# Traffic Intersection Data

*Dan Fanelli*

*October 19, 2016*

Section 5.5 #3 (p222) - Pick a traffic intersection with a traffic light. Collect data on vehicle arrival times and clearing times. Build a Monte Carlo simulation to model traffic flow at this intersection

Note: traffic at the intersection will be analyzed for both the main road (highway, no traffic lights) and the service roada (with traffic lights)

**The Intersection and Traffic Lights (Arial):**

(4 Directions: NB-MAIN, SB-MAIN, NB-SER[vice], SB-SER[vice])

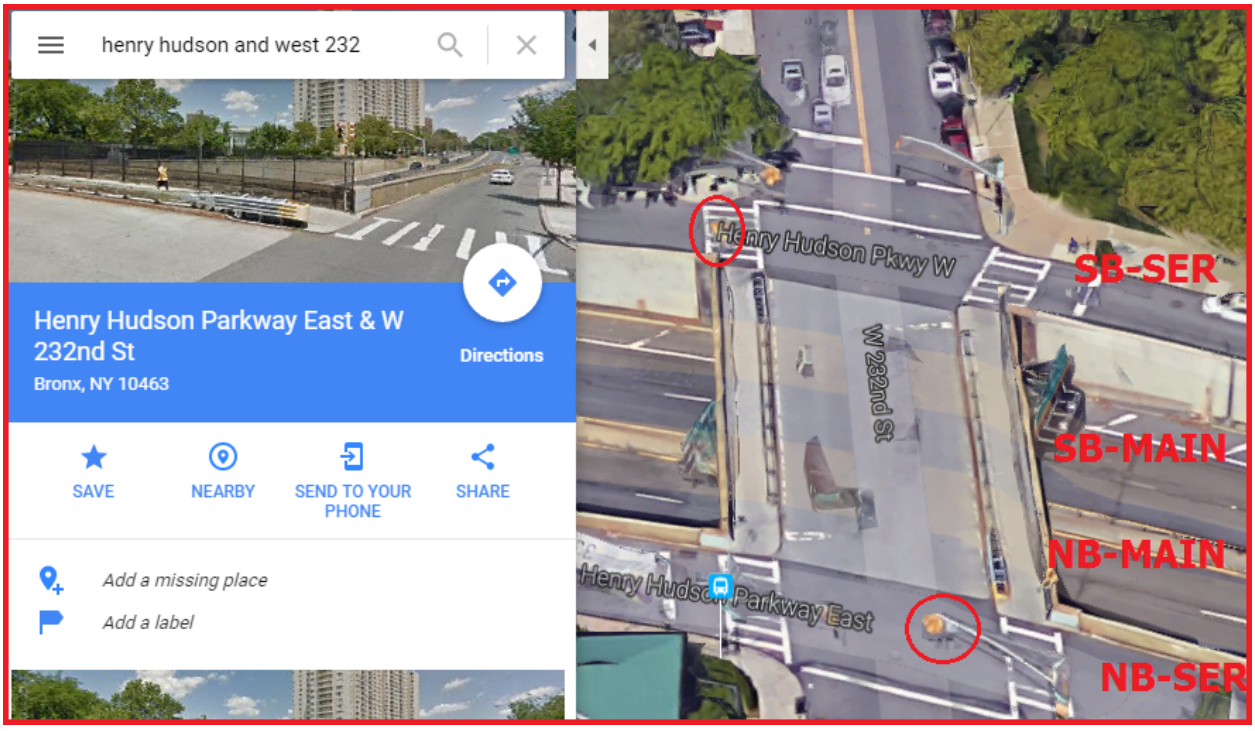



Figure 1:

**The Intersection and Traffic Lights (Map):**

**ALL data for sample date: (02/05/2013)**

| | ID | Segment_ID | Roadway_Name | xFrom | xTo | Direction | Date |
|---|---|---|---|---|---|---|---|
| 5 | 361 | 174566 | HENRY HUDSON PKWY | FAIRFIELD AVE | W 232nd ST | NB-MAIN | 02/05/2013 |
| 12 | 361 | 174566 | HENRY HUDSON PKWY | FAIRFIELD AVE | W 232nd ST | NB-SER | 02/05/2013 |
| 19 | 361 | 174566 | HENRY HUDSON PKWY | FAIRFIELD AVE | W 232nd ST | SB-MAIN | 02/05/2013 |
| 26 | 361 | 174566 | HENRY HUDSON PKWY | FAIRFIELD AVE | W 232nd ST | SB-SER | 02/05/2013 |

| | ID | Segment_ID | Roadway_Name | xFrom | xTo | Direction | Date |
|---|---|---|---|---|---|---|---|
| 33 | 251 | 73220 | HENRY HUDSON PKWY | INDEPENDENCE AVE | W 232nd ST | NB-MAIN | 02/05/2013 |
| 40 | 251 | 73220 | HENRY HUDSON PKWY | INDEPENDENCE AVE | W 232nd ST | NB-SER | 02/05/2013 |
| 47 | 251 | 73220 | HENRY HUDSON PKWY | INDEPENDENCE AVE | W 232nd ST | SB-MAIN | 02/05/2013 |
| 54 | 251 | 73220 | HENRY HUDSON PKWY | INDEPENDENCE AVE | W 232nd ST | SB-SER | 02/05/2013 |

**Transform (Reshape) to Long Format:**

(with DATE and HOUR fields.)

| | ID | Segment_ID | Roadway_Name | xFrom | xTo | Direction | Date |
|---|---|---|---|---|---|---|---|
| 5 | 361 | 174566 | HENRY HUDSON PKWY | FAIRFIELD AVE | W 232nd ST | NB-MAIN | 02/05/2013 |
| 12 | 361 | 174566 | HENRY HUDSON PKWY | FAIRFIELD AVE | W 232nd ST | NB-SER | 02/05/2013 |
| 19 | 361 | 174566 | HENRY HUDSON PKWY | FAIRFIELD AVE | W 232nd ST | SB-MAIN | 02/05/2013 |
| 26 | 361 | 174566 | HENRY HUDSON PKWY | FAIRFIELD AVE | W 232nd ST | SB-SER | 02/05/2013 |
| 33 | 251 | 73220 | HENRY HUDSON PKWY | INDEPENDENCE AVE | W 232nd ST | NB-MAIN | 02/05/2013 |
| 40 | 251 | 73220 | HENRY HUDSON PKWY | INDEPENDENCE AVE | W 232nd ST | NB-SER | 02/05/2013 |
| 47 | 251 | 73220 | HENRY HUDSON PKWY | INDEPENDENCE AVE | W 232nd ST | SB-MAIN | 02/05/2013 |
| 54 | 251 | 73220 | HENRY HUDSON PKWY | INDEPENDENCE AVE | W 232nd ST | SB-SER | 02/05/2013 |
| 61 | 361 | 174566 | HENRY HUDSON PKWY | FAIRFIELD AVE | W 232nd ST | NB-MAIN | 02/05/2013 |
| 68 | 361 | 174566 | HENRY HUDSON PKWY | FAIRFIELD AVE | W 232nd ST | NB-SER | 02/05/2013 |

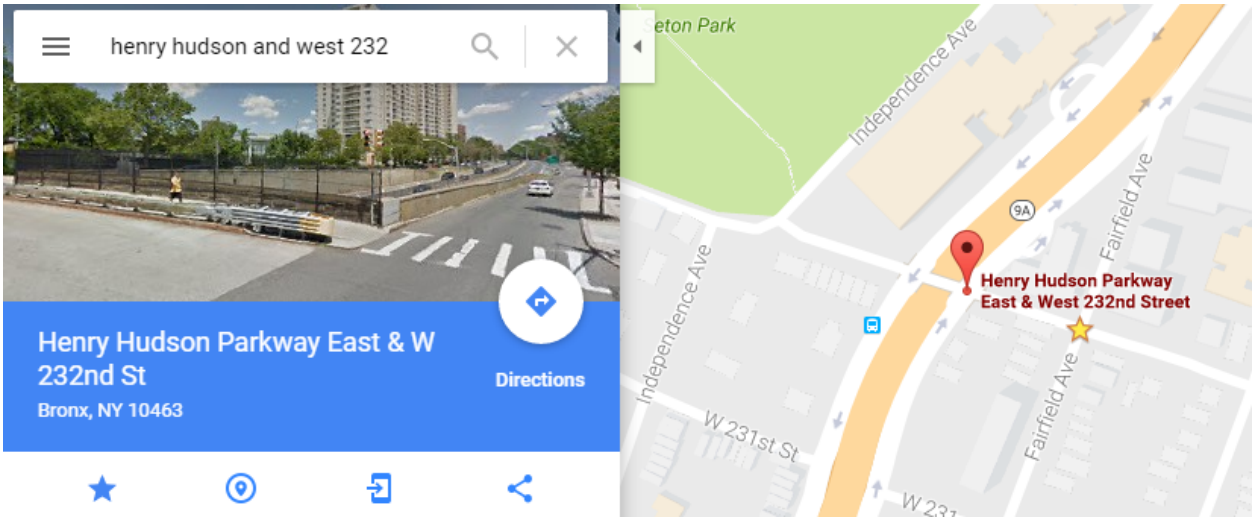
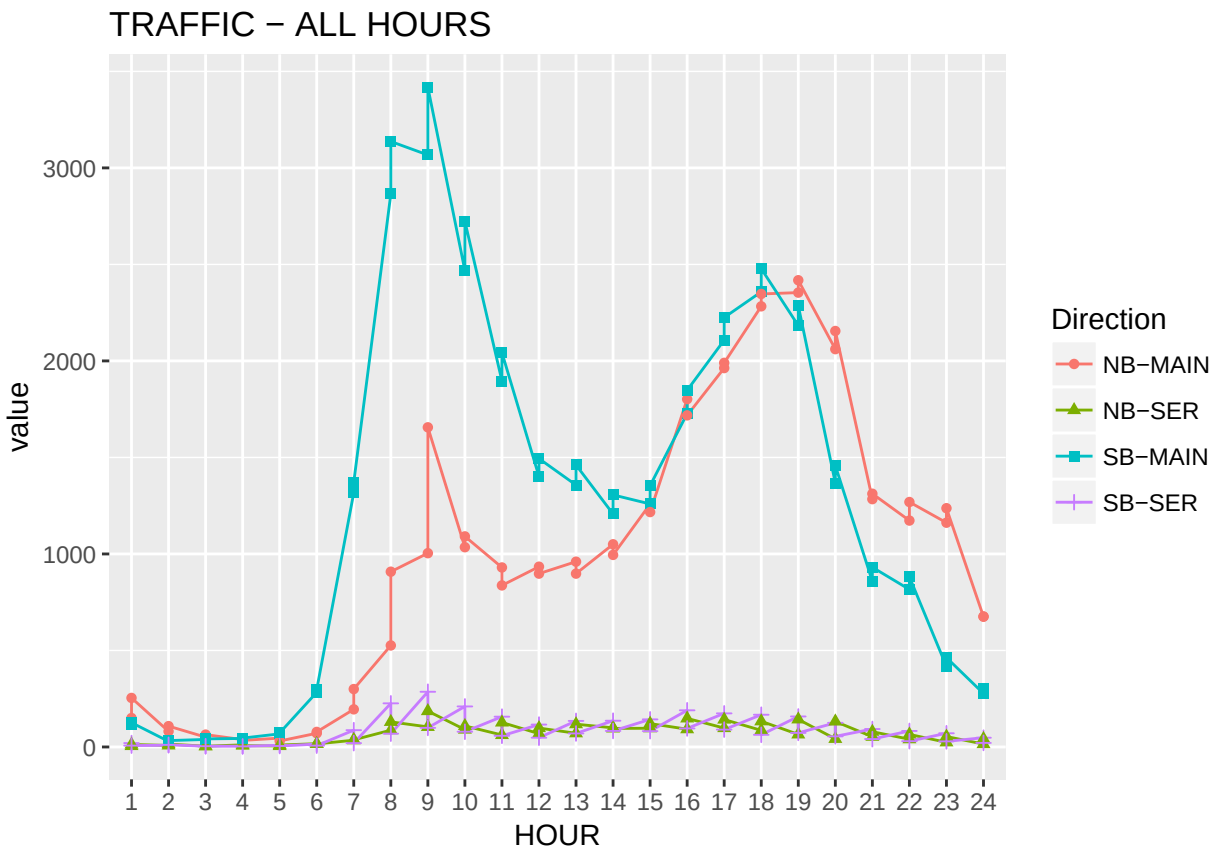
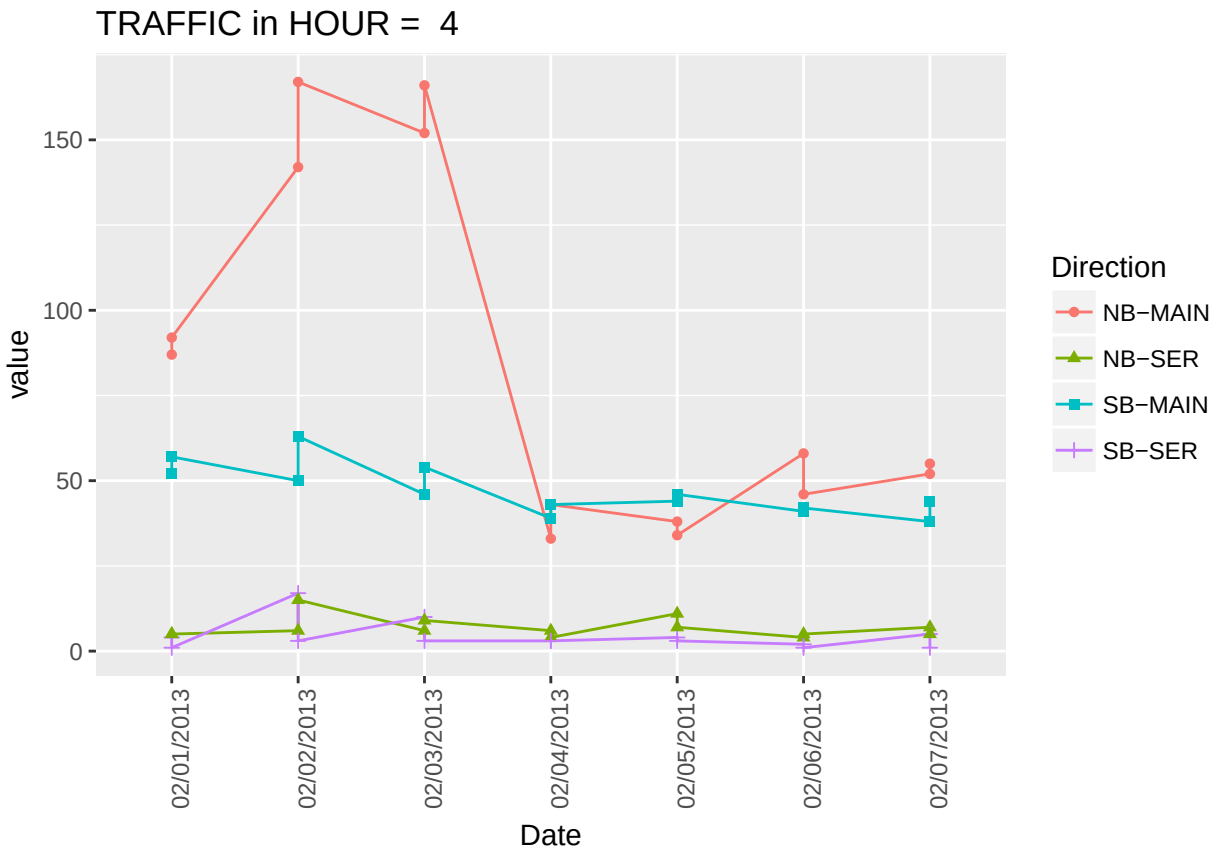



Figure 2:

**HOURLY data for sample date: (02/05/2013)**




Hourly data is NOT RANDOM, so Monte Carlo simulation not applicable. Let's Analze traffic ONLY in ***4th*** HOUR OF THE DAY across multiple days, since perhaps the daily traffic levels will be RANDOM AT THAT TIME and Monte Carlo simulation will then be applicable:




Seems to have some days that are not random, why?

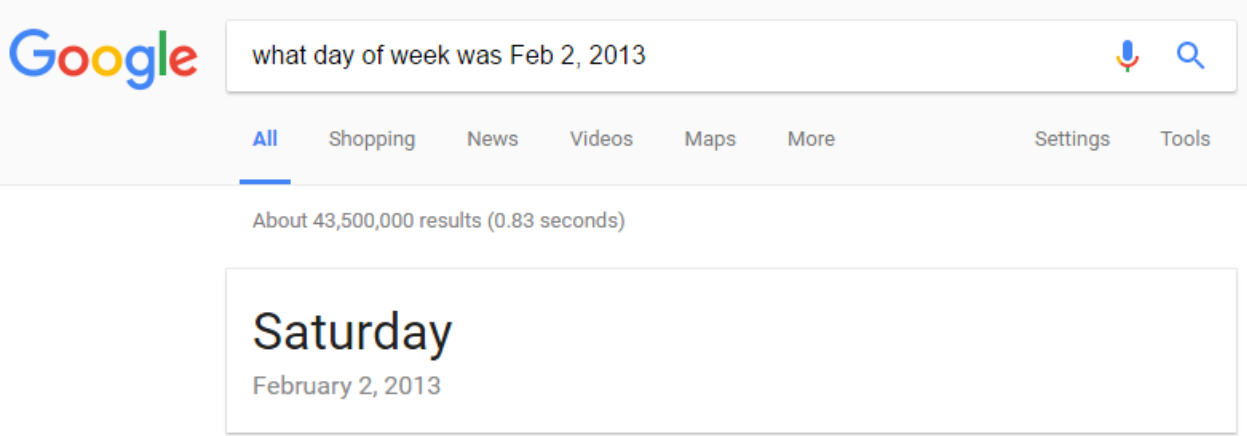
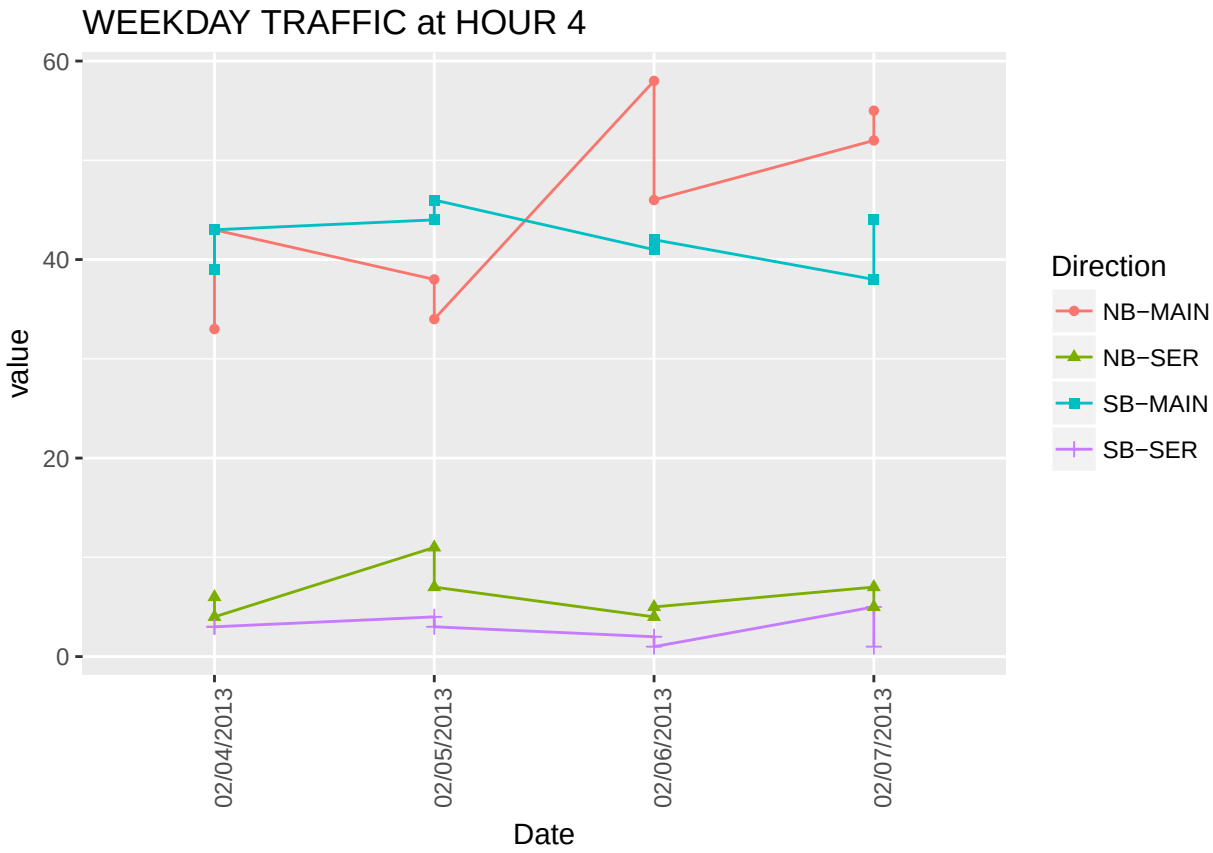



Figure 3:

- Feb 2 and 3 in were Saturday and Sunday
- This explains the extra valume
- So let's only look at the last 4 for rush hour analysis.

**North/South TRAFFIC DATA at HOUR #4:**

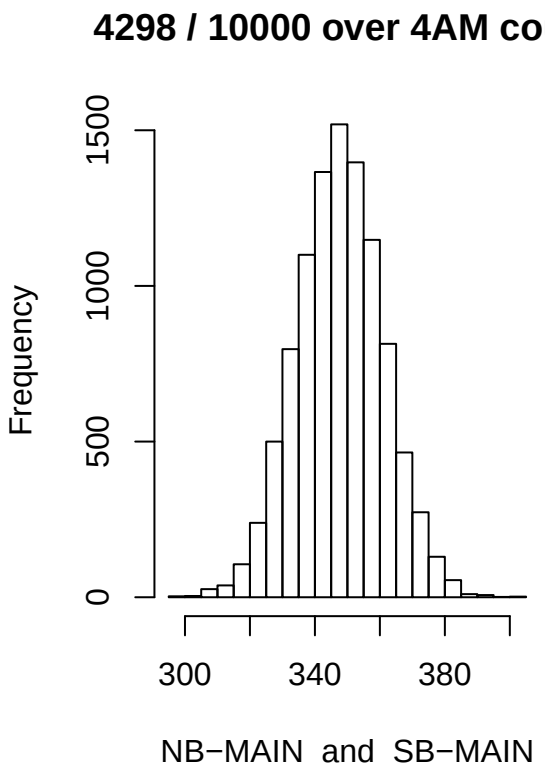


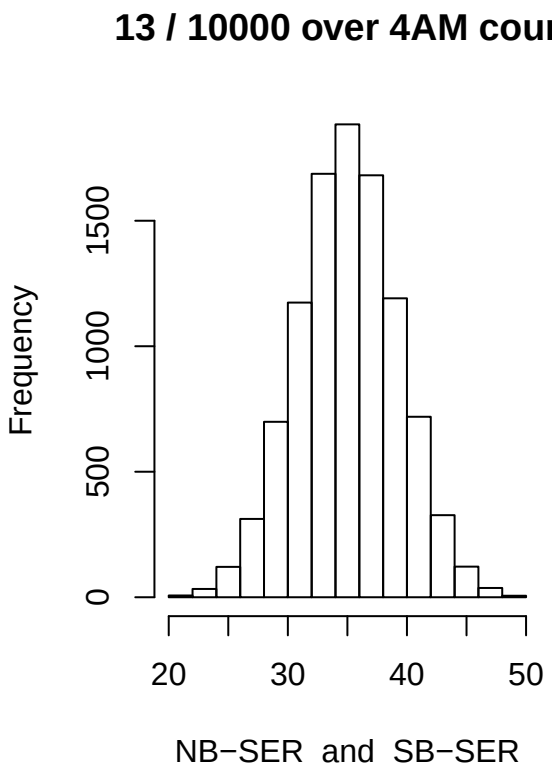

It seems more possible that these values are random, so let's inspect the data for a single day:

| ID | Segment_ID | Roadway_Name | xFrom | xTo | Direction | Date | HO |
|---|---|---|---|---|---|---|---|
| 361 | 174566 | HENRY HUDSON PKWY | FAIRFIELD AVE | W 232nd ST | NB-MAIN | 02/05/2013 | 4 |
| 251 | 73220 | HENRY HUDSON PKWY | INDEPENDENCE AVE | W 232nd ST | NB-MAIN | 02/05/2013 | 4 |
| 361 | 174566 | HENRY HUDSON PKWY | FAIRFIELD AVE | W 232nd ST | NB-SER | 02/05/2013 | 4 |
| 251 | 73220 | HENRY HUDSON PKWY | INDEPENDENCE AVE | W 232nd ST | NB-SER | 02/05/2013 | 4 |
| 361 | 174566 | HENRY HUDSON PKWY | FAIRFIELD AVE | W 232nd ST | SB-MAIN | 02/05/2013 | 4 |
| 251 | 73220 | HENRY HUDSON PKWY | INDEPENDENCE AVE | W 232nd ST | SB-MAIN | 02/05/2013 | 4 |
| 361 | 174566 | HENRY HUDSON PKWY | FAIRFIELD AVE | W 232nd ST | SB-SER | 02/05/2013 | 4 |
| 251 | 73220 | HENRY HUDSON PKWY | INDEPENDENCE AVE | W 232nd ST | SB-SER | 02/05/2013 | 4 |

## Monte Carlo Simulation:

Find p-value for ***null hypothesis***: "In hour ***4*** on a weekday, the northbound and southbound traffic are equal."




**Therefore:**

The probability that ***MAIN*** road fits null hypothesis (no direction bias) at hour ***4*** is: ***0.4298***

The probability that ***SERVICE*** road fits null hypothes (no direction bias) at hour ***4*** is: ***0.0013***

## All Code:

```
HOUR_OF_DAY_TO_USE <- 4
SAMPLE_DATE <- '02/05/2013'
NUM_RUNS_TO_PERFORM <- 10000#0
library(ggplot2)
library(knitr)
library(sqldf)

traffic <- read.csv(file="Traffic_Volume_Counts__2012-2013_.csv",header=TRUE,sep=",");

names(traffic) <- gsub(".", "_", names(traffic), fixed = TRUE)

colnames(traffic)[which(names(traffic) == "From")] <- "xFrom"
colnames(traffic)[which(names(traffic) == "To")] <- "xTo"
colnames(traffic)[which(names(traffic) == "Roadway Name")] <- "Roadway_Name"

all_data <- sqldf("select * from traffic where Roadway_Name = 'HENRY HUDSON PKWY' and direction in ('NB
```

```
for(new_col_name in c(8:31)){
  colnames(all_data)[new_col_name] <- (new_col_name-7)
}

kable(all_data[all_data$Date == SAMPLE_DATE,])

library(reshape2)
all_data <- melt(all_data, id.vars = c("ID", "Segment_ID", "Roadway_Name", "xFrom", "xTo", "Direction",

colnames(all_data)[8] <- "HOUR"

all_data <- all_data[order(all_data$Date, all_data$HOUR, all_data$Roadway_Name, all_data$xFrom, all_data

# SHOW THAT THE RESHAPE WORKED AS PLANNED
kable(head(all_data[all_data$Date == SAMPLE_DATE,], n=10))

ggplot(data=all_data[all_data$Date == SAMPLE_DATE,], aes(x=HOUR, y=value, group=Direction, shape=Direct

ggplot(data=all_data[all_data$HOUR == HOUR_OF_DAY_TO_USE,], aes(x=Date, y=value, group=Direction, shape=

weekdays_data <- all_data[all_data$HOUR == HOUR_OF_DAY_TO_USE,]
#DATES_TO_USE <- c('02/01/2013','02/04/2013','02/05/2013','02/06/2013','02/07/2013')
DATES_TO_USE <- c('02/04/2013','02/05/2013','02/06/2013','02/07/2013')
weekdays_data <- weekdays_data[weekdays_data$Date %in% DATES_TO_USE,]

ggplot(data=weekdays_data, aes(x=Date, y=value, group=Direction, shape=Direction, color=Direction)) + gg

single_weekday_data <- weekdays_data[weekdays_data$Date == SAMPLE_DATE,]

kable(sqldf("select *  from single_weekday_data order by Direction, xFrom, xTo"))
nb_sb_for <- function(NB_TEXT_VAL, SB_TEXT_VAL){
  nb_sum <- sum(weekdays_data[weekdays_data$Direction == NB_TEXT_VAL, ]$value)
  nb_sum
  sb_sum <- sum(weekdays_data[weekdays_data$Direction == SB_TEXT_VAL, ]$value)
  sb_sum

  total_traffic_count_main <- nb_sum + sb_sum
  nb_sum_samples <- c()
  count_over_threshold <- 0
  for(run_num in 1:NUM_RUNS_TO_PERFORM){
    total_1s <- sum(sample(c(0,1),total_traffic_count_main,replace = TRUE))
    nb_sum_samples <- c(nb_sum_samples,total_1s)
    # 2-sided p-test
    if(total_1s >= nb_sum || total_1s <= sb_sum){
      count_over_threshold <- count_over_threshold + 1
    }
  }

  hist(nb_sum_samples, xlab = paste(NB_TEXT_VAL," and ", SB_TEXT_VAL), main = paste(count_over_threshol
  return (count_over_threshold/NUM_RUNS_TO_PERFORM)
}

par(mfrow=c(1,2))
```

```
main_probs <- nb_sb_for('NB-MAIN','SB-MAIN')
service_probs <- nb_sb_for('NB-SER','SB-SER')
##
```

# Social Network

*Dan Fanelli*

*November 22, 2016*

## Social Network Investigation


Figure 1:

**Investigate a social network that is of interest to you: *GENI***

Carefully define what the vertices represent and what the edges represent.

***EDGES: In this model, the edges represent a parent-child relationships***

***NODES: In this model, each node represents a person***

Are there any new modeling techniques that you had to emply?

***No new modelling techniques, just the scraping, the graph, and the Neo4j storage of the graph.***

## Start Node: Kevin Bacon:

```
##  [ 1 ] Kevin Bacon
##      [ 2 ] Edmund Norwood Bacon
##          [ 3 ] Ellis Williams Bacon
##              [ 4 ] Thomas Pryor BACON
##              [ 4 ] Anne Elizabeth BACON
##          [ 3 ] Helen Atkinson Bacon
##              [ 4 ] Robert Comly
##              [ 4 ] Lydia Towsend Comly
##      [ 2 ] Ruth Hilda Holmes
##          [ 3 ] Artemas Holmes
##              [ 4 ] Artemus Henry Holmes
##              [ 4 ] Lillian Holmes
##          [ 3 ] Dorothy Frances Smith
##              [ 4 ] George Campbell Smith
##              [ 4 ] Annie Schwartz
```

```
## [1] "Finished Kevin Bacon"
```


Figure 2:

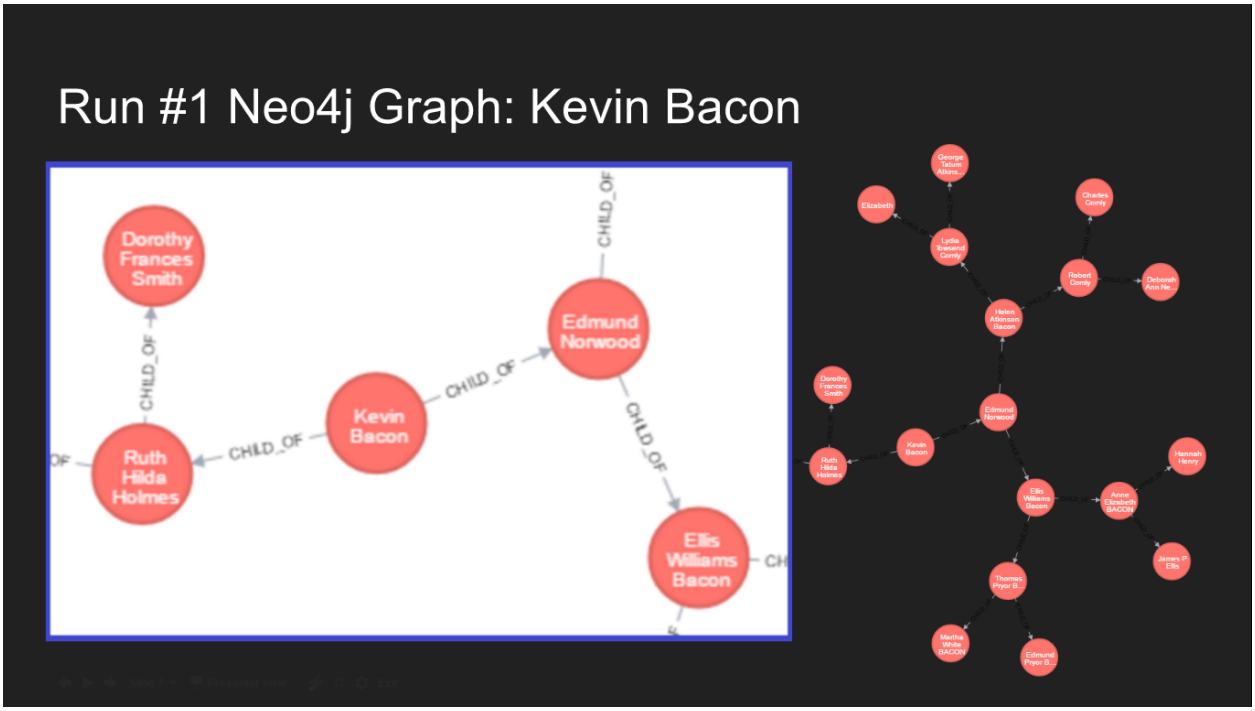



Figure 3:

## Start Node: JFK


Figure 4:

```
##  [ 1 ] John F. Kennedy, 35th President of the USA
##      [ 2 ] Joeseph P. Kennedy, Sr.
##          [ 3 ] Patrick Joseph Kennedy, Jr.
##              [ 4 ] Patrick Joseph Kennedy, Sr.
##              [ 4 ] Bridget Kennedy
##          [ 3 ] Mary Augusta Kennedy
##              [ 4 ] James Hickey
##              [ 4 ] Margaret Hickey
##      [ 2 ] Rose Elizabeth Kennedy
##          [ 3 ] John F. "Honey Fitz" Fitzgerald, Sr.
##              [ 4 ] Thomas Fitzgerald
##              [ 4 ] Rosanna Fitzgerald
##          [ 3 ] Mary Josephine FitzGerald
##              [ 4 ] Michael Hannon
##              [ 4 ] Mary Ann Fitzgerald
```

```
## [1] "Finished JFK"
```

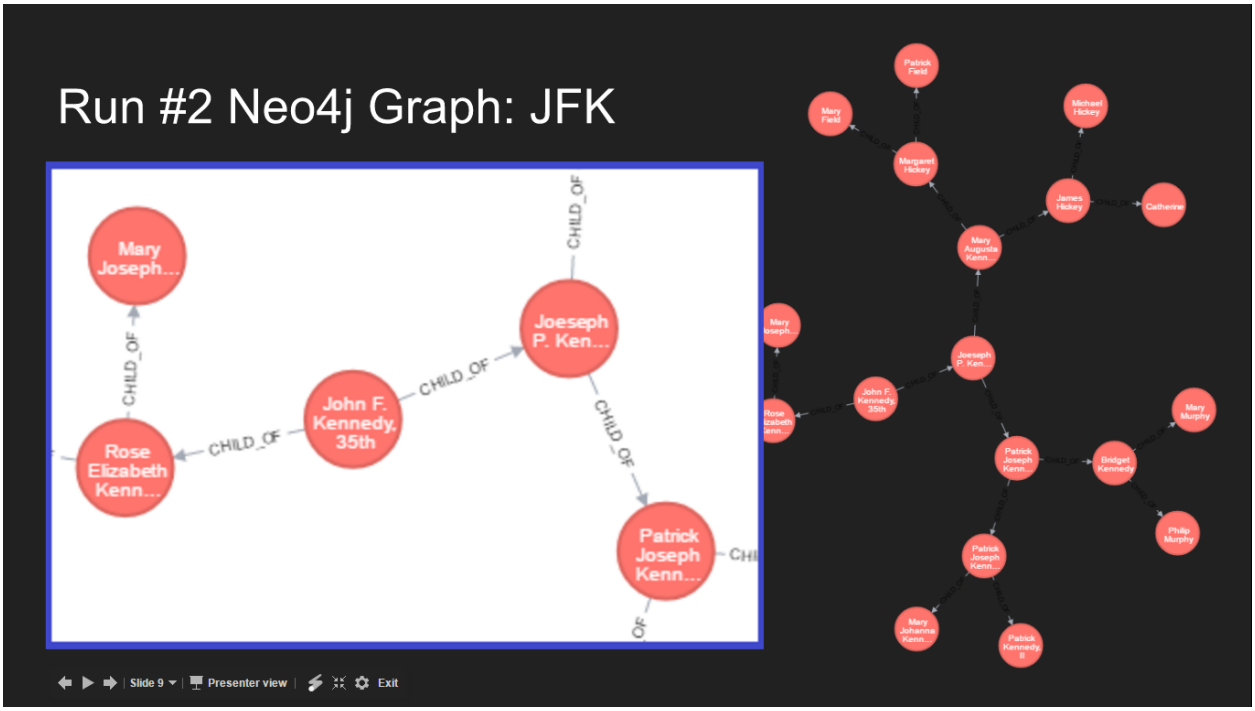



Figure 5:

## All Code:

```
library(rvest)
library(knitr)
library(RNeo4j)

NEO4J_GRAPH_URL <- 'http://localhost:7474/db/data/'
graph = startGraph(NEO4J_GRAPH_URL, username="neo4j", password="neo5j")
clear(graph, FALSE)

addConstraint(graph, "Person", "name")
addConstraint(graph, "Person", "link")

NUM_GENERATIONS <- 3

get_geni <- function(source_person, this_name, this_link, gen_num){

  for(spaces in c(1:gen_num)){
    cat("\t")
  }
  cat("[",gen_num,"]",this_name,"\n")

  page_node <- html(this_link)
  fam_links <- page_node %>% html_nodes("#family_handprint a")
  for(i in c(1:2)){
    f <- fam_links[i]
    next_name <- f %>% html_text()
```

```
    next_link <- f %>% html_attr("href")
    if(!endsWith(next_link, "#")){
      curr_time <- format(Sys.time(), "%H:%M%:s")
      found_person = getOrCreateNode(graph, "Person", name=next_name, link=next_link, time=Sys.time(), g
      relative_rel = createRel(source_person, "CHILD_OF", found_person)
      if(gen_num <= NUM_GENERATIONS){
        gen_above <- gen_num + 1
        get_geni(found_person, next_name, next_link, gen_above)
      }
    }
  }
  return (df)
}
start_name_1 <- "Kevin Bacon"
start_page_1 <- "https://www.geni.com/people/Kevin-Bacon/6000000009325127022"
the_person_1 = createNode(graph, "Person", name=start_name_1, link=start_page_1, time=Sys.time(), gen_al
geni_run_1 <- get_geni(the_person_1, start_name_1, start_page_1, 1)

sprintf('Finished Kevin Bacon')
start_name_2 <- "John F. Kennedy, 35th President of the USA"
start_page_2 <- "https://www.geni.com/people/John-F-Kennedy-35th-President-of-the-USA/600000002880912
the_person_2 = createNode(graph, "Person", name=start_name_2, link=start_page_2, time=Sys.time(), gen_al
geni_run_2 <- get_geni(the_person_2, start_name_2, start_page_2, 1)
sprintf('Finished JFK')
##
```