Bayesian Data Analysis of an Uber Driver's Optimal Daily Circuit

Krishnan Raman

12/3/2020

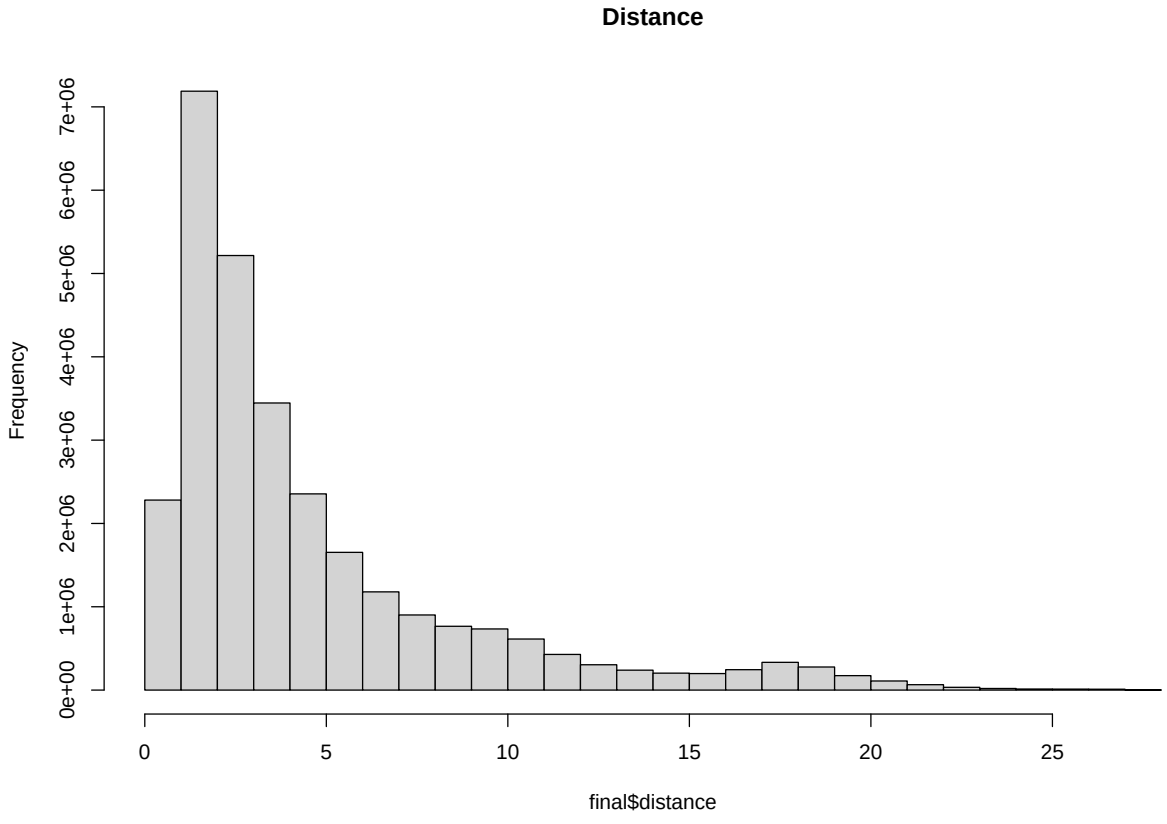
Contents

Uber Dataset

- 29 million rows

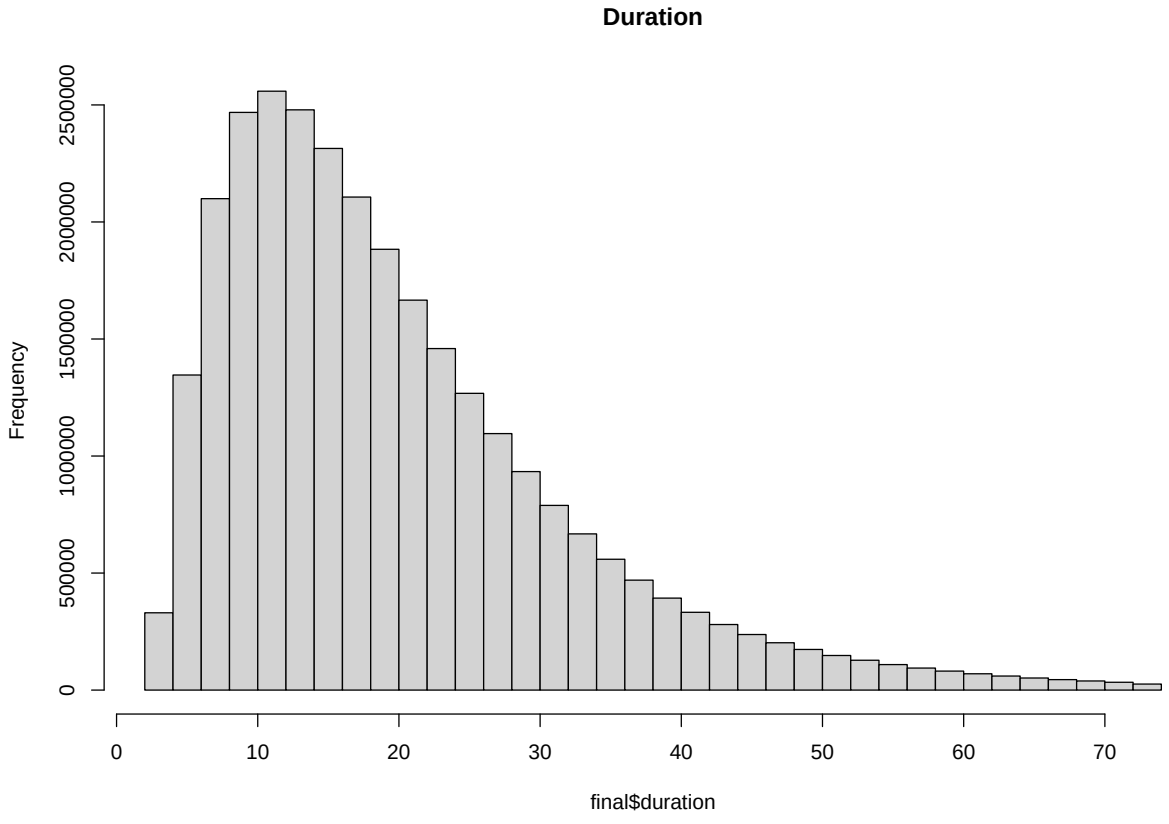
	orig	dest	pickup_datetime	distance	duration	fare
1	7C	6A	2014-09-01 09:00:00	4.25	15.183	15.30
2	7B	15	2014-09-01 18:00:00	10.17	34.083	32.28
3	11	2A	2014-09-01 17:00:00	4.02	17.100	15.57
4	3B	4A	2014-09-01 13:00:00	1.46	6.533	8.00
5	2A	10	2014-09-01 14:00:00	8.31	26.283	26.29
6	5B	4C	2014-09-01 12:00:00	1.04	8.583	8.00

Distance Distribution



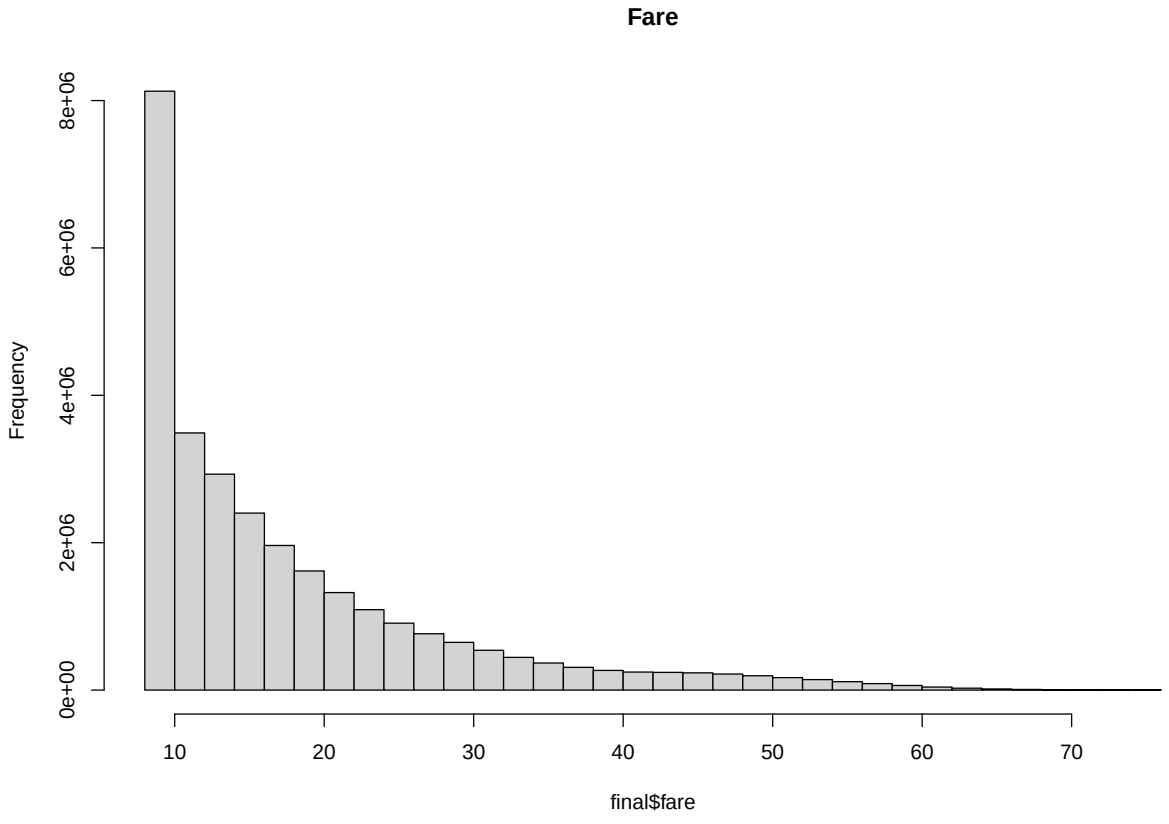

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.40    1.68    2.96    4.60    5.73   27.17
```

Duration Distribution



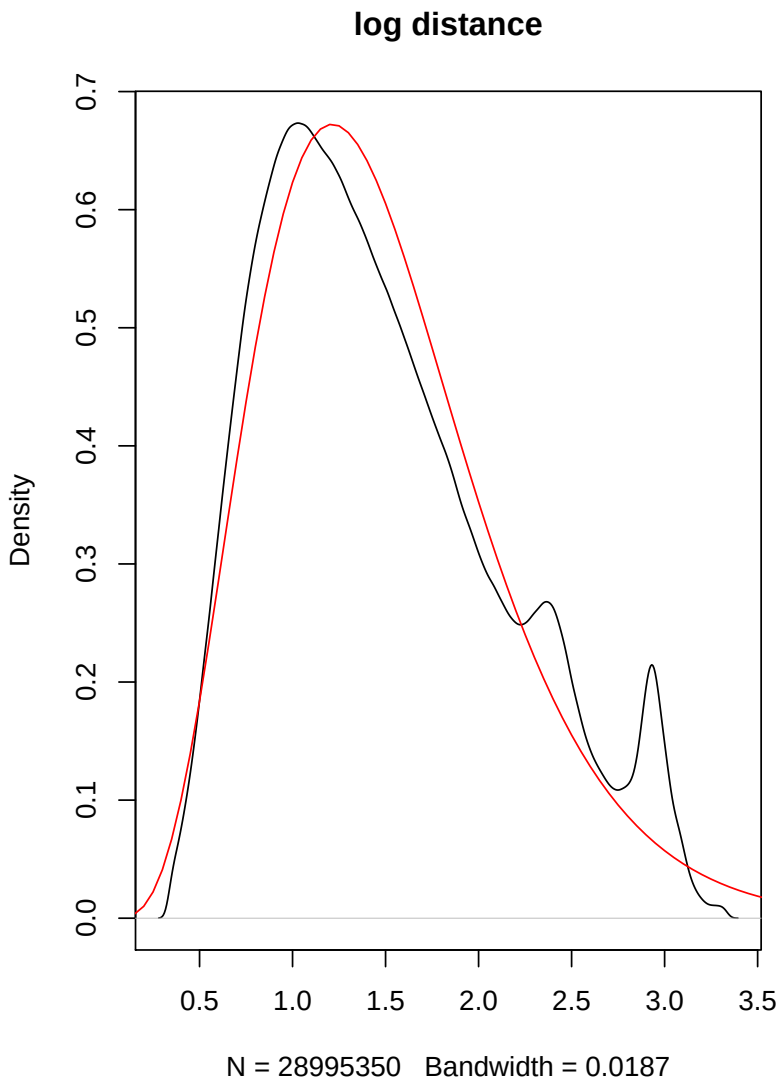

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   2.867  10.800  16.833  19.774  25.617  73.750
```

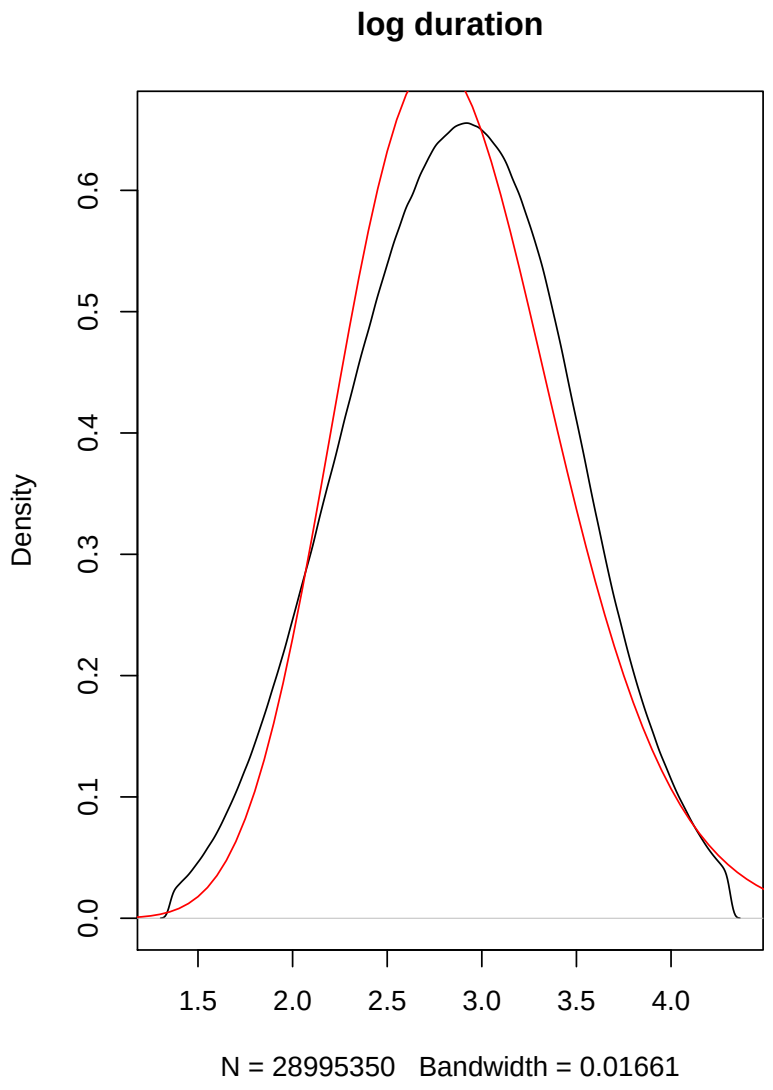
Fare Distribution




```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   8.000   9.533  13.960  17.724  21.829  75.823
```

Modeling log(Distance), log(Duration) with Gamma Priors





Uber Traffic as a Complete Graph

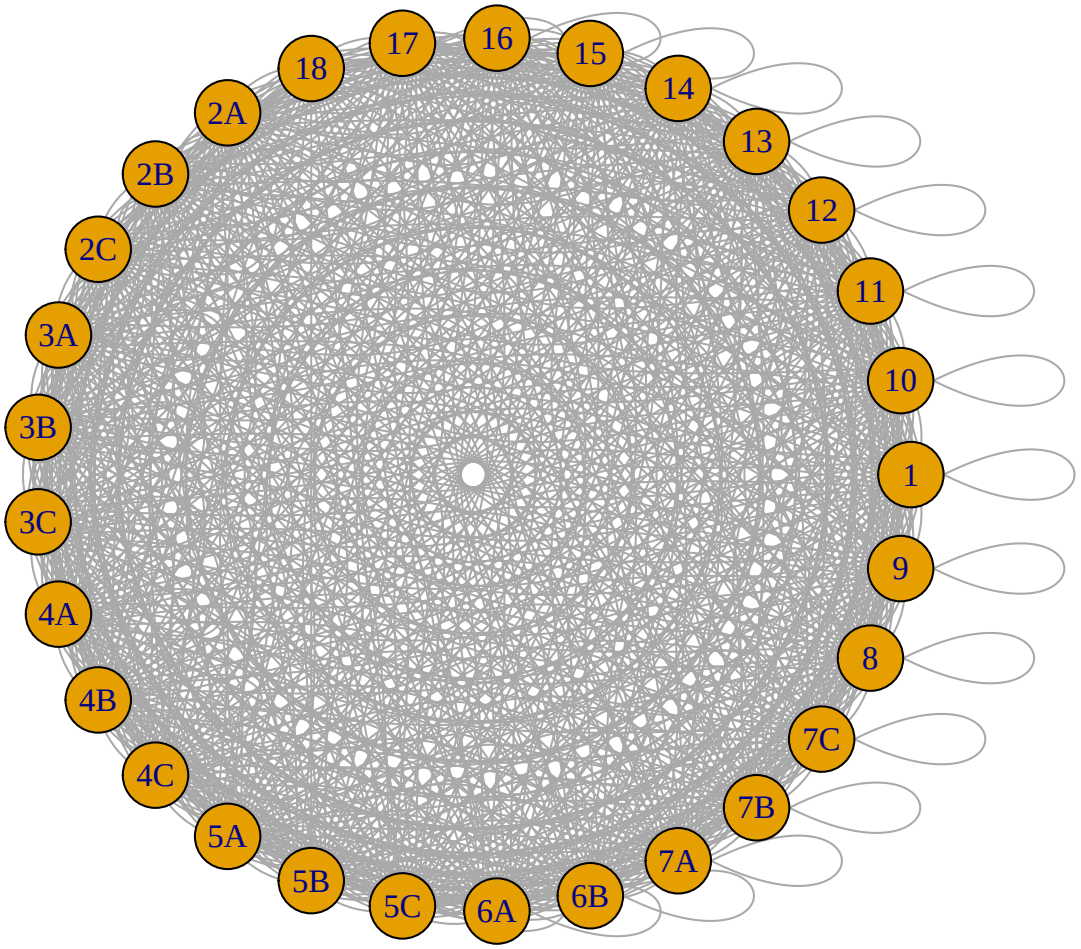
```
library(igraph)
```

```
##
## Attaching package: 'igraph'

## The following objects are masked from 'package:stats':
##
##     decompose, spectrum

## The following object is masked from 'package:base':
##
##     union
```

```
# make graph of original dataset
setwd("~/Desktop/695/uber-tlc-foil-response/uber-trip-data/
all_edges <- readRDS("all_edges.Rda")
n<- dim(all_edges)[1]
```

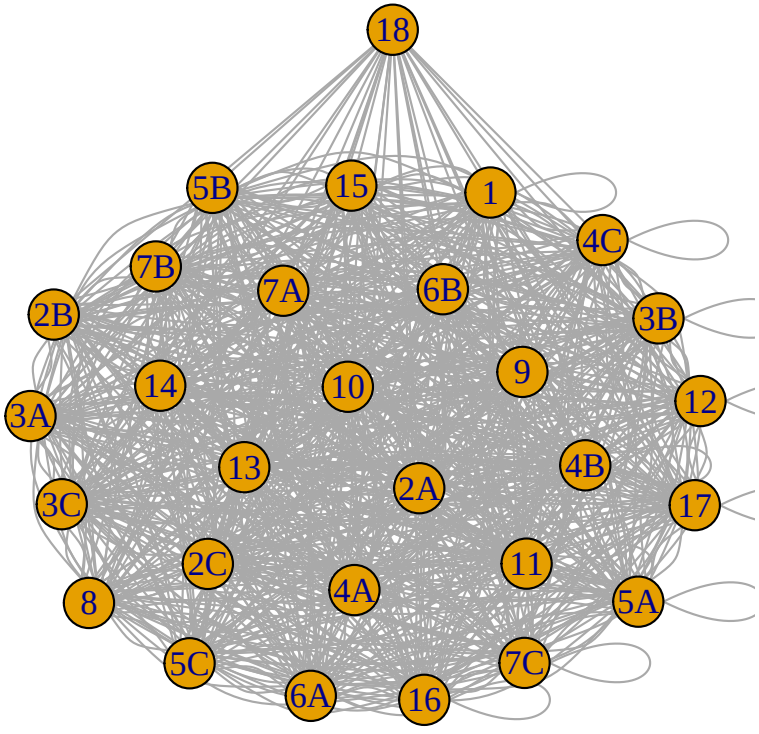
Uber Traffic as a Complete Graph

Vertices: 29 Edges: 812

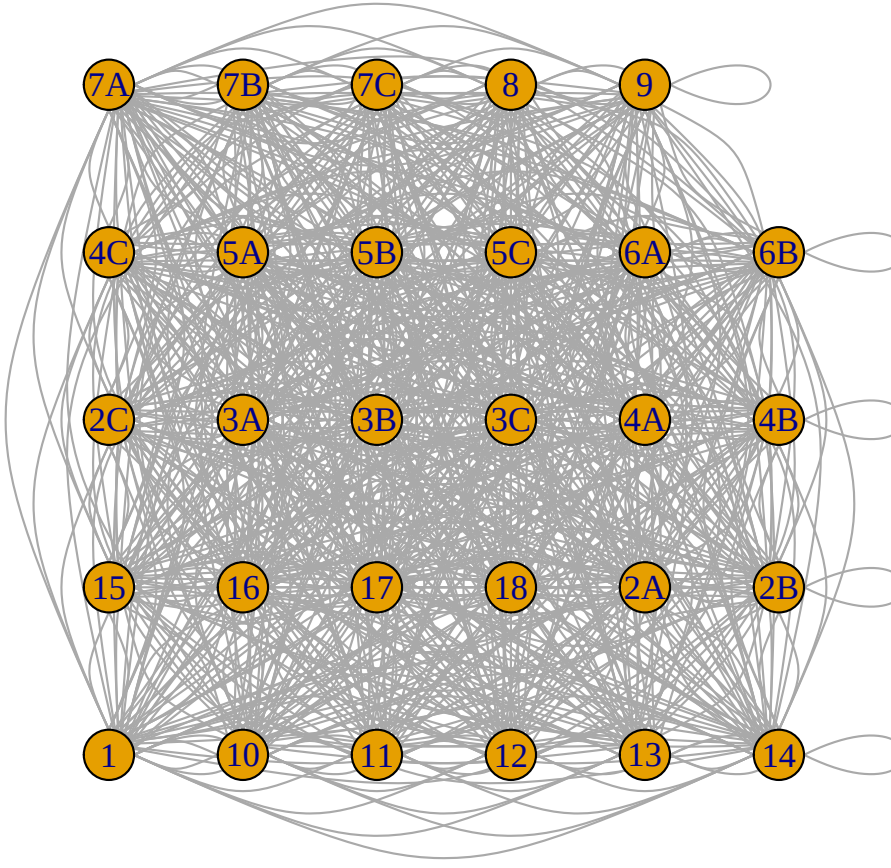


Uber Traffic as a Complete Graph

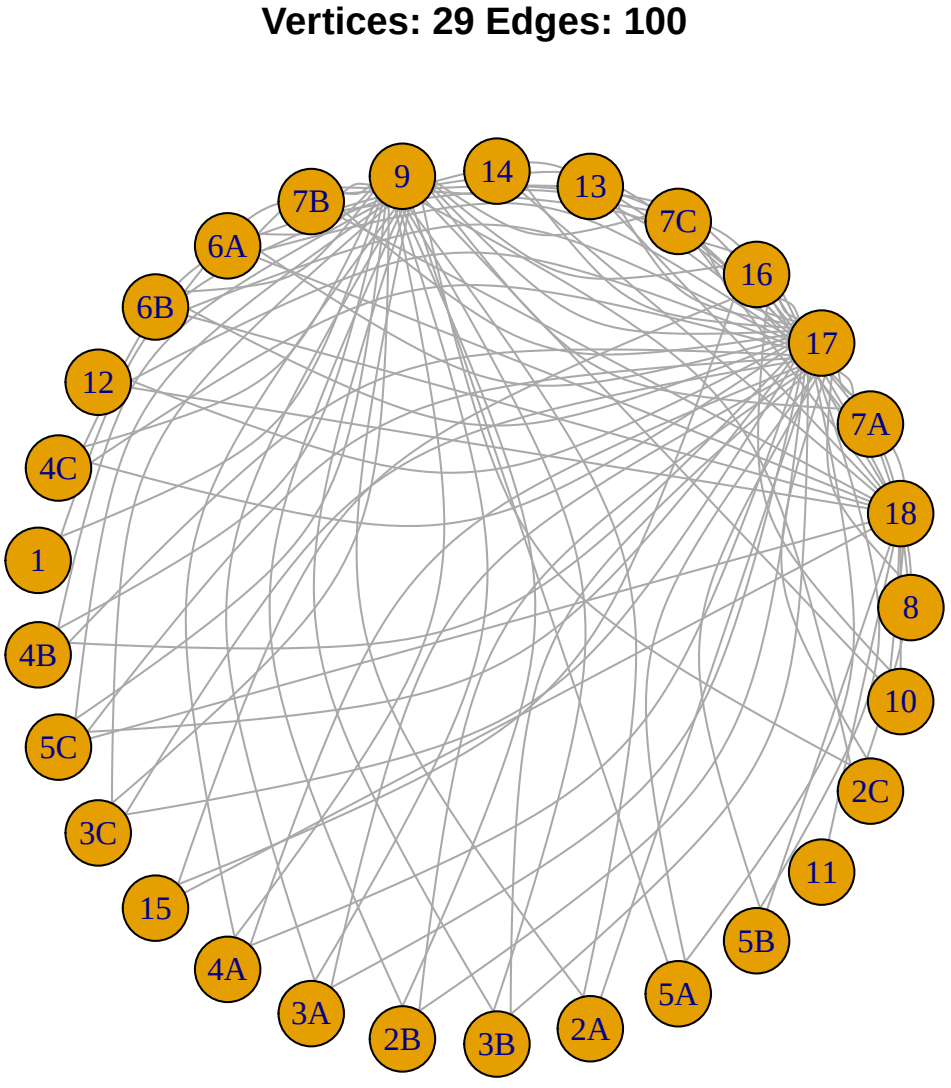
Vertices: 29 Edges: 812



Vertices: 29 Edges: 812



Subgraph induced by Top-100 (most lucrative) edges.



What's an Optimal Daily Circuit ?

- Circuit: Collection of edges
- Want a closed circuit aka Cycle or loop
- Want a "simple path" : each node visited only once

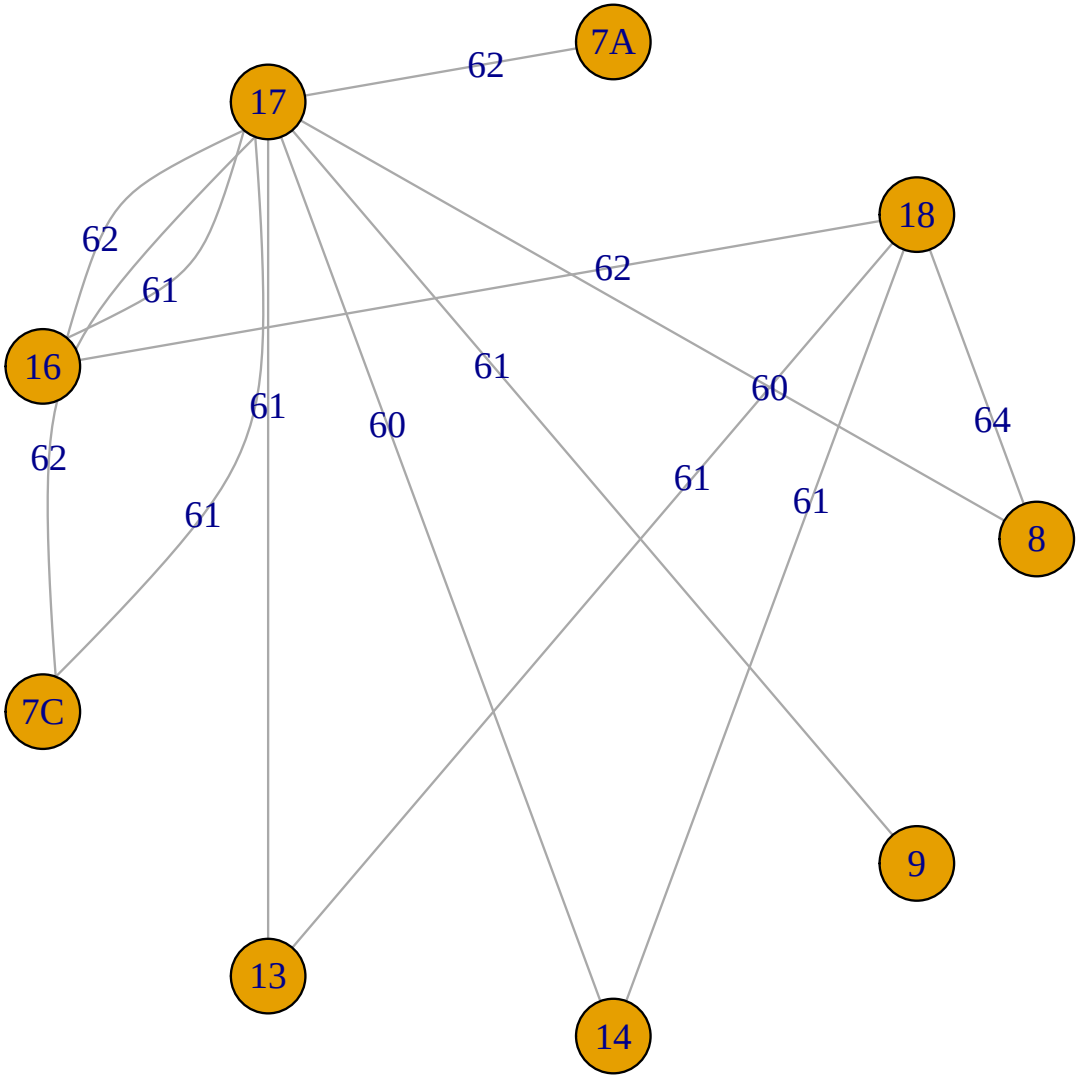
Optimal Circuit

- Circuit that makes the most money for the Uber Driver

Optimal Daily Circuit

- Driver works 9AM-5PM: 8 HOURS
- Make 1 Uber trip per hour
- Want an 8-cycle aka Simple Path with 8 Nodes aka 8-gon

Does a $60 per edge sparse graph contain an 8-gon ?



How many edges guarantee an 8-gon ?

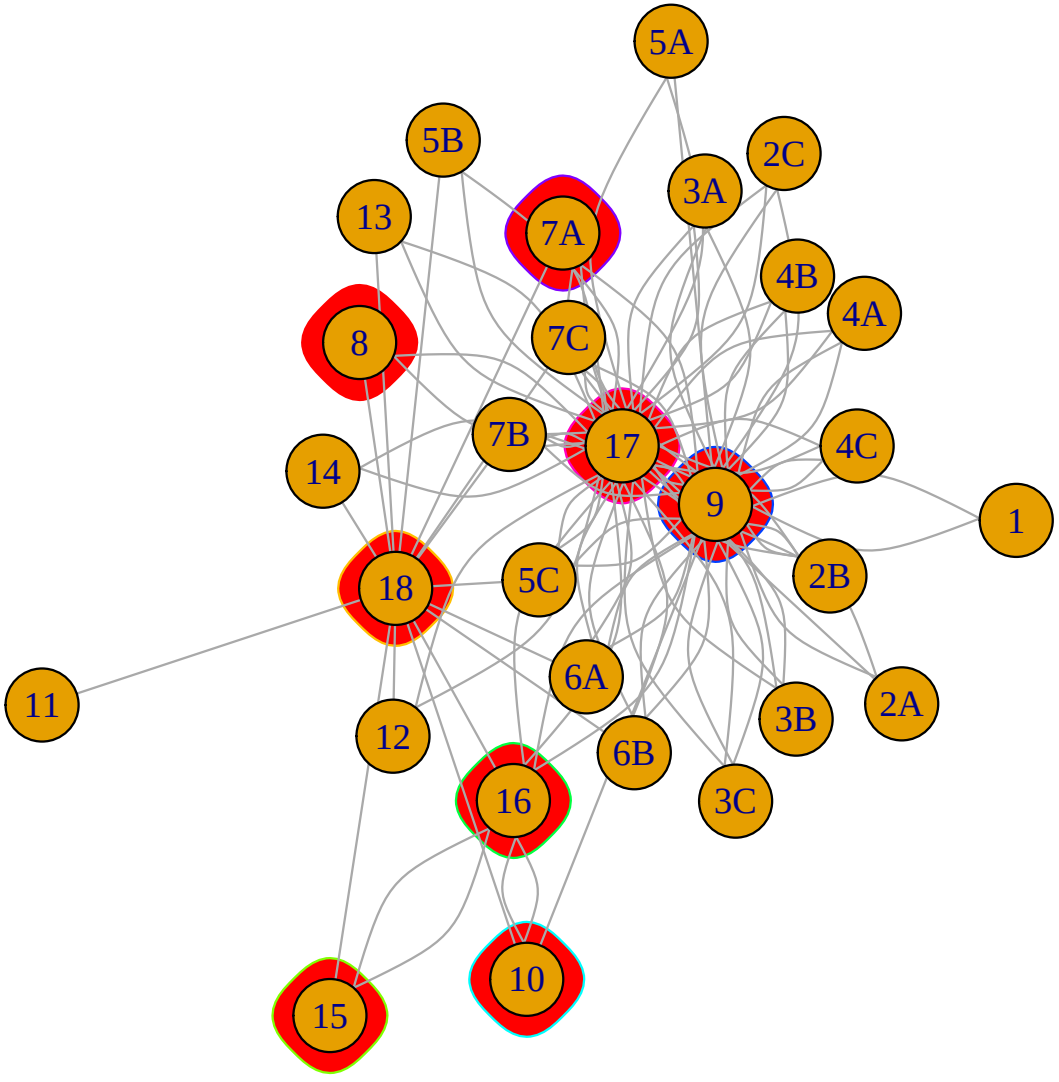
```
for(fares in seq(60,44,-4)) {
  highfare_matrix = edgematrix[as.numeric(edgematrix[,3]) >
  g2<-graph_from_edgelist(highfare_matrix[,1:2], directed=F
  lengths = c()

  for (v in V(g2)$name) {
    res = all_simple_paths(g2,v)
   lengths = c(lengths, max(sapply(1:length(res), function(
  }
  cat(sprintf("Fare: $%.0f, Longest path length: %.0f, Vert
}
```

```
## Fare: $60, Longest path length: 5, Vertices:9 Edges 13
## Fare: $56, Longest path length: 5, Vertices:13 Edges 24
## Fare: $52, Longest path length: 7, Vertices:20 Edges 44
## Fare: $48, Longest path length: 8, Vertices:29 Edges 89
```

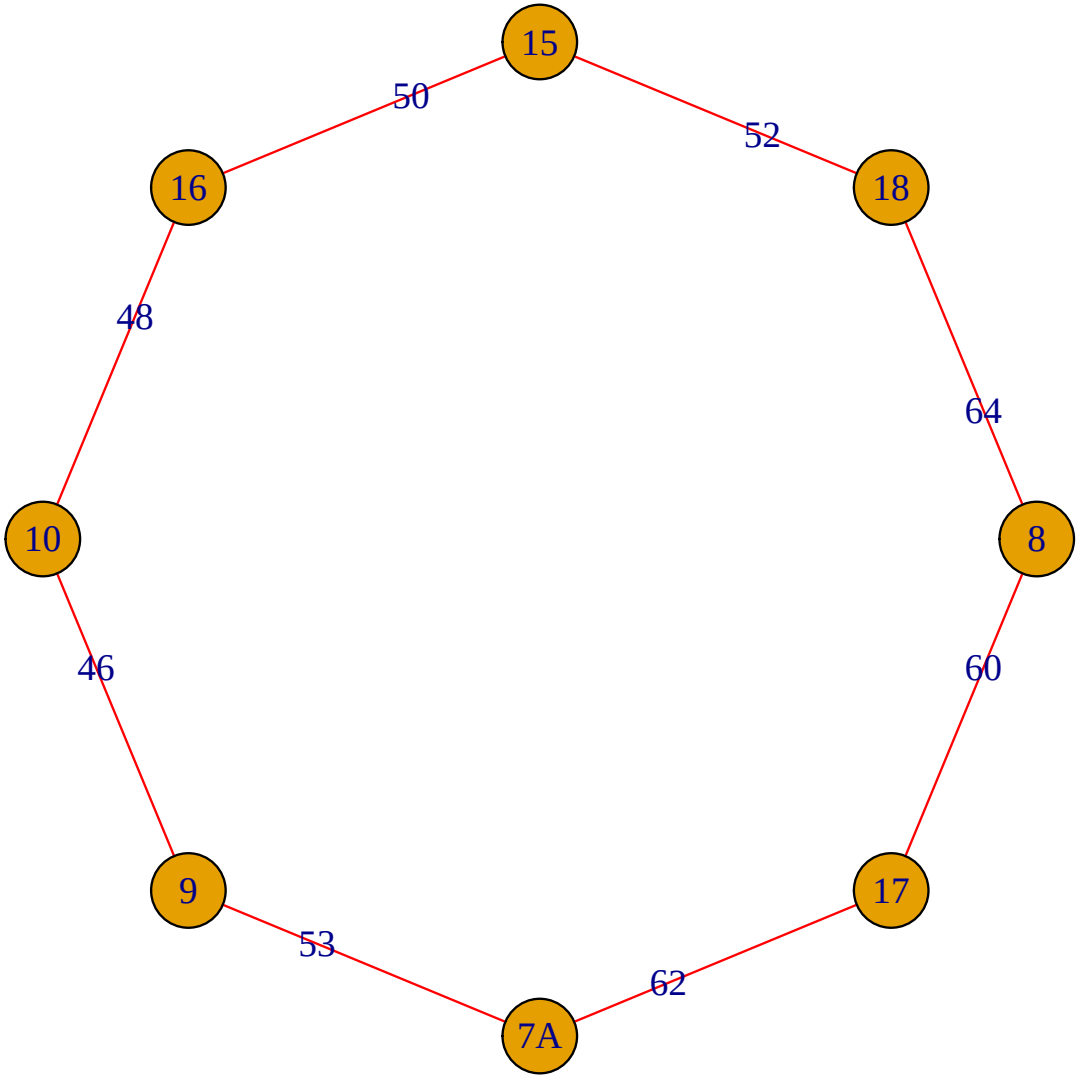
Finding the 8-gon

```
# visualize the graph with simple path of length 8
highfare_matrix = edgematrix[as.numeric(edgematrix[,3]) > 4
highfare_weights = round(as.numeric(highfare_matrix[,3]))
g3<-graph_from_edgelist(highfare_matrix[,1:2], directed=FAI
paths_of_length_8 = c()
found=FALSE
for (src in V(g3)$name) {
    res = all_simple_paths(g3,src)
    for(i in 1:length(res)) {
      l = length(res[[i]])
      last = res[[i]][l]$name
      d = distances(g3,v=last,to=src)
      if (l == 8 & d[1] == 1) {
          paths_of_length_8 = c(paths_of_length_8, res[[i]]
          print(res[[i]])
      }
```

The Optimal 8-gon!



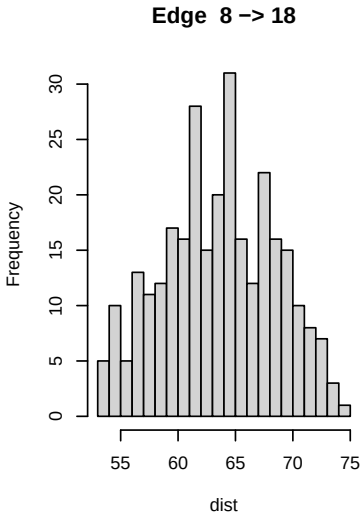
How much does the 8-gon Uber driver make ?

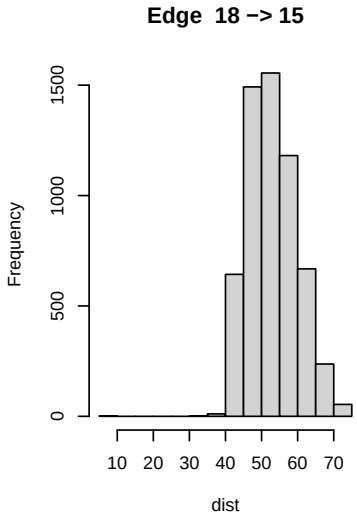


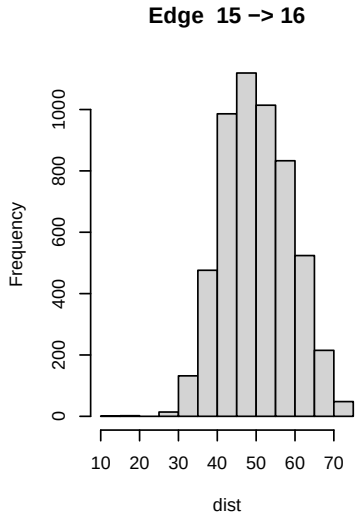
Optimal Circuit Summary Statistics

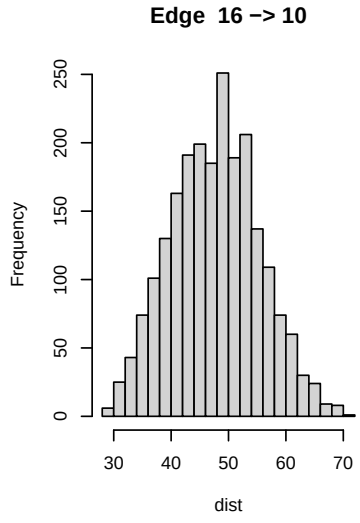
	Src	Dest	Mean	Median	Sigma	Var
1	8	18	63.54	63.69	4.90	23.97
2	18	15	53.06	52.44	6.71	45.06
3	15	16	50.18	49.72	8.65	74.84
4	16	10	47.70	47.86	7.69	59.07
5	10	9	47.20	46.39	7.40	54.79
6	9	7A	54.00	53.42	4.66	21.70
7	7A	17	61.56	61.96	6.22	38.72
8	17	8	60.19	59.89	6.04	36.47

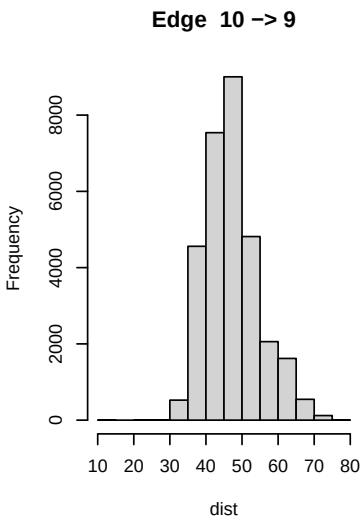
Optimal Circuit: Edge Distributions

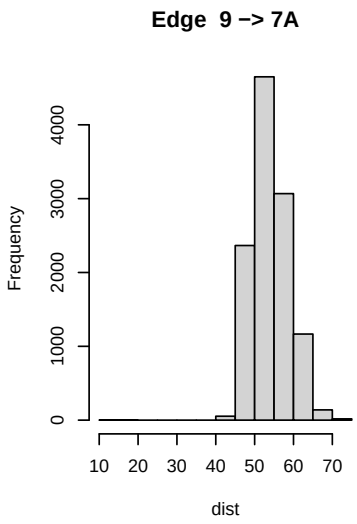


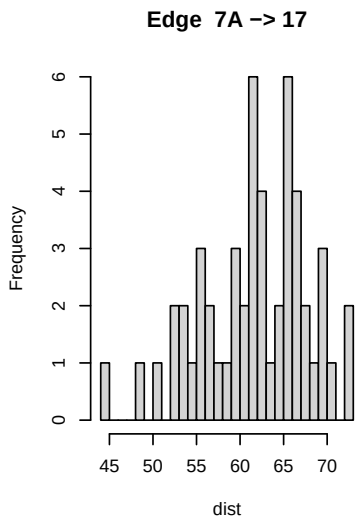


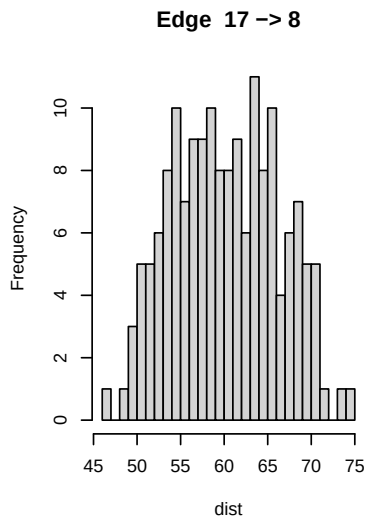




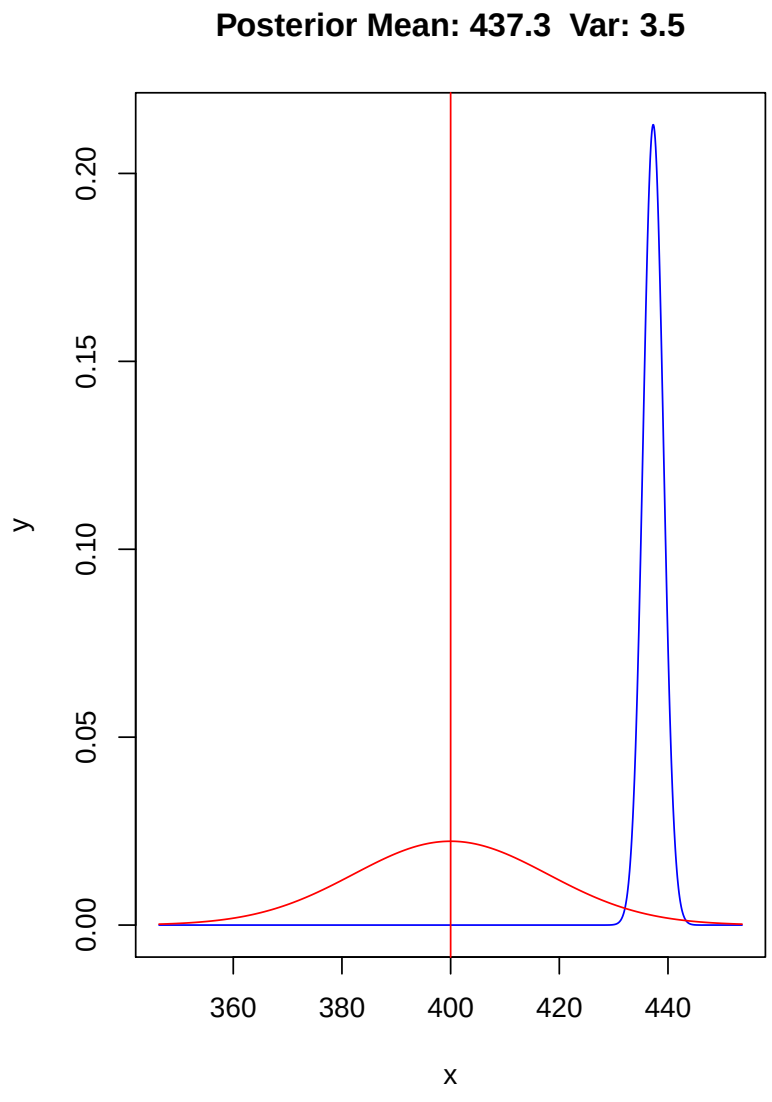


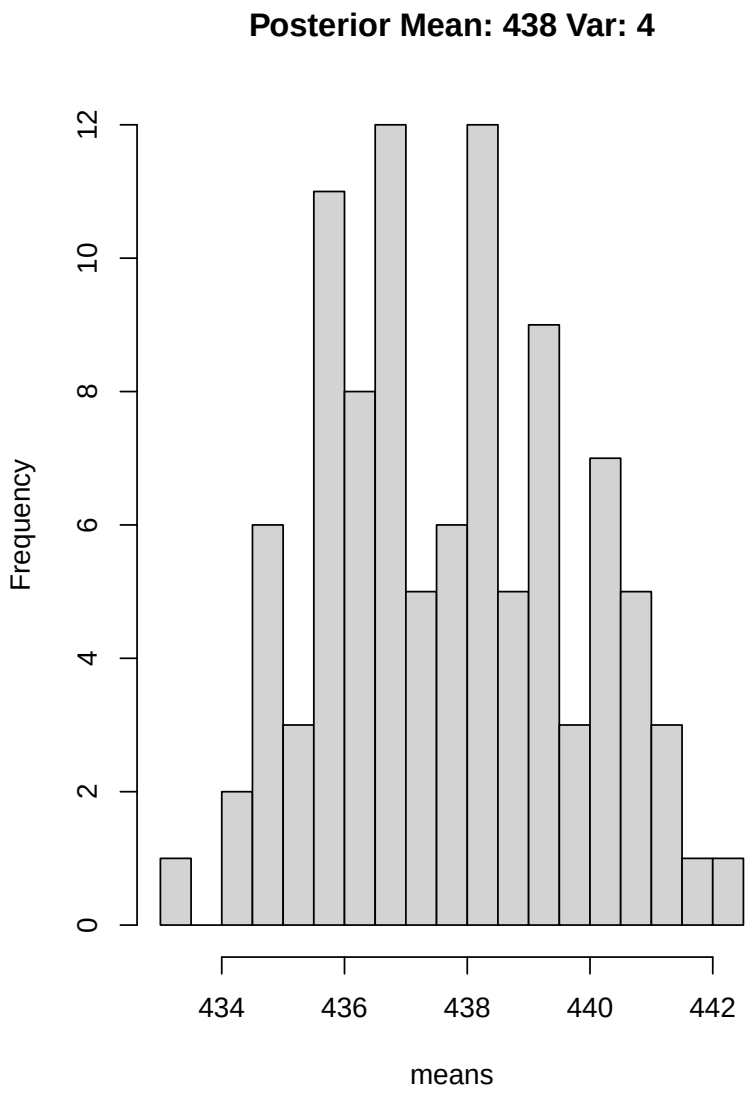






Posterior Mean of the Optimal Daily Circuit



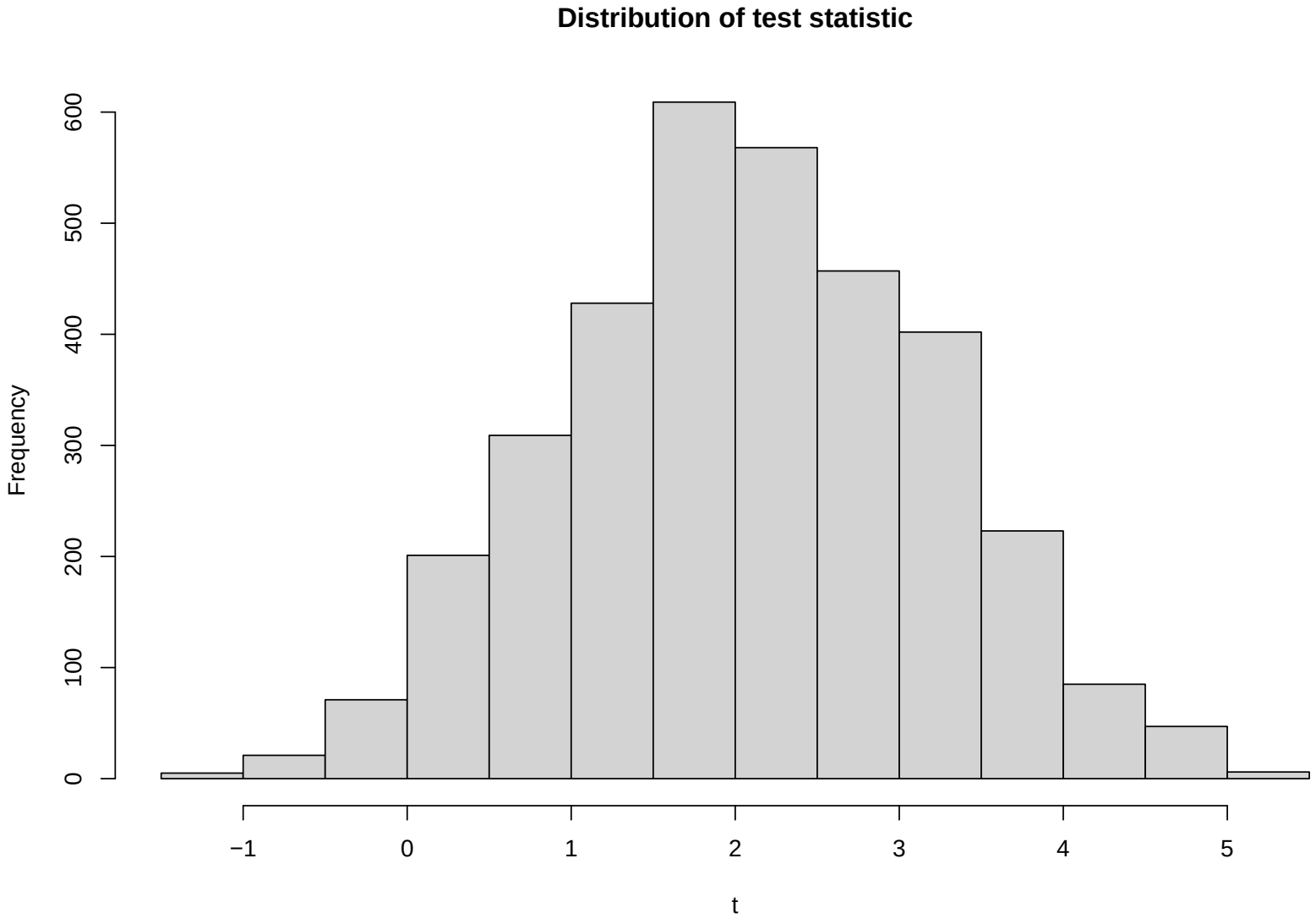


Comparing two Uber circuits via Confidence Distributions

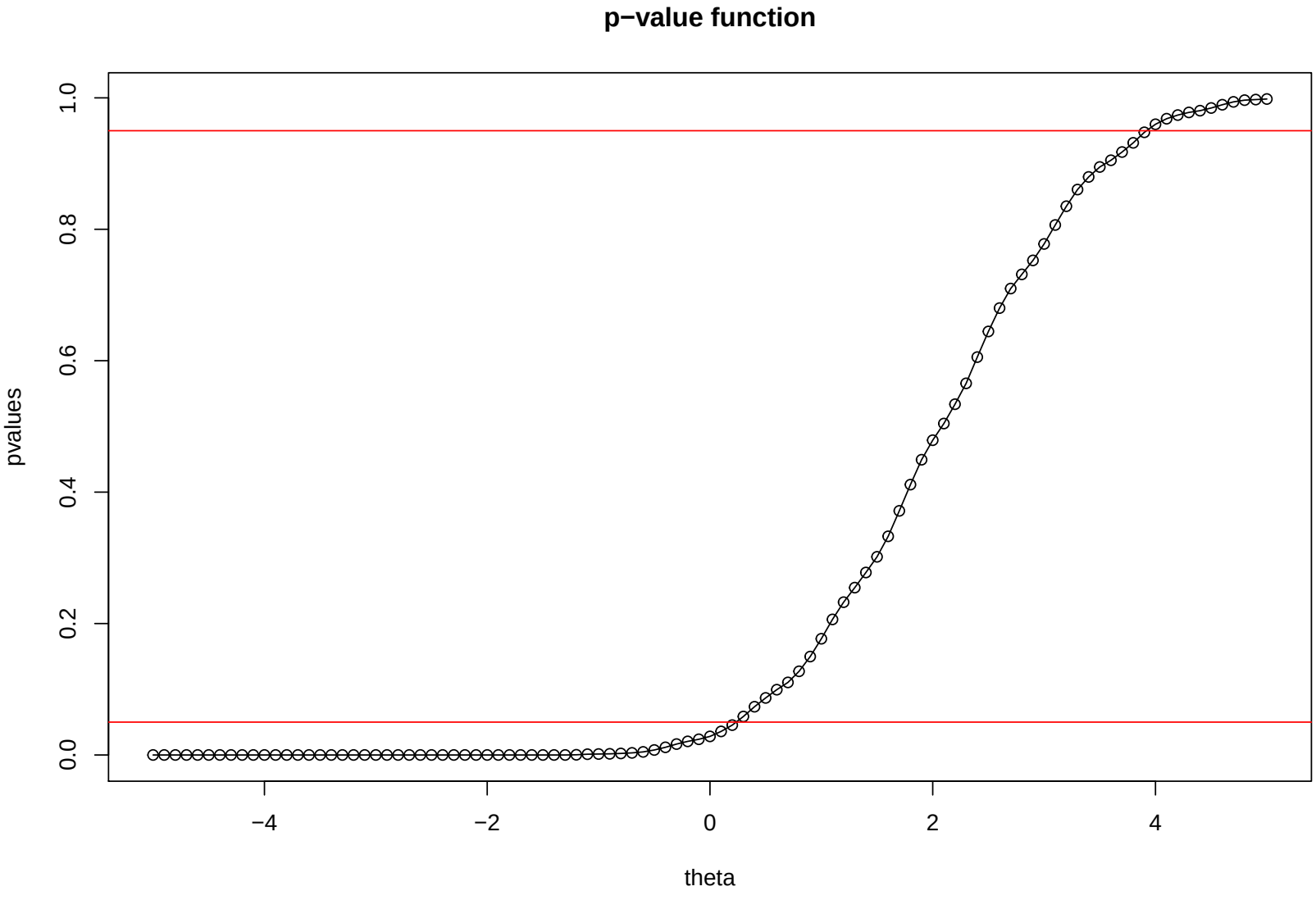
- Professor Don Rubin: Fisher Randomization Test is a Stochastic Proof-By-Contradiction.
- FRT = distribution-free test to compare two (multimodal) Uber circuits
- Fisher Sharp Null compares individual potential outcomes $Y_i(1)$ *vs* $Y_i(0)$ for every observation.
- An assignment is a boolean vector over two weeks.
- An assignment simply means on the given day, the Uber driver drove Circuit c2.
- A non-assignment means the wages would come from Ckt c1 aka Null Hypothesis.
- Compute the biweekly average and compare with the null hypothesis biweekly average
- This comparison is a simple difference test statistic.

```
##       ab    bc    ca    ef    fg    ge Ckt_c1 Ckt_c2
## 1  40.07 51.10 55.51 52.58 52.85 52.10  156.6  157.0
```

Comparing two Uber circuits via Confidence Distributions



Comparing two Uber circuits via Confidence Distributions



WIP

- Alternate (non-Gaussian) modeling choices for Circuits
- Inference via MCMC/Stan & Variational Bayes
- Compare naive driver vs optimal driver

THANK YOU