On-Demand

Mingze Li 300137754

2025-02-04

```r
library(mvtnorm)
library(traveltimeCLT)
library(data.table)
```

```
## Warning:   'data.table' R 4.3.3
```

```r
library(dplyr)
```

```
##
##    'dplyr'

## The following objects are masked from 'package:data.table':
##
##     between, first, last

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(ggplot2)
library(traveltimeHMM)
```

```
##
##    'traveltimeHMM'

## The following objects are masked from 'package:traveltimeCLT':
##
##     rules2timebins, time_bins, time_bins_functional,
##     time_bins_readable, to7daybins
```

```r
library(lubridate)
```

```
##
##    'lubridate'
## The following objects are masked from 'package:data.table':
##
##     hour, isoweek, mday, minute, month, quarter, second, wday, week,
##     yday, year

## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
library(patchwork)
```

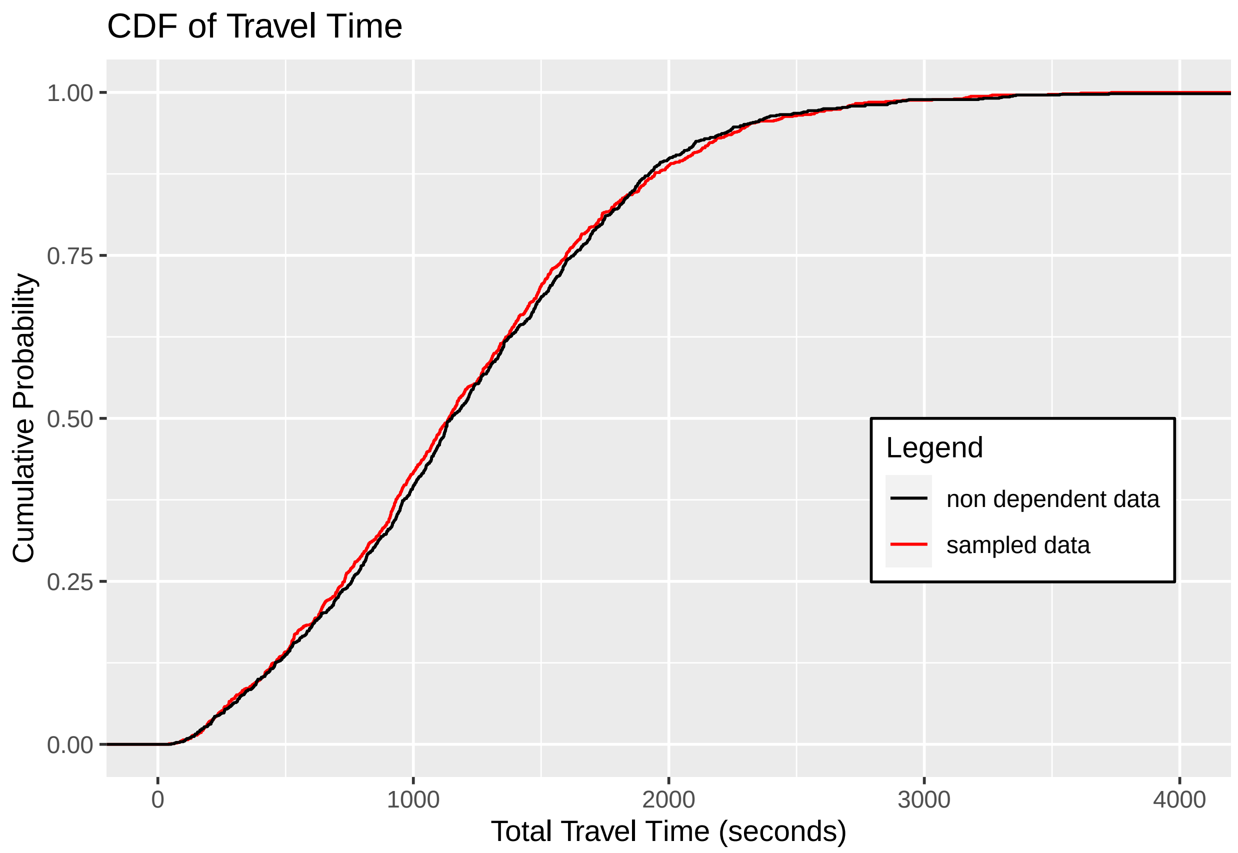
Sample data

```
timebin_x_edge_sorted=read.csv("data/timebin_x_edge_sorted.csv")
timebin_x_edge=read.csv("data/timebin_x_edge.csv")
id = sample(unique(timebin_x_edge$trip),1000)
sampled_trips = timebin_x_edge[timebin_x_edge$trip %in% id,]
sampled_trips <- sampled_trips%>% arrange(desc(trip))
sampled_time <- sampled_trips%>%group_by(trip)%>%
  summarise(sampled_time=sum(duration_secs))
log_no_0<-function(x){
  l=length(x)
  result=c()
  for (i in 1:l) {
      if(x[i]==0)result=c(result,0)
      else result=c(result,log(x[i]))
  }
  result
}
sd_na_is_0<-function(x){
  if(length(x)>=2)return(sd(x))
  else return(0)
}
```

Non-dependence model

```
non_dependent_simulator <- function(edges,rho=0.31){
  l <- length(edges)
  U <- runif(l)
  mu <- (timebin_x_edge_sorted[match(edges,timebin_x_edge_sorted$timebin_x_edge_continuous), 4])
  sigma <- (timebin_x_edge_sorted[match(edges,timebin_x_edge_sorted$timebin_x_edge_continuous), 5])
  sum(exp(mu + sigma * qnorm(U)))
}
non_dependent_time <- sampled_trips%>%group_by(trip)%>%
  summarise(simulated_time=non_dependent_simulator(timebin_x_edge_continuous))
travel_time <- sampled_time
travel_time$non_dependent_time <- non_dependent_time$simulated_time
```

```
plot1<-ggplot(travel_time) +
  stat_ecdf(aes(x = sampled_time,color="sampled data")) +
  stat_ecdf(aes(x = non_dependent_time,color="non dependent data")) +
  labs(title = "CDF of Travel Time", x = "Total Travel Time (seconds)", y = "Cumulative Probability")+
  coord_cartesian(xlim = c(0, 4000), ylim = c(0, 1))+
  scale_color_manual(name="Legend",values = c("black","red"))+
    theme(legend.position = c(0.95, 0.5),
      legend.justification = c(1, 1),
      legend.text.align = 0,
      legend.background = element_rect(color = "black", fill = "white"))
plot1
```

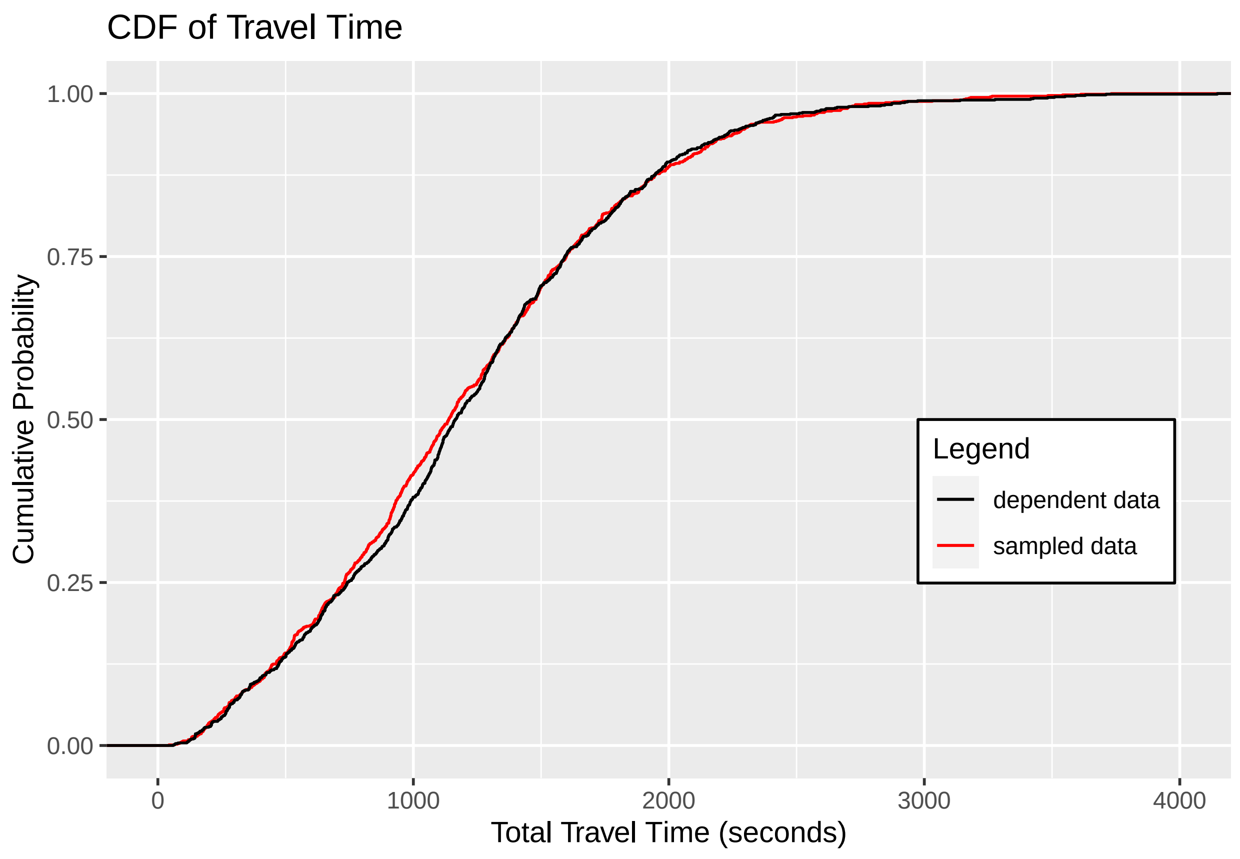


## The Full Dependent Model

```
dependent_uniform<-function(n, rho=0.31) {
    S <-diag(n)
    for (i in 1:n) {
        for (j in 2:n) {
            S[i, j] <- rho^(abs(i-j))
    }
}
    S = S +t(S)
    diag(S)<-1
    St = 2 * sin(S * pi/6) # must be positive definite
    U = c(pnorm(rmvnorm(1, sigma = St)))
    U
```

```r
}
time_simulator <- function(edges,rho=0.31){
  l <- length(edges)
  if(l>1)U <- dependent_uniform(l, rho)
  else U<-runif(1)
  mu <- (timebin_x_edge_sorted[match(edges,timebin_x_edge_sorted$timebin_x_edge_continuous), 4])
  sigma <- (timebin_x_edge_sorted[match(edges,timebin_x_edge_sorted$timebin_x_edge_continuous), 5])
  sum(exp(mu + sigma * qnorm(U)))
}
simulated_time <- sampled_trips%>%group_by(trip)%>%
  summarise(simulated_time=time_simulator(timebin_x_edge_continuous))
travel_time$dependent <- simulated_time$simulated_time
```

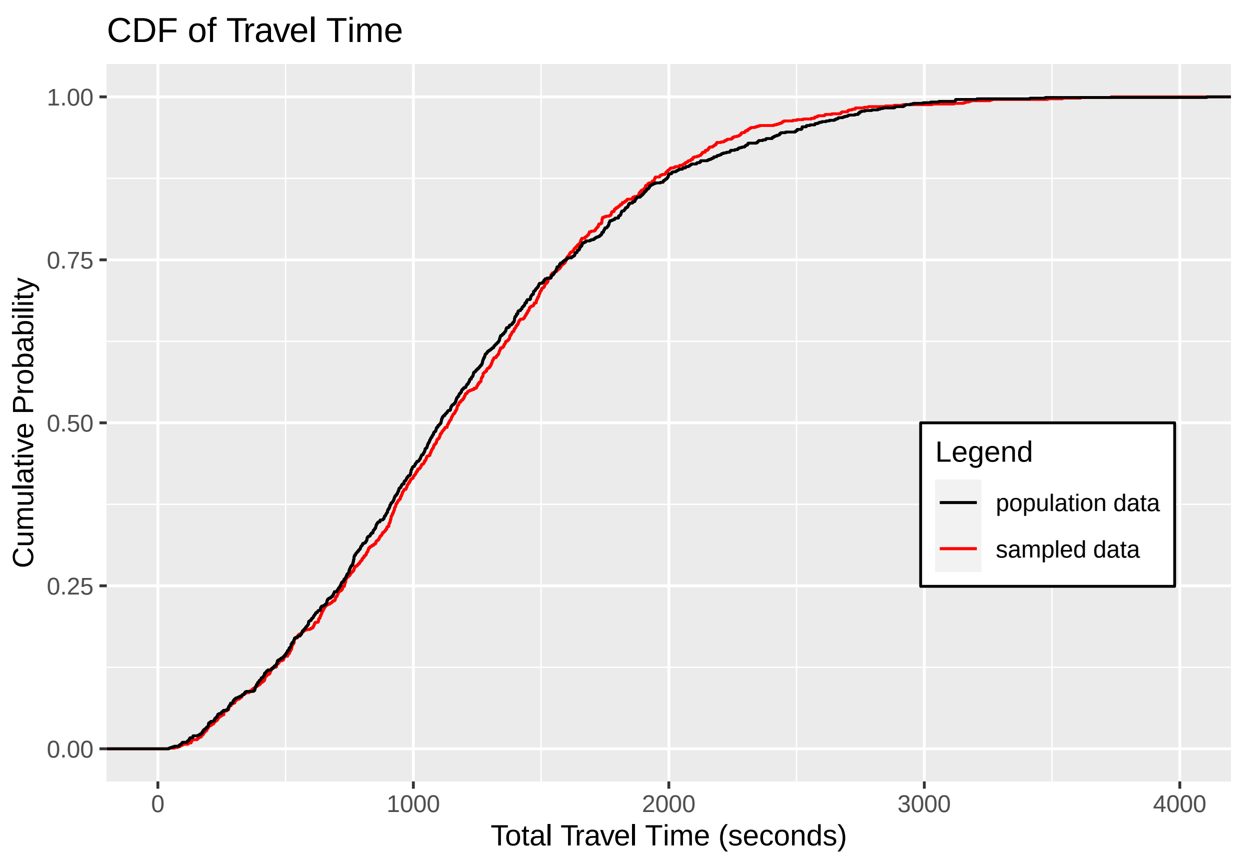
```r
plot2<-ggplot(travel_time) +
  stat_ecdf(aes(x = sampled_time,color="sampled data")) +
  stat_ecdf(aes(x = dependent,color="dependent data")) +
  labs(title = "CDF of Travel Time", x = "Total Travel Time (seconds)", y = "Cumulative Probability")+
  coord_cartesian(xlim = c(0, 4000), ylim = c(0, 1))+
  scale_color_manual(name="Legend",values = c("black","red"))+
    theme(legend.position = c(0.95, 0.5),
      legend.justification = c(1, 1),
      legend.text.align = 0,
      legend.background = element_rect(color = "black", fill = "white"))
plot2
```



Population model

```r
population_simulator <- function(duration_secs){
  l <- length(duration_secs)+1
  Z <- rnorm(l , 0, sqrt(1))
  mu <- mean(duration_secs)
  sigma <- sd_na_is_0(duration_secs)
  t = numeric(l)
   for(i in 2:l) {
    t[i] = t[i-1] + mu  + sigma * Z[i]
  }
  t[l]
}
population_time <- sampled_trips%>%group_by(trip)%>%
  summarise(simulated_time=population_simulator(duration_secs))
travel_time$population_time <- population_time$simulated_time
```

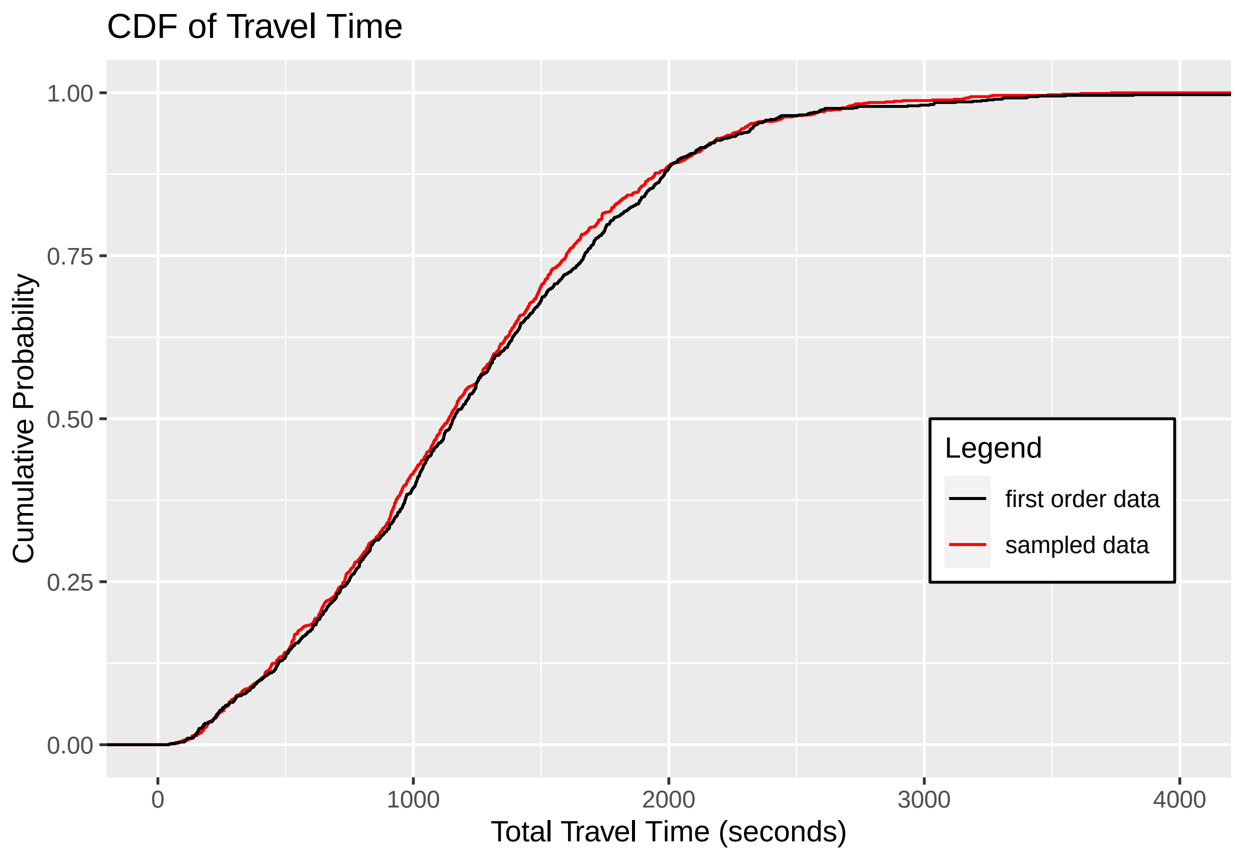
```r
plot3<-ggplot(travel_time) +
  stat_ecdf(aes(x = sampled_time,color="sampled data")) +
  stat_ecdf(aes(x = population_time,color="population data")) +
  labs(title = "CDF of Travel Time", x = "Total Travel Time (seconds)", y = "Cumulative Probability")+
  coord_cartesian(xlim = c(0, 4000), ylim = c(0, 1))+
  scale_color_manual(name="Legend",values = c("black","red"))+
    theme(legend.position = c(0.95, 0.5),
      legend.justification = c(1, 1),
      legend.text.align = 0,
      legend.background = element_rect(color = "black", fill = "white"))
plot3
```



## First Order Model

```
first_order<-function(n, rho=0.31) {
    S <-diag(n)
    if(n>1){
      if(n>2)for (i in 2:(n-1)) {
        for (j in (i-1):(i+1)) {
            S[i, j] <- 2*rho^(abs(1))
    }
      }
    S[n,n-1]=2*rho
    S[1, 2]=rho
    S[2, 1]=rho
    diag(S)<-1
    eigen_values <- eigen(S, symmetric = TRUE)$values
    if(!all(eigen_values >= 0))
    S <- as.matrix(Matrix::nearPD(S, cor = TRUE)$mat)
    U = c(pnorm(rmvnorm(1, sigma = S)))
    }else U = runif(1)
    U
}
first_order_simulator <- function(edges,rho=0.31){
  l <- length(edges)
  U <- first_order(l)
  mu <- (timebin_x_edge_sorted[match(edges,timebin_x_edge_sorted$timebin_x_edge_continuous), 4])
  sigma <- (timebin_x_edge_sorted[match(edges,timebin_x_edge_sorted$timebin_x_edge_continuous), 5])
  sum(exp(mu + sigma * qnorm(U)))
}
first_order_time <- sampled_trips%>%group_by(trip)%>%
  summarise(simulated_time=first_order_simulator(timebin_x_edge_continuous))
travel_time$first_order_time <- first_order_time$simulated_time
```

```
plot4<-ggplot(travel_time) +
  stat_ecdf(aes(x = sampled_time,color="sampled data")) +
  stat_ecdf(aes(x = first_order_time,color="first order data")) +
  labs(title = "CDF of Travel Time", x = "Total Travel Time (seconds)", y = "Cumulative Probability")+
  coord_cartesian(xlim = c(0, 4000), ylim = c(0, 1))+
  scale_color_manual(name="Legend",values = c("black","red"))+
    theme(legend.position = c(0.95, 0.5),
      legend.justification = c(1, 1),
      legend.text.align = 0,
      legend.background = element_rect(color = "black", fill = "white"))
plot4
```

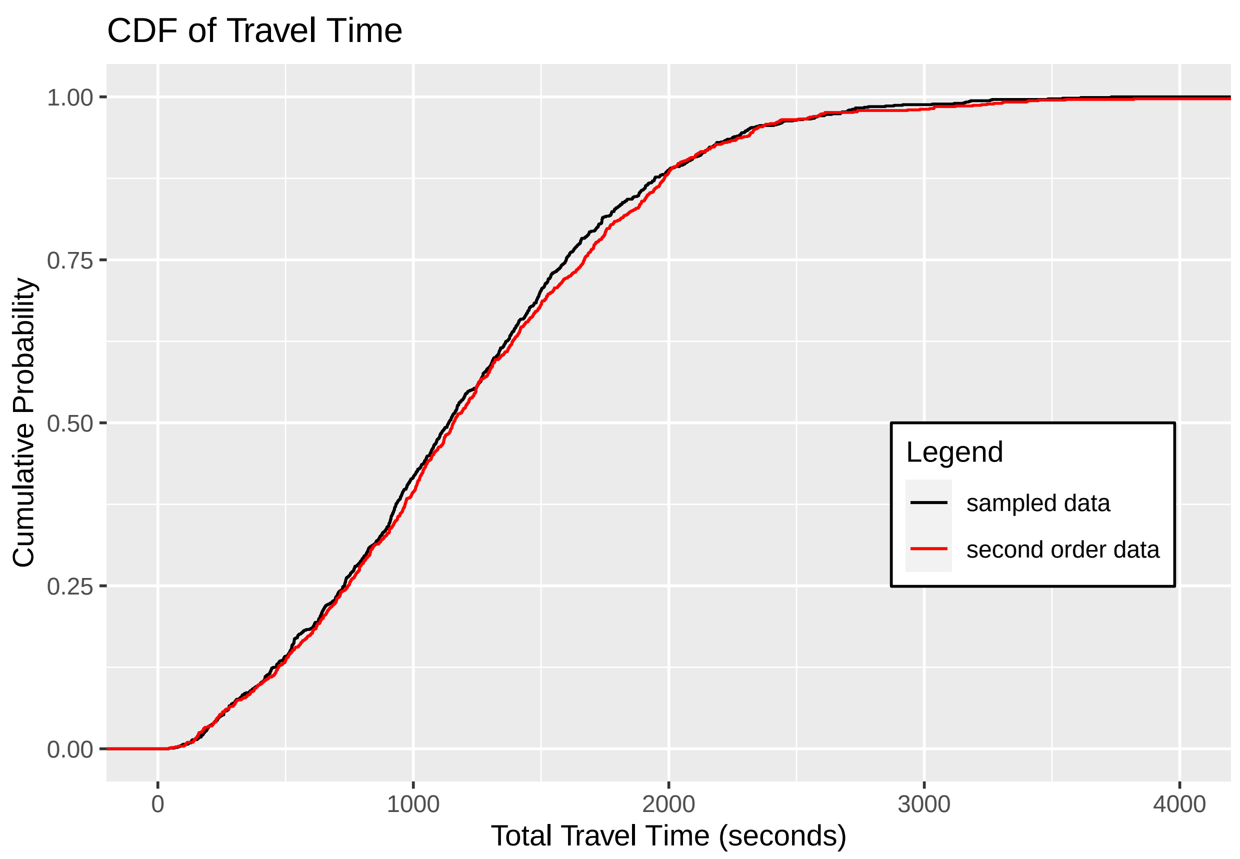


## Second Order Model

```
second_order<-function(n, rho=0.31) {
    S <-diag(n)
    if(n>2){
      for (i in 1:n) {
            if(i-2>0)S[i, (i-2)] <- 2*rho
            if(i+2<=n)S[i, (i+2)] <- 2*rho
      }
```

```r
      S[1, 3]=rho
      S[3, 1]=rho
      diag(S)<-1
      eigen_values <- eigen(S, symmetric = TRUE)$values
      if(!all(eigen_values >= 0))
      S <- as.matrix(Matrix::nearPD(S, cor = TRUE)$mat)
      U = c(pnorm(rmvnorm(1, sigma = S)))
    }else U = runif(n)
    U
}
second_order_simulator <- function(edges,rho=0.31){
  l <- length(edges)
  U <- second_order(l)
  mu <- (timebin_x_edge_sorted[match(edges,timebin_x_edge_sorted$timebin_x_edge_continuous), 4])
  sigma <- (timebin_x_edge_sorted[match(edges,timebin_x_edge_sorted$timebin_x_edge_continuous), 5])
  sum(exp(mu + sigma * qnorm(U)))
}
second_order_time <- sampled_trips%>%group_by(trip)%>%
  summarise(simulated_time=second_order_simulator(timebin_x_edge_continuous))
travel_time$second_order_time <- first_order_time$simulated_time
```

```r
plot5<-ggplot(travel_time) +
  stat_ecdf(aes(x = sampled_time,color="sampled data")) +
  stat_ecdf(aes(x = second_order_time,color="second order data")) +
  labs(title = "CDF of Travel Time", x = "Total Travel Time (seconds)", y = "Cumulative Probability")+
  coord_cartesian(xlim = c(0, 4000), ylim = c(0, 1))+
  scale_color_manual(name="Legend",values = c("black","red"))+
    theme(legend.position = c(0.95, 0.5),
      legend.justification = c(1, 1),
      legend.text.align = 0,
      legend.background = element_rect(color = "black", fill = "white"))
plot5
```



```
tripdata <- data.frame(logspeed = sampled_trips$logspeed,
                      tripID = sampled_trips$trip,
                      timeBin = sampled_trips$timeBins,
                      linkID = sampled_trips$linkID,
                      length = sampled_trips$length,
                      time = (sampled_trips$time),
                      traveltime = sampled_trips$duration_secs)
tripdata <- tripdata %>% group_by(tripID) %>%
  arrange(time, .by_group = TRUE)
unique(tripdata$timeBin)
```

```
## [1] "Other" "MR"    "ER"
```

```
tripdata$time <- as.POSIXct( tripdata$time, format = "%Y-%m-%dT%H:%M:%OSZ")
tripdata$timeBin<-time_bins_readable(tripdata$time)
unique(tripdata$timeBin)
```

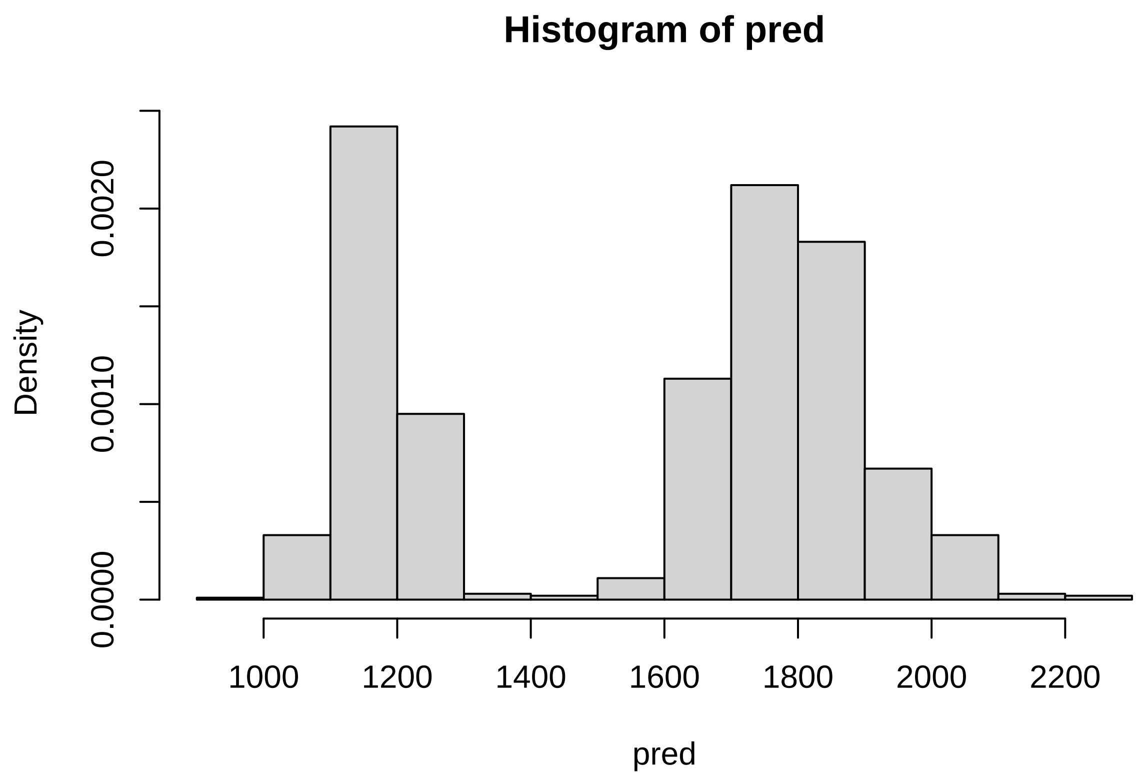
```
## [1] "EveningNight" "Weekday"      "MorningRush"  "EveningRush"  "Weekendday"
```

```
fit <- traveltimeHMM(data = tripdata,
                     nQ = 2,max.it = 20, model = "HMM")
```

```
## max.speed is not specified, setting at default value: 130 km/h
## Warning: Many observations are higher than speed limit (130km/h)!, about 0.11%
## of observations.  It is advised to remove these observations.

## Model HMM with 1000 trips over 16212 roads and 5 time bins...

## Expected completion of 20 iterations in 68.3 secs

## Reached maximum number of iterations
```

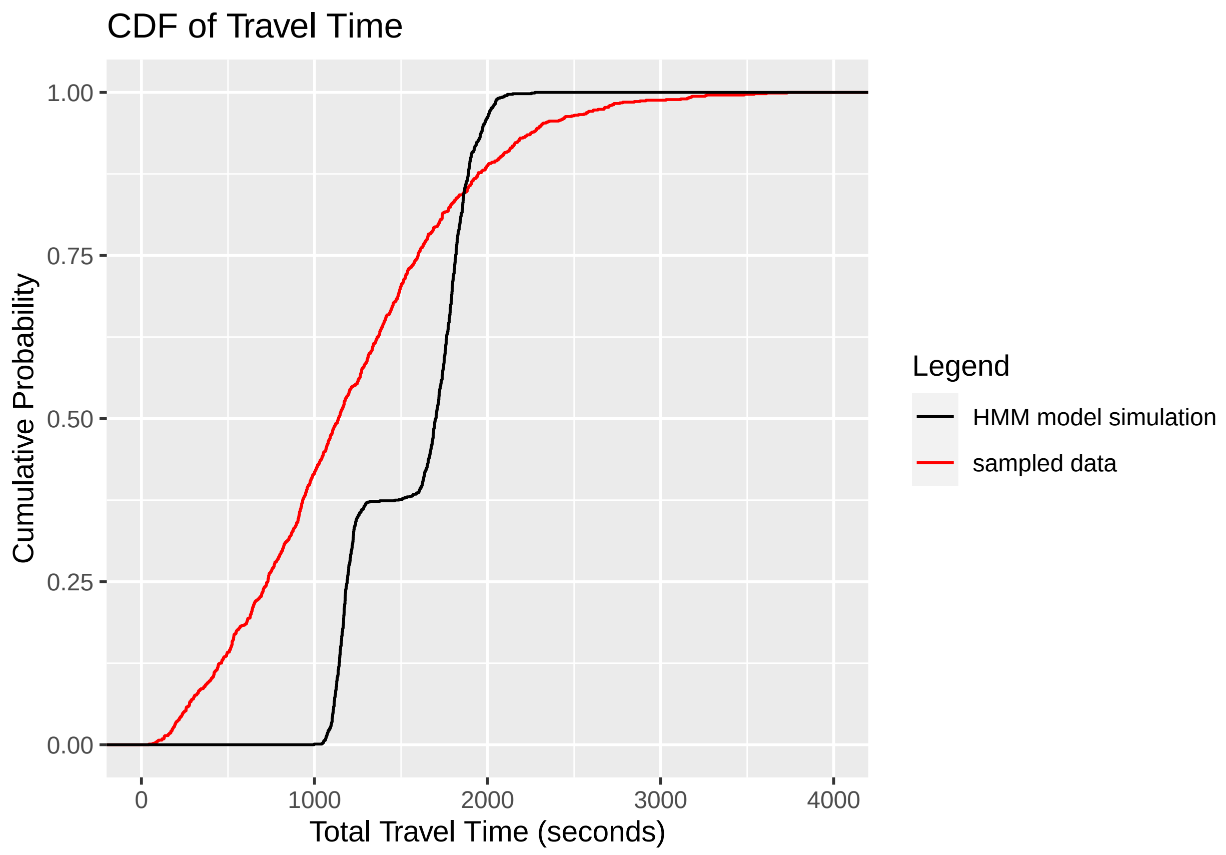
```
single_trip <- subset(tripdata, tripID==unique(tripdata$tripID)[1])
```

```
pred <- predict(object = fit,
              tripdata = single_trip,
              starttime = single_trip$time[1],
              n = 1000)
hist(pred, freq = FALSE)
```



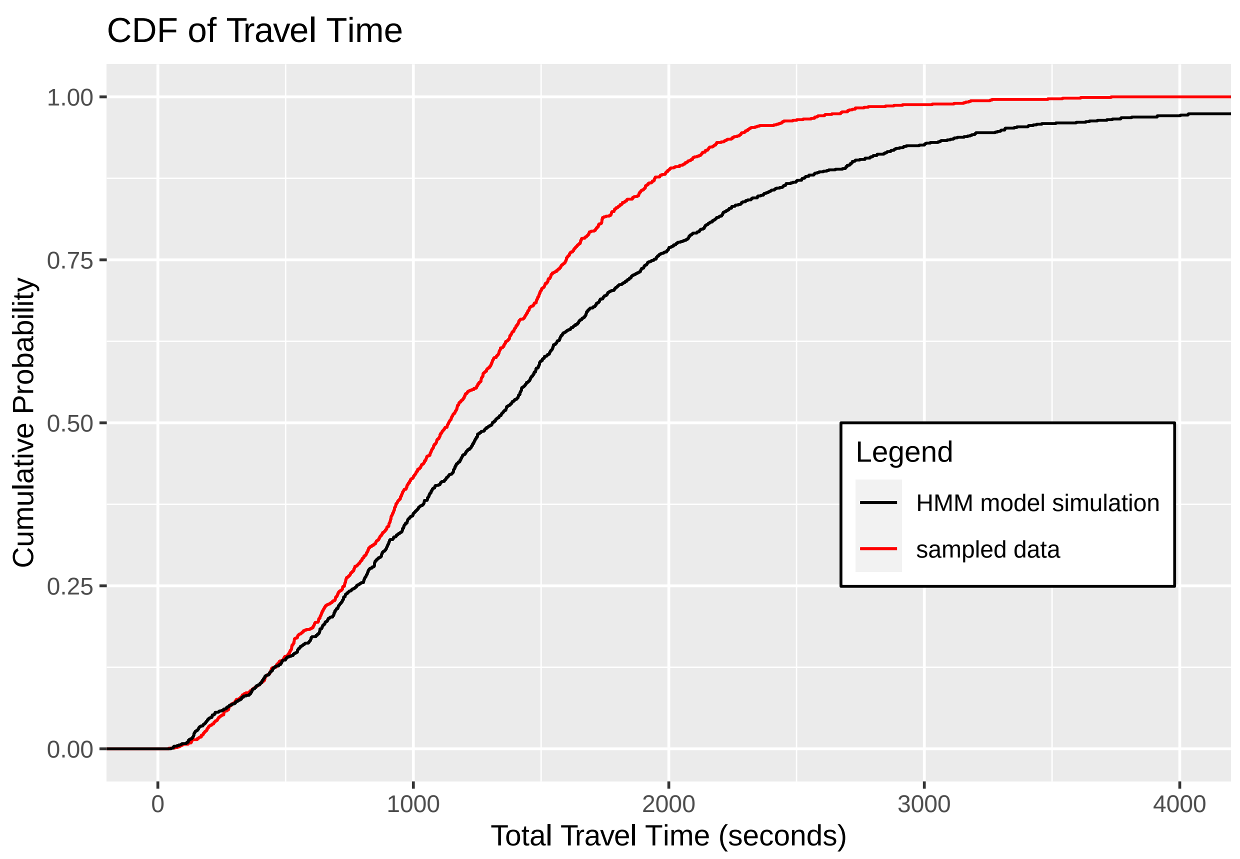
```
travel_time$HMM <- pred
```

```
ggplot(travel_time) +
  stat_ecdf(aes(x = sampled_time,color="sampled data")) +
  stat_ecdf(aes(x = pred,color="HMM model simulation")) +
  labs(title = "CDF of Travel Time", x = "Total Travel Time (seconds)", y = "Cumulative Probability")+
  coord_cartesian(xlim = c(0, 4000), ylim = c(0, 1))+
  scale_color_manual(name="Legend",values = c("black","red"))
```



```r
starttime <- tripdata%>%group_by(tripID)%>%summarise(first(time))
pred <- c()
for (i in 1:1000) {
  single_trip <- subset(tripdata, tripID==unique(tripdata$tripID)[i])
  newpred <- predict(object = fit,
                tripdata = single_trip,
                starttime = starttime$`first(time)`[i],
                n = 1)
  pred<-c(pred,newpred)
}
travel_time$HMM <- pred
```

```r
plot6<-ggplot(travel_time) +
  stat_ecdf(aes(x = sampled_time,color="sampled data")) +
  stat_ecdf(aes(x = pred,color="HMM model simulation")) +
  labs(title = "CDF of Travel Time", x = "Total Travel Time (seconds)", y = "Cumulative Probability")+
  coord_cartesian(xlim = c(0, 4000), ylim = c(0, 1))+
  scale_color_manual(name="Legend",values = c("black","red"))+
    theme(legend.position = c(0.95, 0.5),
      legend.justification = c(1, 1),
      legend.text.align = 0,
      legend.background = element_rect(color = "black", fill = "white"))
plot6
```



```r
plot_list <- list(plot1,plot2,plot3,plot4,plot5,plot6)
ggsave("plot/R_combined_simulation_plots.jpg",
       plot = wrap_plots(plot_list, nrow=3),
       device = "jpg",
       width = 21,
       height = 29.7,
       units = "cm",
       dpi = 600)
```