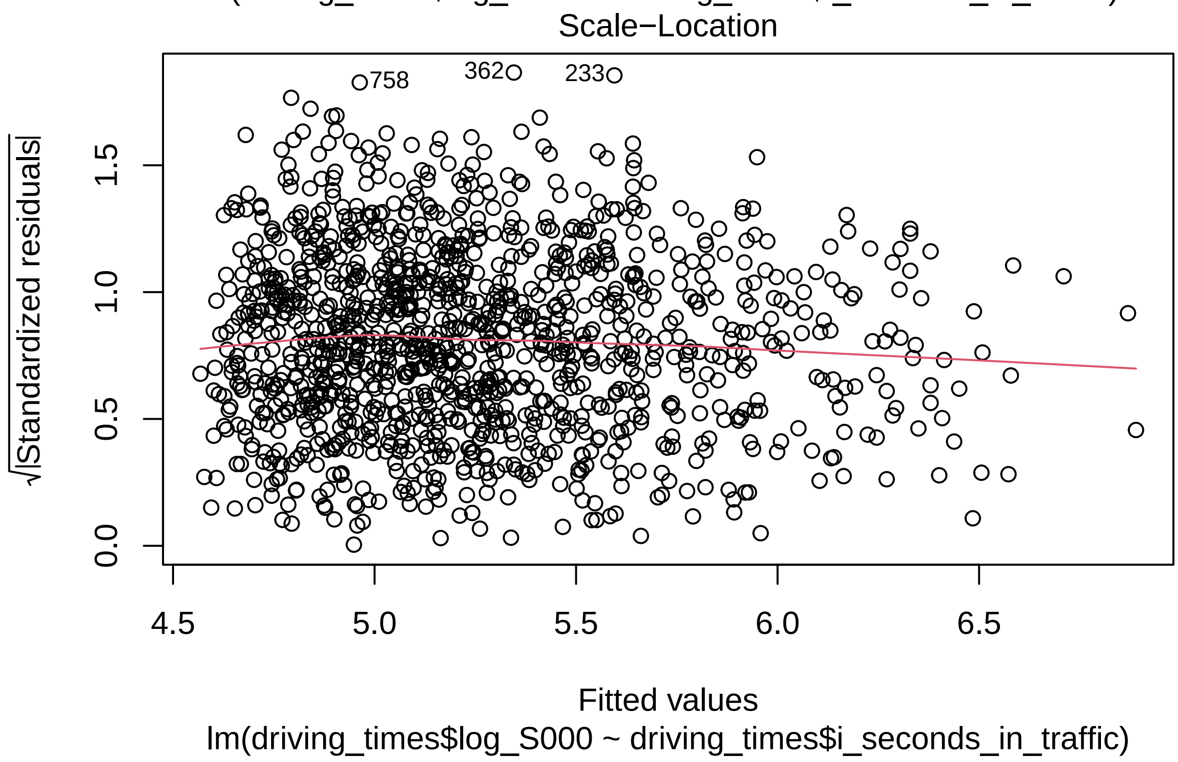
```
##
```

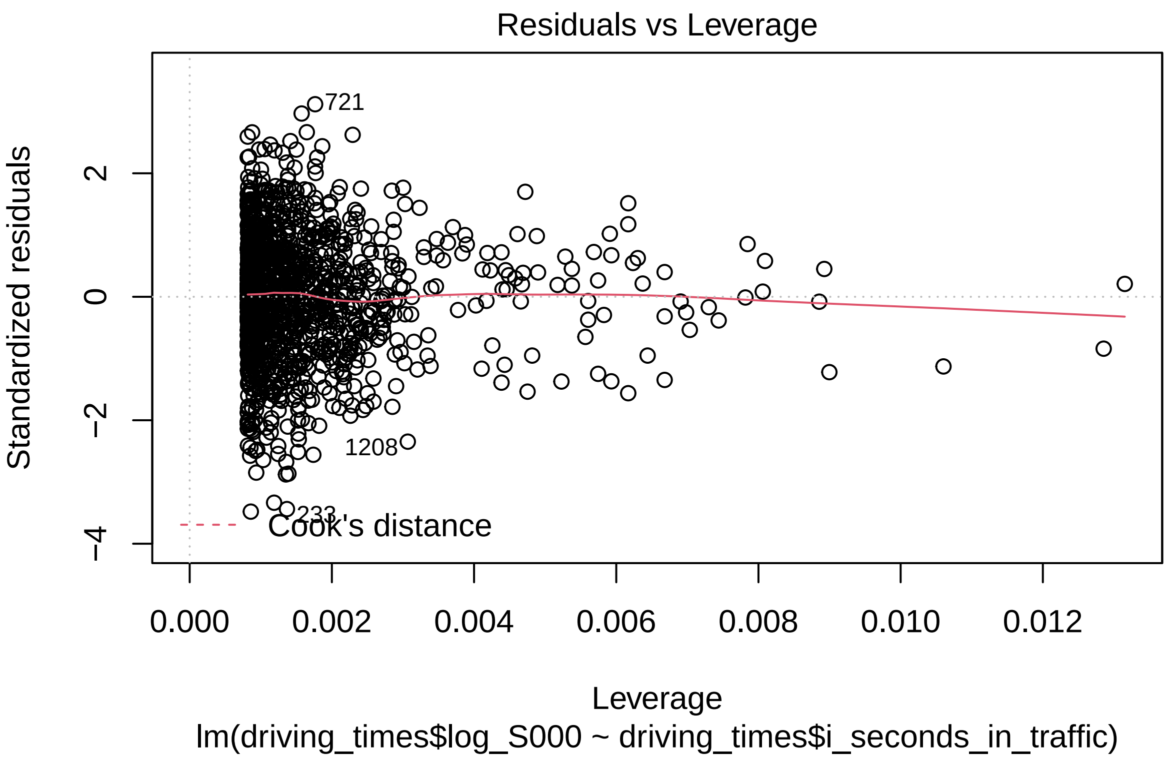
```
## Call:
## lm(formula = driving_times$log_S000 ~ driving_times$i_seconds_in_traffic)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -3.14840 -0.56631  0.01954  0.60172  2.82033
##
## Coefficients:
##                                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)                         -1.9856     0.4438  -4.474 8.39e-06 ***
## driving_times$i_seconds_in_traffic  17.9177     1.0974  16.327  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9051 on 1223 degrees of freedom
## Multiple R-squared:  0.179,  Adjusted R-squared:  0.1783
## F-statistic: 266.6 on 1 and 1223 DF,  p-value: < 2.2e-16
```

```
plot(driving_model)
```




Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(driving_times$log_S000 ~ driving_times$i_seconds_in_traffic)
758
233
362
−3
−2
−1
0
1
2
3


Scale−Location
√|Standardized residuals|
Fitted values
lm(driving_times$log_S000 ~ driving_times$i_seconds_in_traffic)
758
362
233
0.0
0.5
1.0
1.5
4.5
5.0
5.5
6.0
6.5

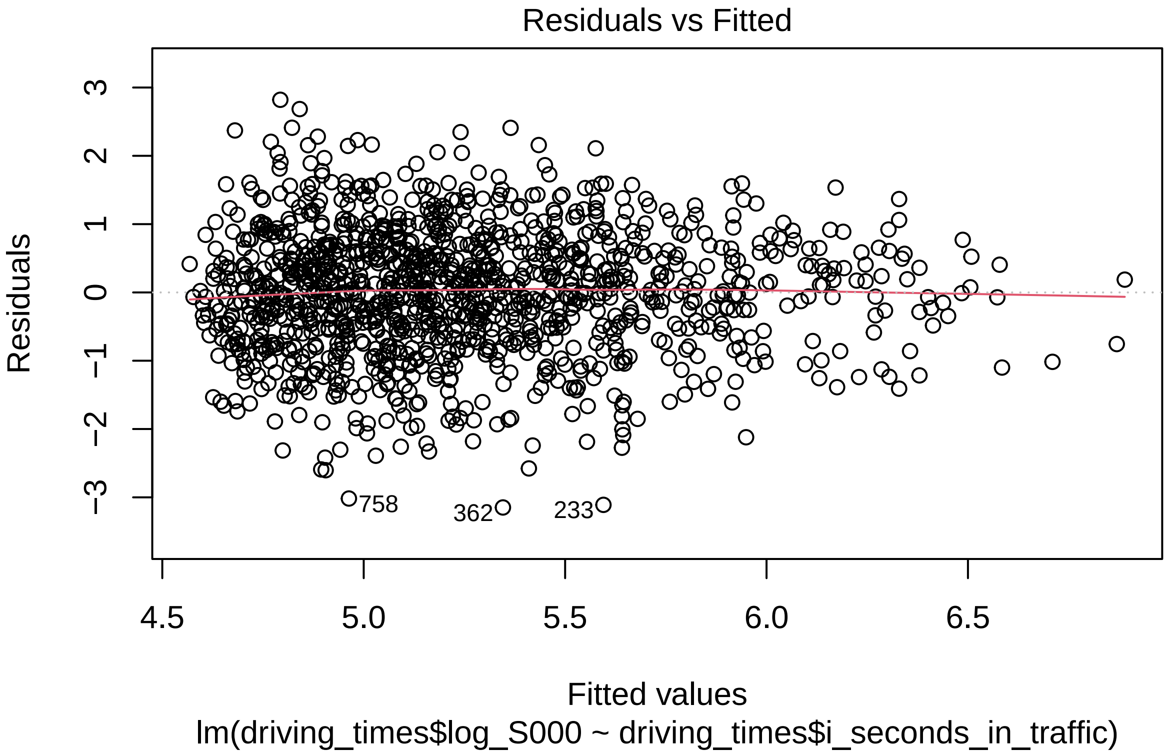


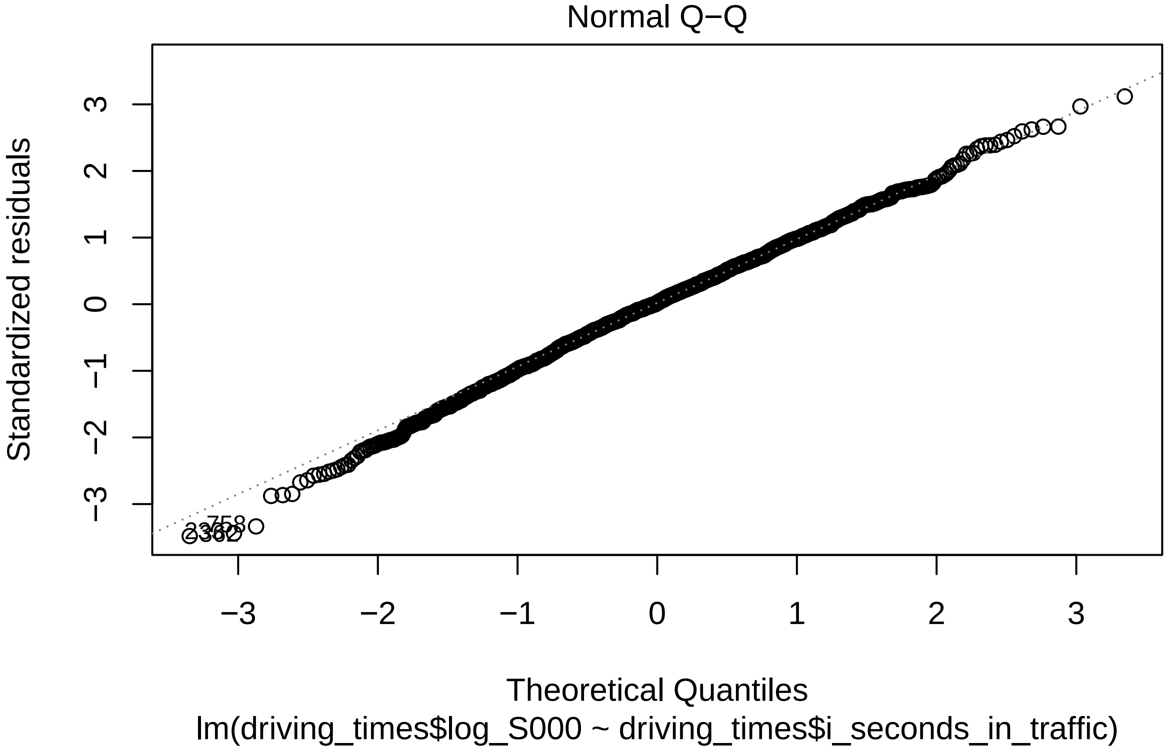
Walking

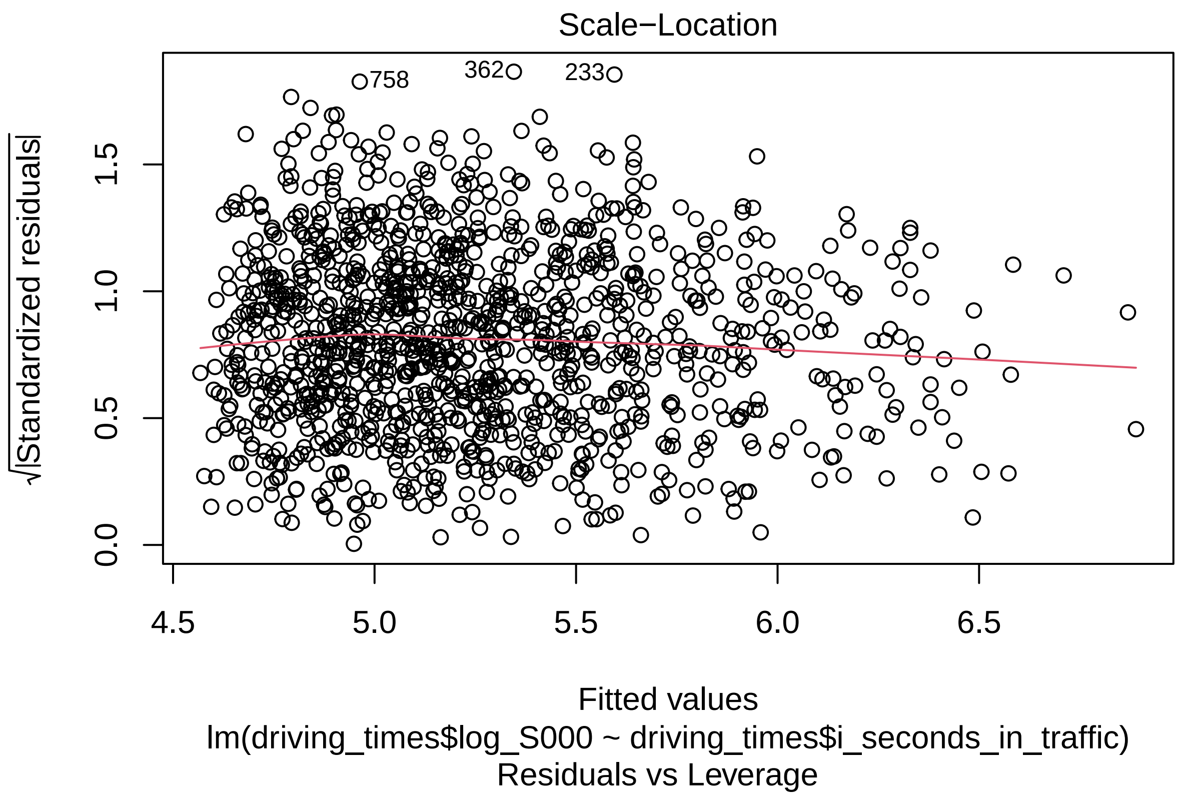
```
walking_model <- lm(walking_times$log_S000 ~ walking_times$i_seconds_of_walking)
summary(walking_model)
```

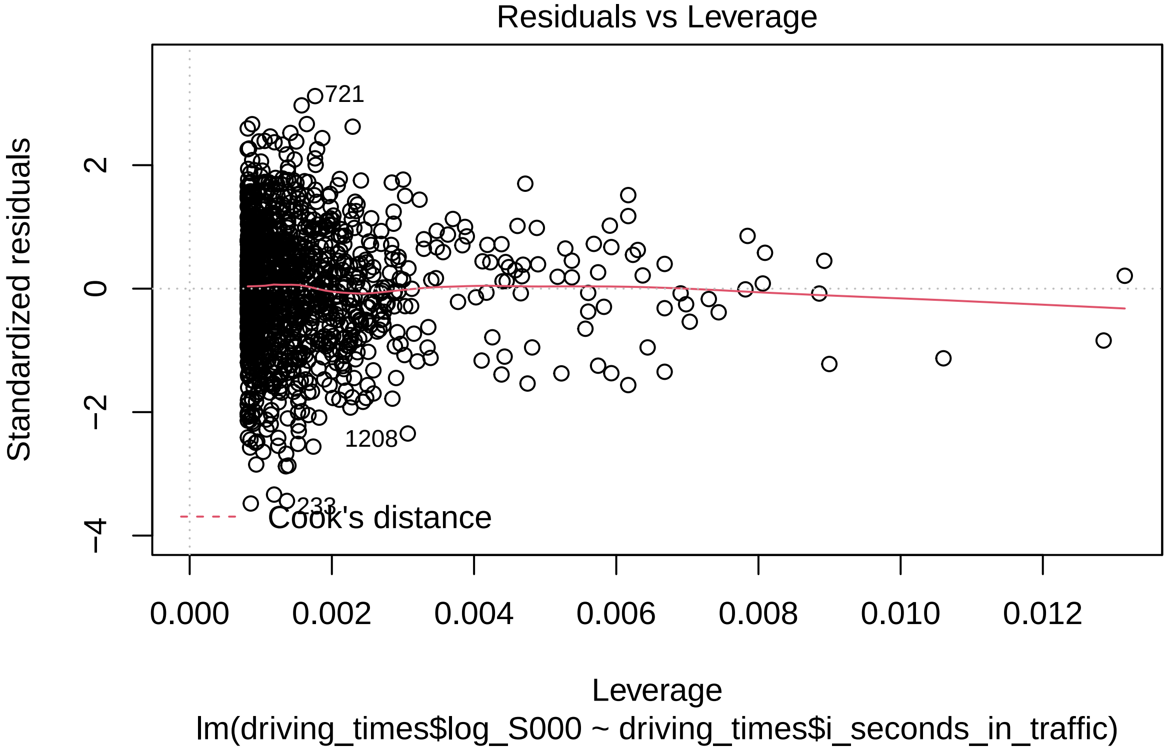
```
##
## Call:
## lm(formula = walking_times$log_S000 ~ walking_times$i_seconds_of_walking)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -3.01771 -0.56471  0.02807  0.58272  2.80456
##
## Coefficients:
##                                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)                         -1.9091     0.3532  -5.406 7.76e-08 ***
## walking_times$i_seconds_of_walking  20.8372     1.0257  20.315  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8637 on 1223 degrees of freedom
## Multiple R-squared:  0.2523, Adjusted R-squared:  0.2517
## F-statistic: 412.7 on 1 and 1223 DF,  p-value: < 2.2e-16
```

```
plot(driving_model)
```


Residuals vs Fitted
Residuals
Fitted values
lm(driving_times$log_S000 ~ driving_times$i_seconds_in_traffic)
758
362
233


Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(driving_times$log_S000 ~ driving_times$i_seconds_in_traffic)





Multiple linear regression for all three factors

Equations plotted for all factors

```
subway_connected_eq <- function(t) exp(-3.0033 + 22.1264 / t^(1/8))
driving_eq <- function(t) exp(-1.9856 + 17.9177 / t^(1 / 8))
```

```r
walking_eq <- function(t) exp(-1.9091 + 20.8372 / t^(1 / 8))

summary(subway_times_connected$seconds_in_transit)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     645    2274    2984    3061    3821    6924
```

```r
summary(driving_times$seconds_in_traffic)
```

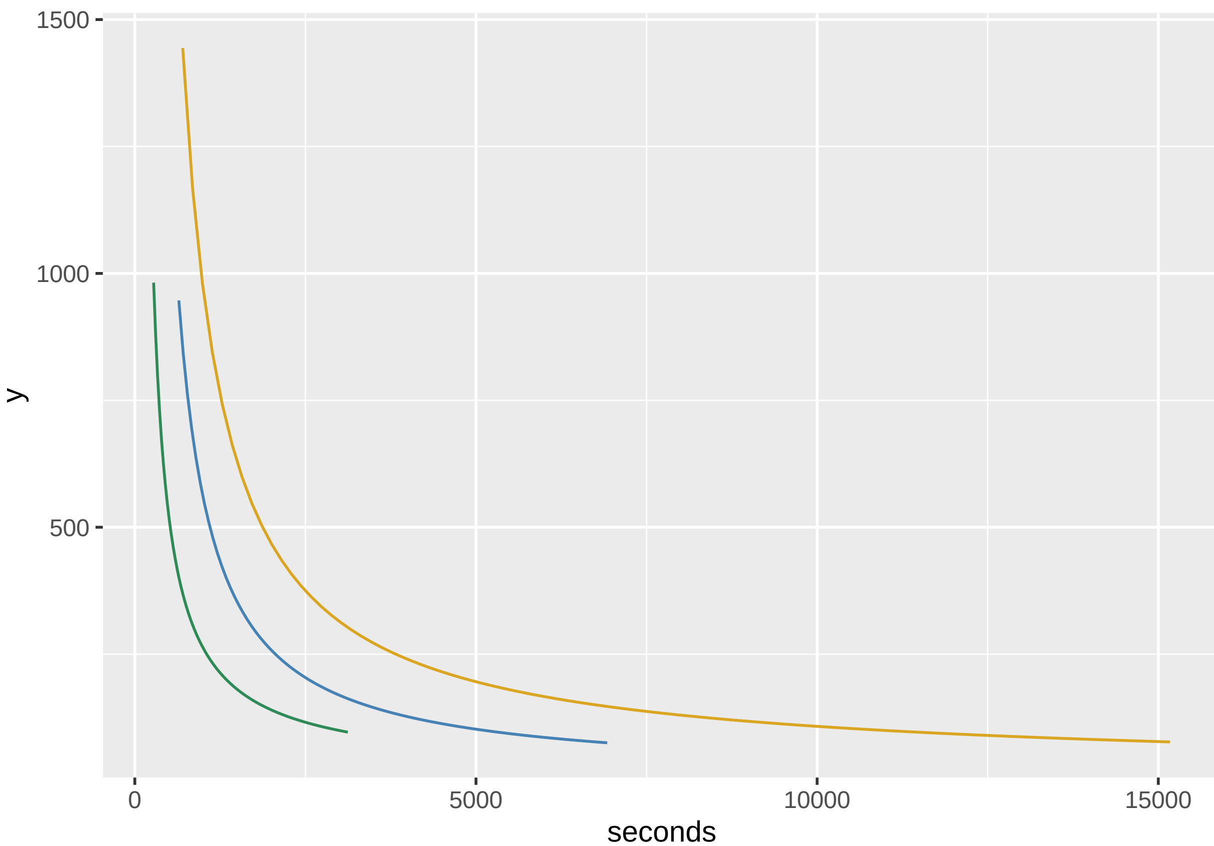
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     276    1076    1547    1576    2053    3122
```

```r
summary(walking_times$seconds_of_walking)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     702    3755    5725    5991    7974   15171
```

```r
ggplot(
  dplyr::data_frame(
    seconds = seq(from = 301, to = 15200, by = 14.9)
  ), aes(seconds)) +
  stat_function(fun = subway_connected_eq, color = "steelblue", xlim = c(645, 6924)) +
  stat_function(fun = driving_eq, color = "seagreen", xlim = c(276, 3122)) +
  stat_function(fun = walking_eq, color = "goldenrod", xlim = c(702, 15171))
```

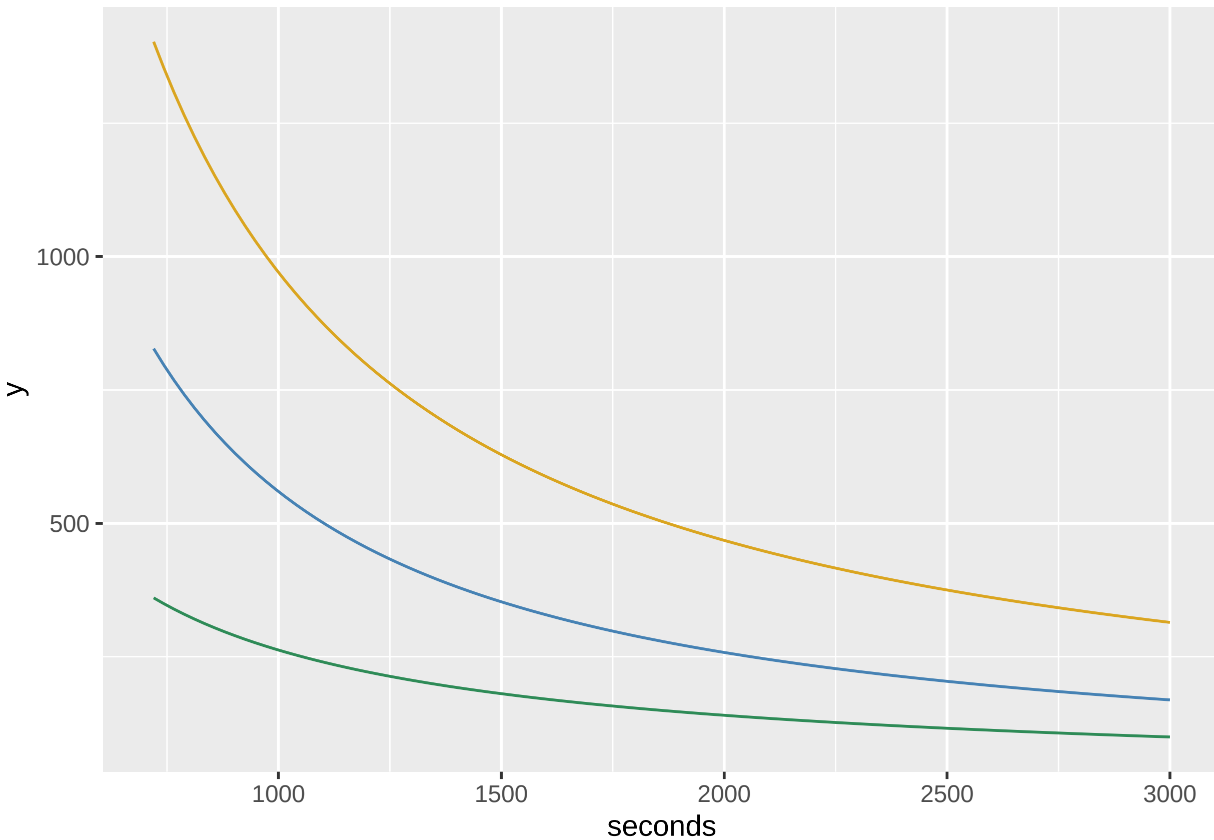
```
## Warning: `data_frame()` was deprecated in tibble 1.1.0.
## i Please use `tibble()` instead.
```



Along all axis

Cut to most pivotal times (10 to 50 minutes) Table of values at 10, 25, 50

```
ggplot(
  dplyr::data_frame(
    seconds = seq(from = 720, to = 3000, by = 2.28)
  ), aes(seconds)) +
  stat_function(fun = subway_connected_eq, color = "steelblue") +
  stat_function(fun = driving_eq, color = "seagreen") +
  stat_function(fun = walking_eq, color = "goldenrod")
```



Auto Correlation of Subway, Driving, and Walking

Global Moran's I

```
subway_times <- subway_times %>%
  dplyr::mutate(
    i_c_seconds_in_transit = ifelse(subway_lines$line_count > 0, i_seconds_in_transit, 0),
  )

subway_graph <- subway_times %>%
  dplyr::select(
    c(
      nta_one,
      nta_two,
      i_c_seconds_in_transit
    )
  ) %>%
  dplyr::rename(
```

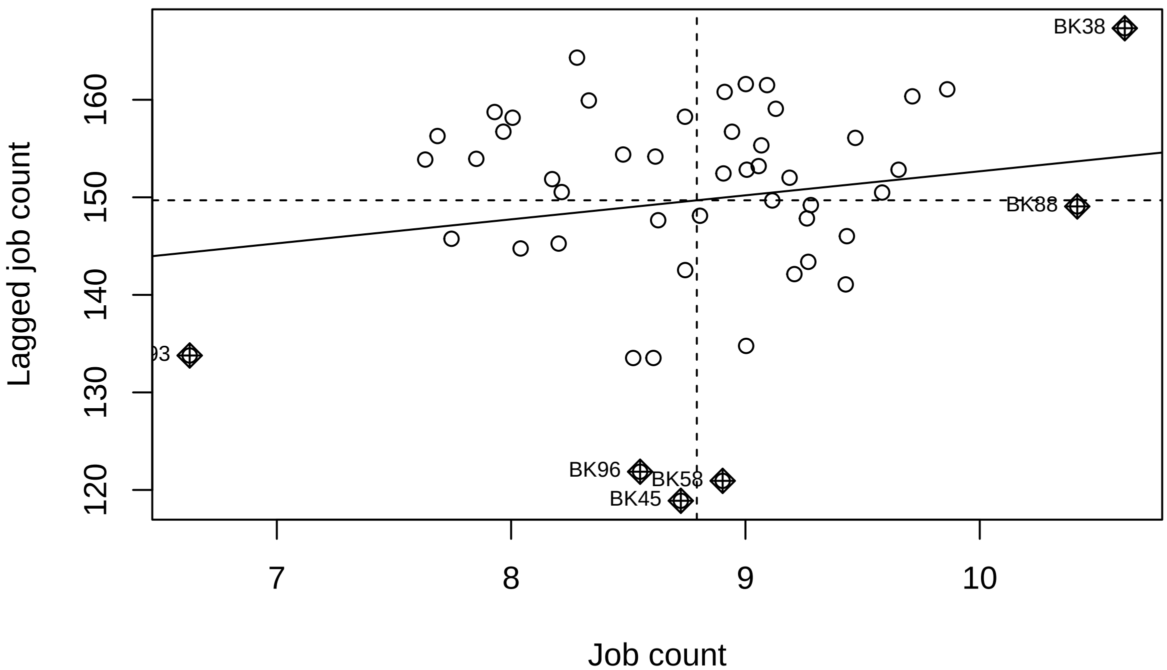
```r
    from = nta_one,
    to = nta_two,
    weight = i_c_seconds_in_transit,
  ) %>%
  igraph::graph.data.frame(
    directed = FALSE
  )

subway_weights <- subway_graph %>%
  igraph::as_adjacency_matrix(attr = "weight") %>%
  spdep::mat2listw()

driving_weights <- driving_times %>%
  dplyr::mutate(
    i_u_seconds_in_traffic = 1 / seconds_in_traffic
  ) %>%
  dplyr::select(
    c(
      nta_one,
      nta_two,
      i_seconds_in_traffic
    )
  ) %>%
  dplyr::rename(
    from = nta_one,
    to = nta_two,
    weight = i_seconds_in_traffic
  ) %>%
  igraph::graph.data.frame(
    directed = FALSE
  ) %>%
  igraph::as_adjacency_matrix(attr = "weight") %>%
  spdep::mat2listw()

walking_weights <- walking_times %>%
  dplyr::select(
    c(
      nta_one,
      nta_two,
      i_seconds_of_walking
    )
  ) %>%
  dplyr::rename(
    from = nta_one,
    to = nta_two,
    weight = i_seconds_of_walking
  ) %>%
  igraph::graph.data.frame(
    directed = FALSE
  ) %>%
  igraph::as_adjacency_matrix(attr = "weight") %>%
  spdep::mat2listw()
```

```r
subway_global_morans <- spdep::moran.test(
  job_counts$log_S000,
  subway_weights,
  zero.policy = TRUE,
)
print(subway_global_morans)
```

Subway

```
##
##  Moran I test under randomisation
##
## data:  job_counts$log_S000
## weights: subway_weights
##
## Moran I statistic standard deviate = -0.63359, p-value = 0.7368
## alternative hypothesis: greater
## sample estimates:
## Moran I statistic       Expectation          Variance
##     -2.477782e-02     -2.040816e-02      4.756382e-05
```

```r
spdep::moran.plot(
  job_counts$log_S000,
  subway_weights,
  zero.policy = TRUE,
  xlab = "Job count",
  ylab = "Lagged job count"
)
```



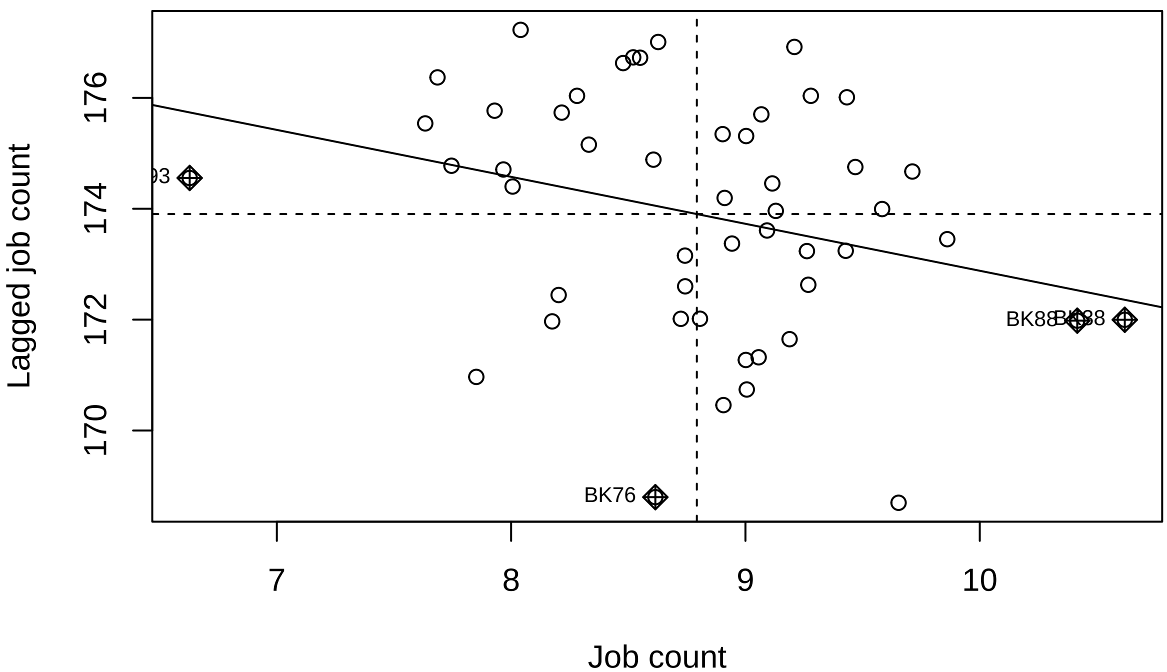
Driving

```r
driving_global_morans <- spdep::moran.test(
  job_counts$log_S000,
  driving_weights,
```

```
  zero.policy = TRUE,
)
print(driving_global_morans)
```

```
##
##  Moran I test under randomisation
##
## data:  job_counts$log_S000
## weights: driving_weights
##
## Moran I statistic standard deviate = 1.9231, p-value = 0.02724
## alternative hypothesis: greater
## sample estimates:
## Moran I statistic       Expectation          Variance
##     -1.736419e-02     -2.040816e-02      2.505457e-06
```

```
spdep::moran.plot(
  job_counts$log_S000,
  driving_weights,
  zero.policy = TRUE,
  xlab = "Job count",
  ylab = "Lagged job count"
)
```

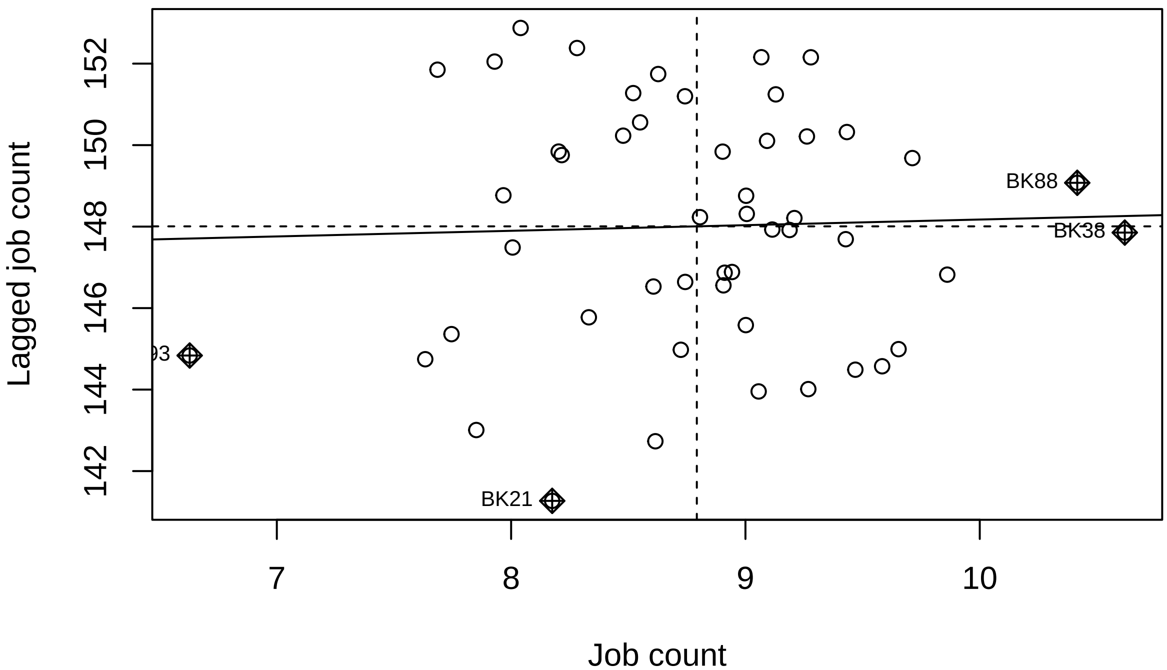



```
walking_global_morans <- spdep::moran.test(
  job_counts$log_S000,
  walking_weights,
  zero.policy = TRUE,
)
print(walking_global_morans)
```

Walking

```
##
##  Moran I test under randomisation
##
## data:  job_counts$log_S000
## weights: walking_weights
##
## Moran I statistic standard deviate = 1.7419, p-value = 0.04077
## alternative hypothesis: greater
## sample estimates:
## Moran I statistic       Expectation          Variance
##     -1.722520e-02     -2.040816e-02      3.339181e-06
```

```
spdep::moran.plot(
  job_counts$log_S000,
  walking_weights,
  zero.policy = TRUE,
  xlab = "Job count",
  ylab = "Lagged job count"
)
```



LISA

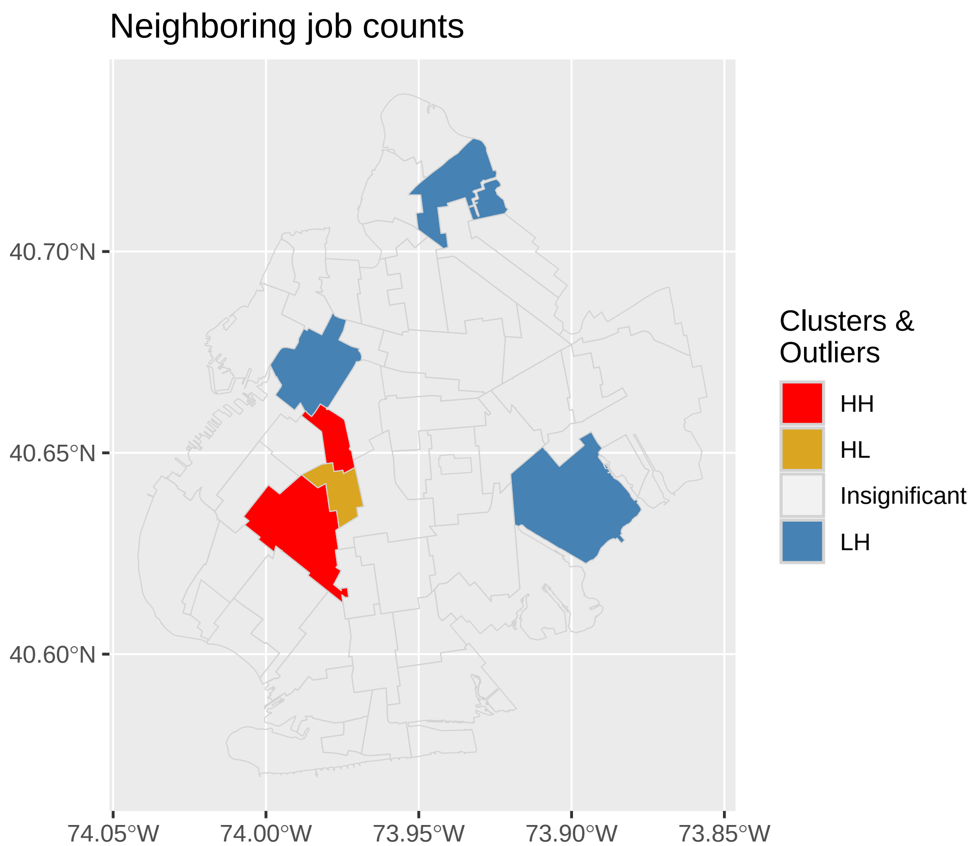
```
avg_jobs <- mean(job_counts$log_S000)

classify_co_types <- function(mode_lisa, l_job_counts, avg_job_count) {
  mode_lisa %>%
    tibble::as_tibble() %>%
    magrittr::set_colnames(
      c("Ii","E.Ii","Var.Ii","Z.Ii","Pr(z > 0)")
    ) %>%
    dplyr::mutate(
      co_type = dplyr::case_when(
        `Pr(z > 0)` <= 0.05 &
```

```r
          Ii >= 0 &
          l_job_counts >= avg_job_count ~ "HH",
        `Pr(z > 0)` <= 0.05 &
          Ii >= 0 &
          l_job_counts <  avg_job_count ~ "LL",
        `Pr(z > 0)` <= 0.05 &
          Ii < 0 &
          l_job_counts >= avg_job_count ~ "HL",
        `Pr(z > 0)` <= 0.05 &
          Ii < 0 &
          l_job_counts < avg_job_count ~ "LH"
      )
    )
}

subway_lisa <- spdep::localmoran(
  job_counts$log_S000,
  subway_weights,
  zero.policy = TRUE,
  na.action = na.omit
)

driving_lisa <- spdep::localmoran(
  job_counts$log_S000,
  driving_weights,
  zero.policy = TRUE,
  na.action = na.omit
)

walking_lisa <- spdep::localmoran(
  job_counts$log_S000,
  walking_weights,
  zero.policy = TRUE,
  na.action = na.omit
)

subway_classes <- classify_co_types(subway_lisa, job_counts$log_S000, avg_jobs)
driving_classes <- classify_co_types(driving_lisa, job_counts$log_S000, avg_jobs)
walking_classes <- classify_co_types(walking_lisa, job_counts$log_S000, avg_jobs)

subway_bk_nta_border <- bk_nta_border %>%
  dplyr::mutate(
    co_type = ifelse(is.na(subway_classes$co_type), "Insignificant", subway_classes$co_type)
  )

driving_bk_nta_border <- bk_nta_border %>%
  dplyr::mutate(
    co_type = ifelse(is.na(driving_classes$co_type), "Insignificant",  driving_classes$co_type)
  )

walking_bk_nta_border <- bk_nta_border %>%
  dplyr::mutate(
    co_type = ifelse(is.na(walking_classes$co_type), "Insignificant", walking_classes$co_type)
```

```
)
```

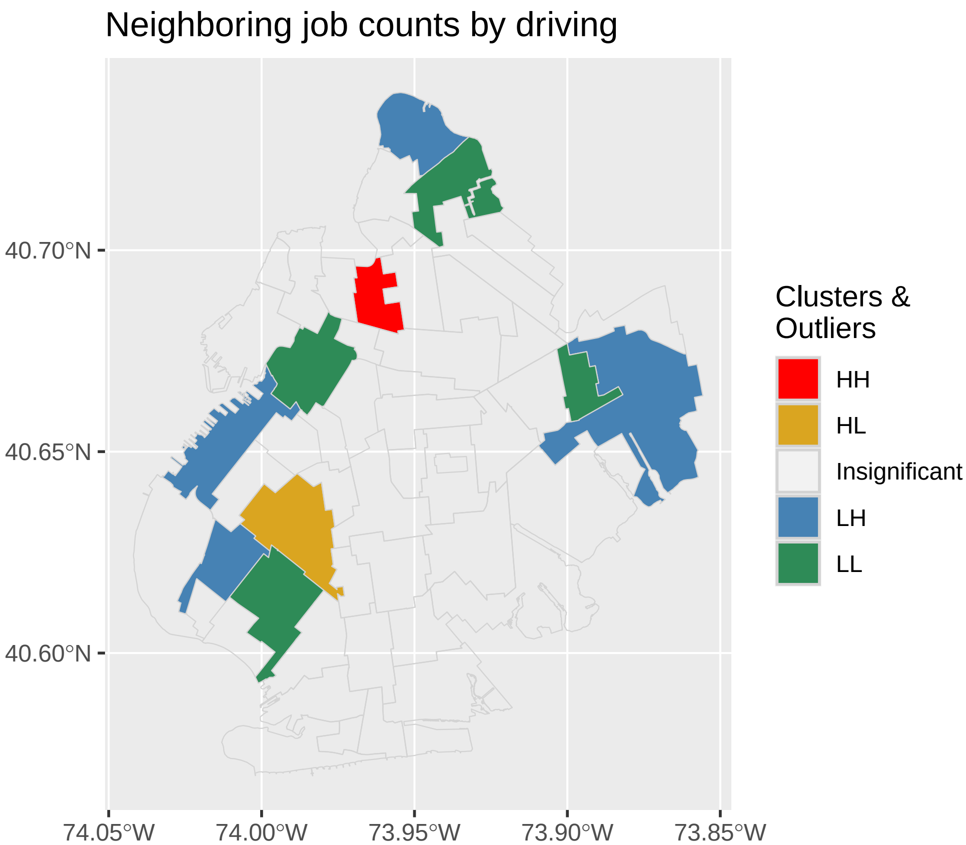
Subway Statistic Plot with subway lines

```
ggplot(subway_bk_nta_border) +
  geom_sf(aes(fill = co_type), col = 'lightgrey') +
  scale_fill_manual(
    values = c("red", "goldenrod", "NA", "steelblue"),
    name = "Clusters & \nOutliers"
  ) +
  labs(
    title = "Neighboring job counts"
  )
```



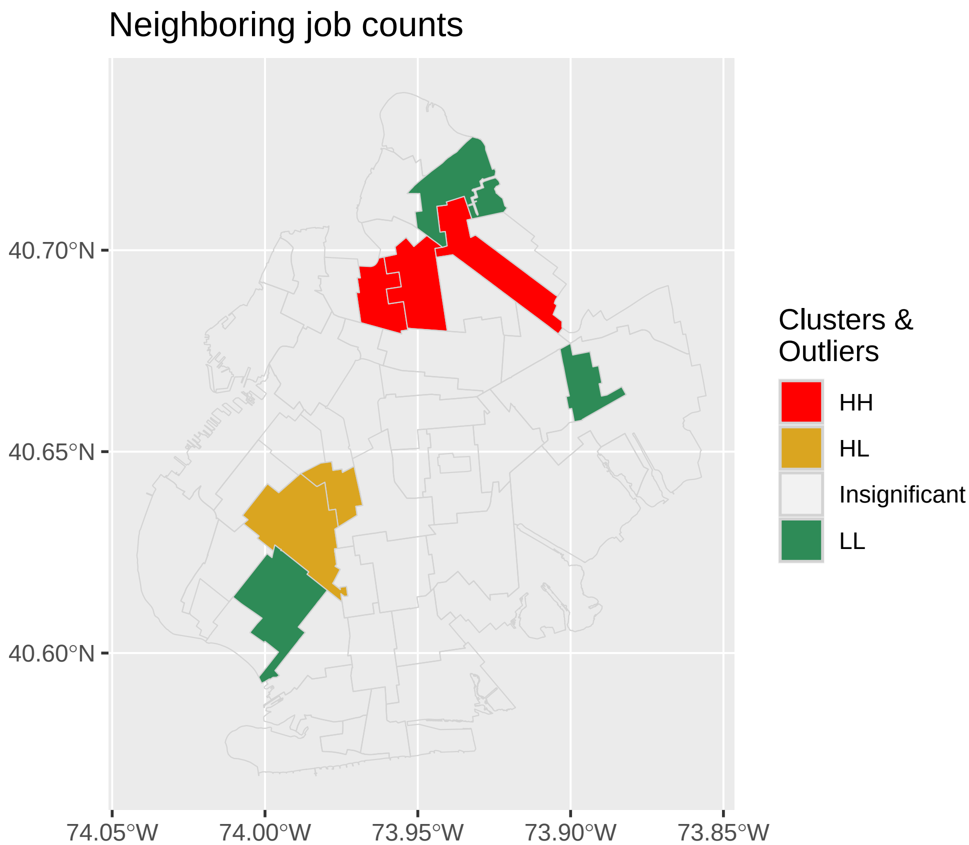
Driving

```
ggplot(driving_bk_nta_border) +
  geom_sf(aes(fill = co_type), col = 'lightgrey') +
  scale_fill_manual(
    values = c("red", "goldenrod", "NA", "steelblue", "seagreen"),
    name = "Clusters & \nOutliers"
  ) +
  labs(
    title = "Neighboring job counts by driving"
  )
```



Statistic Plot with major roadways overlayed

```
ggplot(walking_bk_nta_border) +
  geom_sf(aes(fill = co_type), col = 'lightgrey') +
  scale_fill_manual(
    values = c("red", "goldenrod", "NA", "seagreen"),
    name = "Clusters & \nOutliers"
  ) +
  labs(
    title = "Neighboring job counts"
  )
```



Walking subway Statistic Plot of local area #### Network autocorrelation

Visualization of network

Visualization of network's complement

Global Moran's I

LISA Plot by coloring desire lines