HW3

113078506

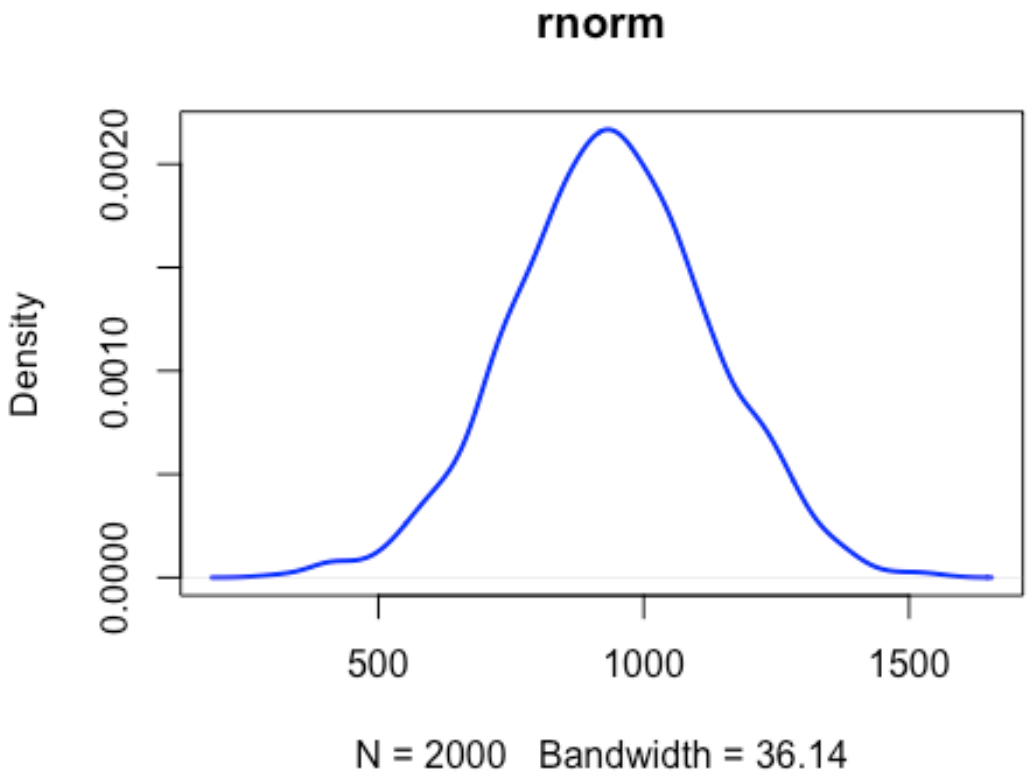
2025-03-06

- Gen AI used: I used Gen AI to make code more “clean”. I also used it to refine English explanation.

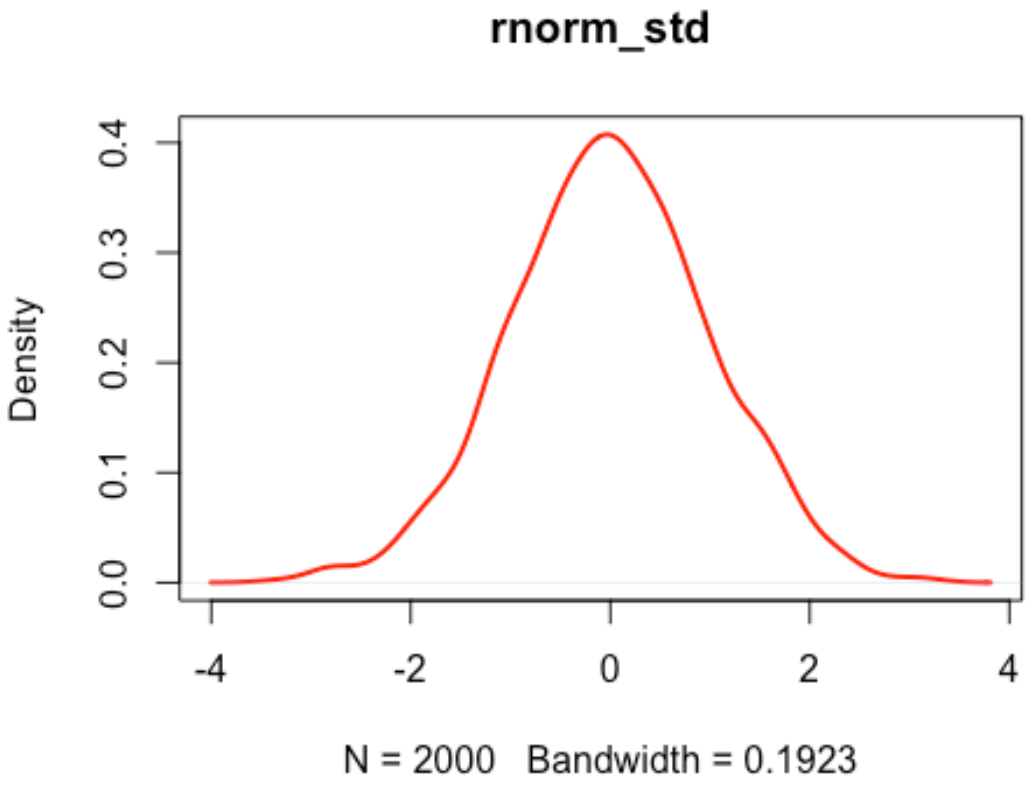
Question 1) Let’s reexamine how to standardize data: subtract the mean of a vector from all its values, and divide this difference by the standard deviation to get a vector of standardized values.

(a) Create a normal distribution (mean=940, sd=190) and standardize it (let’s call it rnorm_std)

```
set.seed(1234)
rnorm_1 <- rnorm(2000, mean = 940, sd = 190)
std <- function(vec) {
  vec <- (vec - mean(vec)) / sd(vec)
  return(vec)
}

rnorm_std <- std(rnorm_1)
mean(rnorm_std)

## [1] 2.591261e-16

sd(rnorm_std)

## [1] 1

plot(density(rnorm_1), main="rnorm", col="blue", lwd=2)
```




```r
plot(density(rnorm_std), main="rnorm_std", col="red", lwd=2)
```




```
rm(rnorm)

## Warning in rm(rnorm): object 'rnorm' not found
```

i) What should we expect the mean and standard deviation of rnorm_std to be, and why?

- The mean should be approximately 0, and the standard deviation should be approximately 1 because each value in the standardized dataset is obtained by subtracting the mean and dividing by the standard deviation. Essentially, the standardized dataset represents the distance of each data point from the mean, measured in units of standard deviation.

ii) What should the distribution (shape) of rnorm_std look like, and why?

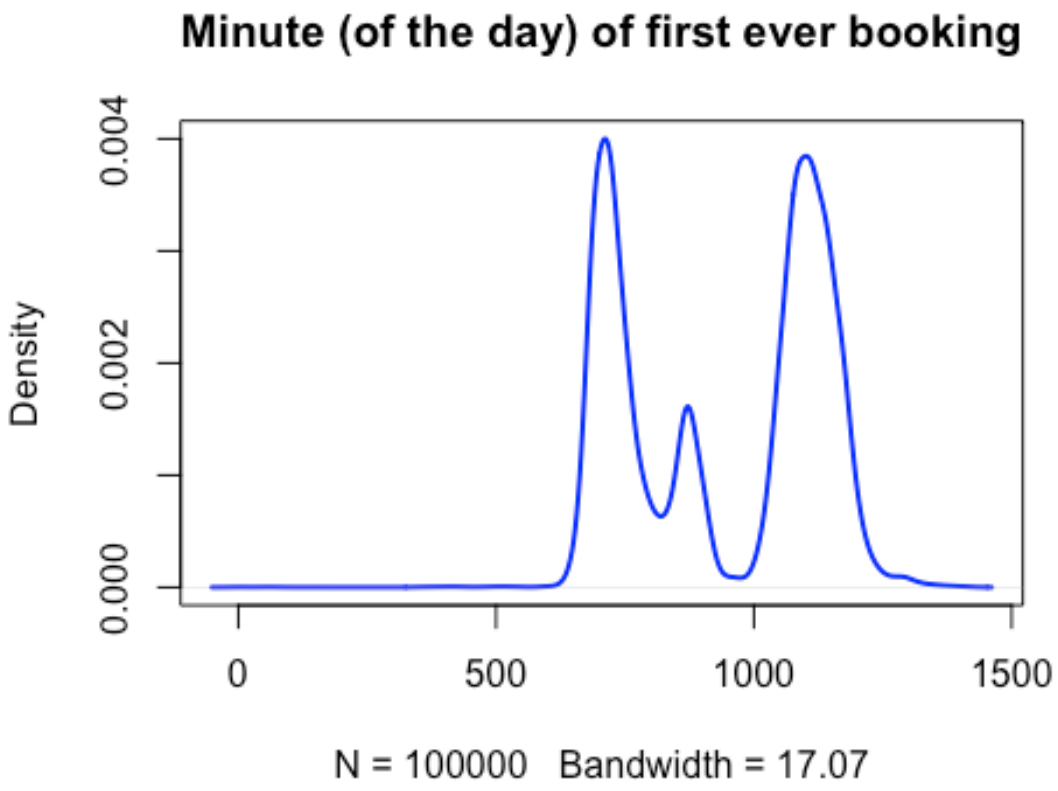
- The shape of rnorm_std should be normal distribution as the plots shown because by standardizing we only rescale the dataset, didn't change the distribution.

iii) What do we generally call distributions that are normal and standardized?

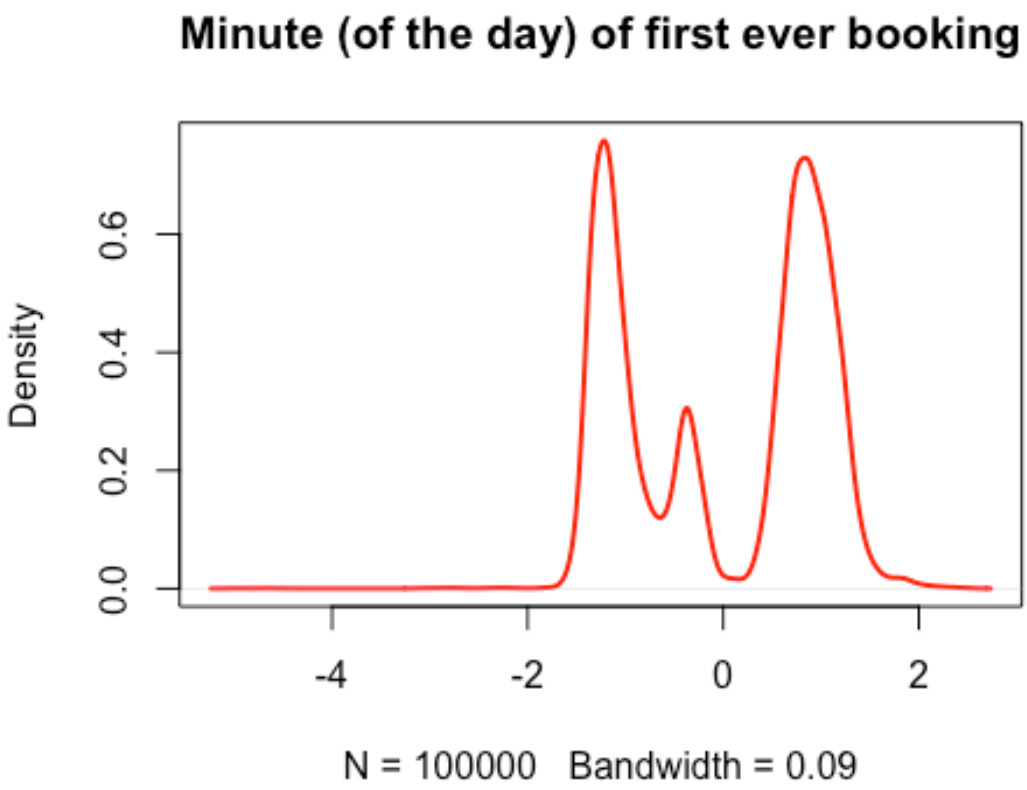
- We generally call it z-distribution.

(b) Create a standardized version of minday discussed in question 3 (let's call it minday_std)

```
# Code in Q3
bookings <- read.table("first_bookings_datetime_sample.txt", header=TRU
E)
hours  <- as.POSIXlt(bookings$datetime, format="%m/%d/%Y %H:%M")$hour
mins   <- as.POSIXlt(bookings$datetime, format="%m/%d/%Y %H:%M")$min
minday <- hours*60 + mins
plot(density(minday), main="Minute (of the day) of first ever booking",
 col="blue", lwd=2)
```




```
#std ver of minday
min_std <- (minday - mean(minday)) / sd(minday)
mean(min_std)

## [1] 2.302986e-15

sd(min_std)

## [1] 1

plot(density(min_std), main="Minute (of the day) of first ever booking",
 col="red", lwd=2)
```



i) What should we expect the mean and standard deviation of minday_std to be, and why?

- Although the data distribution is different from (a), the mean should still be approximately equals to 0 and the sd should still be qpproximately equals to 1. As previous mentioned, the standardization actually represent the distance between each data point and mean, measured in units of standard deviation.

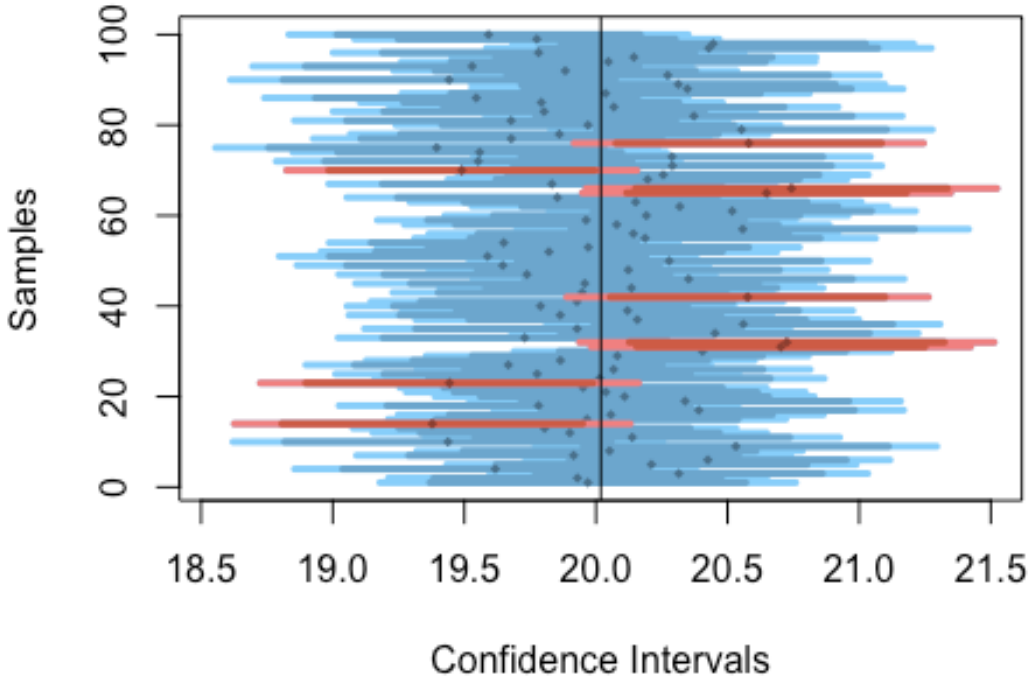
ii) What should the distribution of minday_std look like compared to minday, and why?

- The distribution of minday_std should look the same as minday, as the plots shown. The standardization only rescale the data, it cannot change data distribution. The standardized version of data distribution will remain the same charachteristics with original data distribution.

Question 2) Install the “compstatslib” package from Github (see week 2 class handout) and run the plot_sample_ci() function that simulates samples drawn randomly from a population. Each sample is a horizontal line with a dark band for its 95% CI, and a lighter band for its 99% CI, and a dot for its mean. The population mean is a vertical black line. Samples whose 95% CI includes the population mean are blue, and others are red.

(a) Simulate 100 samples (each of size 100), from a normally distributed population of 10,000:

```
#install.packages("remotes")
#remotes::install_github("soumyaray/compstatslib", force = TRUE)

library("compstatslib")
set.seed(1234)
plot_sample_ci(num_samples = 100, sample_size = 100, pop_size=10000,
               distr_func= rnorm, mean = 20, sd = 3)
```



i) How many samples do we expect to NOT include the population mean in its 95% CI?

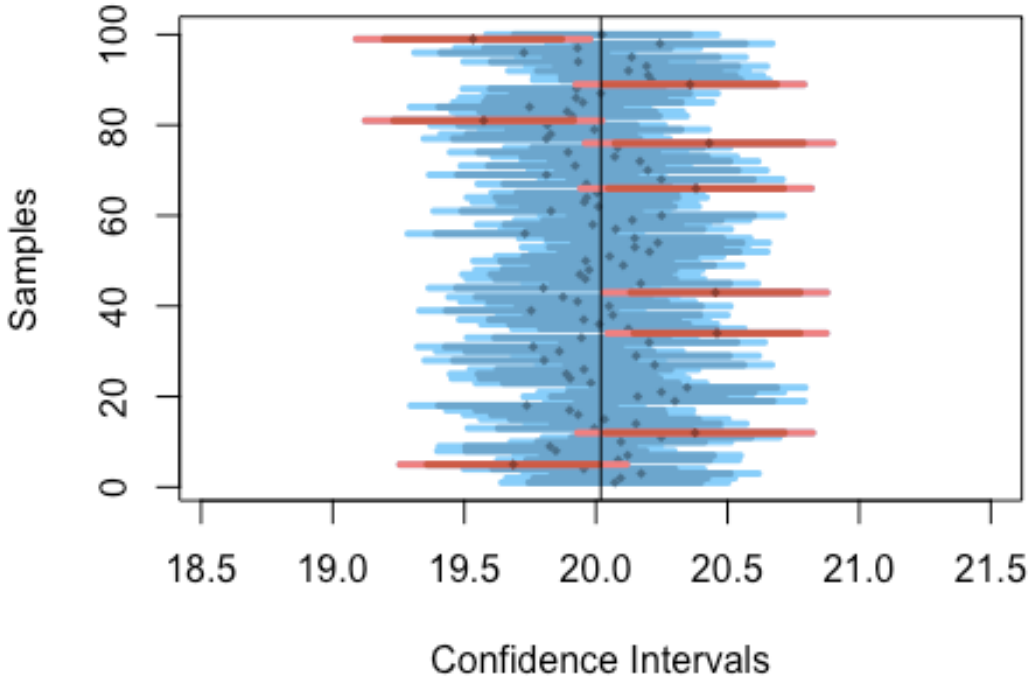
- There should be 5 samples not include in the population mean in it's 95% CI according to 100 - (100 samples*95%) = 5. But in my simulation there are 8 red lines, according to Teams discussion which I think it make sense: maybe the results are different on multiple trials.

ii) How many samples do we expect to NOT include the population mean in their 99% CI?

- There should be 1 samples not include in the population mean in it's 99% CI according to 100 - (100 samples*99%) = 1. In this simulation, all the red line reach the population mean, so there are no sample not include the population mean in 99% CI.

(b) Rerun the previous simulation with the same number of samples, but larger sample size (sample_size=300):

```
set.seed(1234)
plot_sample_ci(num_samples = 100, sample_size = 300, pop_size=10000,
               distr_func= rnorm, mean = 20, sd = 3)
```



i) Now that the size of each sample has increased, do we expect their 95% and 99% CI to become wider or narrower than before?

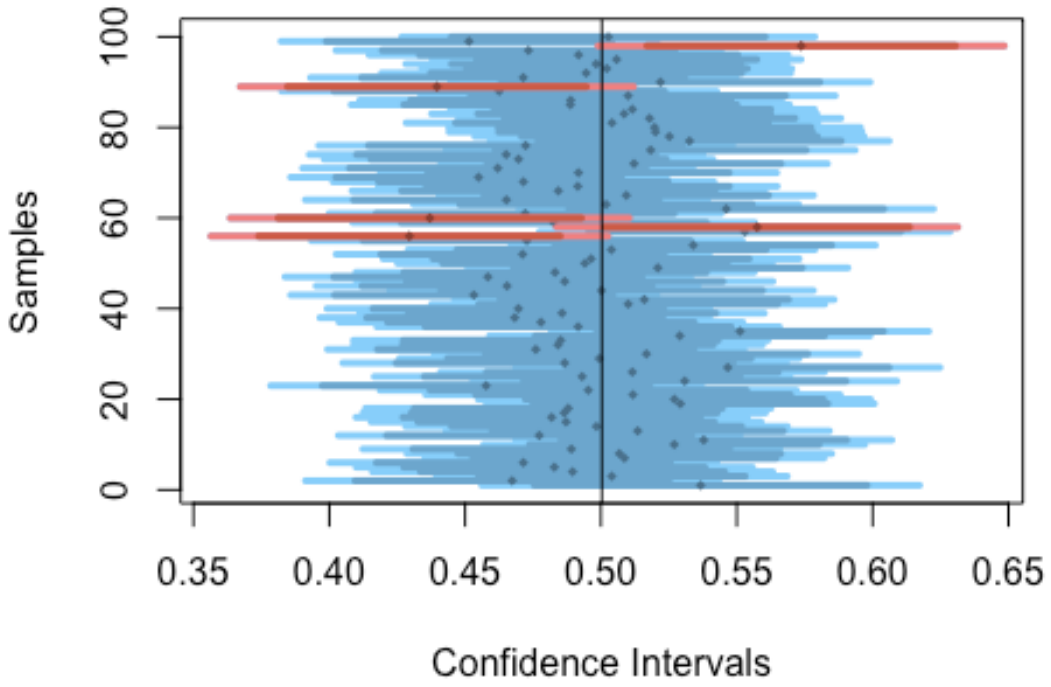
- The 95% and 99% CI seems to be narrower after we increased the sample size, it is because when we increase sample size, the standard error decrease: SE = s/n^0.5. The bigger n is, the smaller SE is.

ii) This time, how many samples (out of the 100) would we expect to NOT include the population mean in its 95% CI?

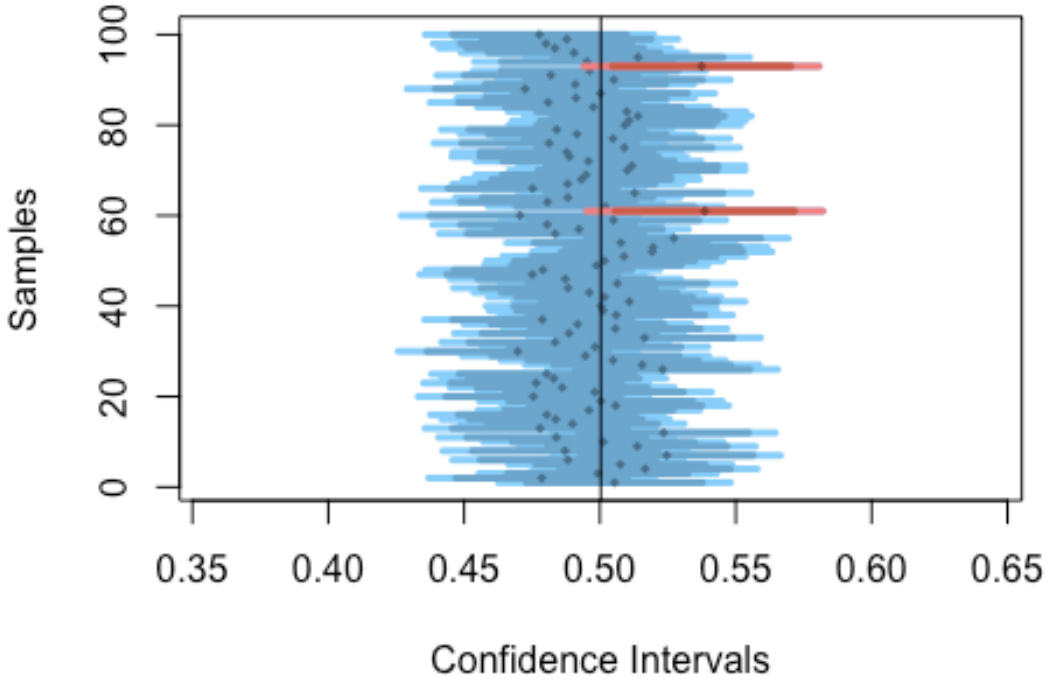
- The expected samples amount is calculated: 100 - (100 samples*95%) = 5, which should be consistent even when sample size become larger. In my simulation, there are 9 samples that not include the population mean in it's 95 % CI.

c) If we ran the above two examples (a and b) using a uniformly distributed population (specify parameter distr_func=runif for plot_sample_ci), how do you expect your answers to (a) and (b) to change, and why?

```
set.seed(1234)
plot_sample_ci(num_samples = 100, sample_size = 100, pop_size=10000,
               distr_func= runif)
```




```
set.seed(1234)
plot_sample_ci(num_samples = 100, sample_size = 300, pop_size=10000,
               distr_func= runif)
```

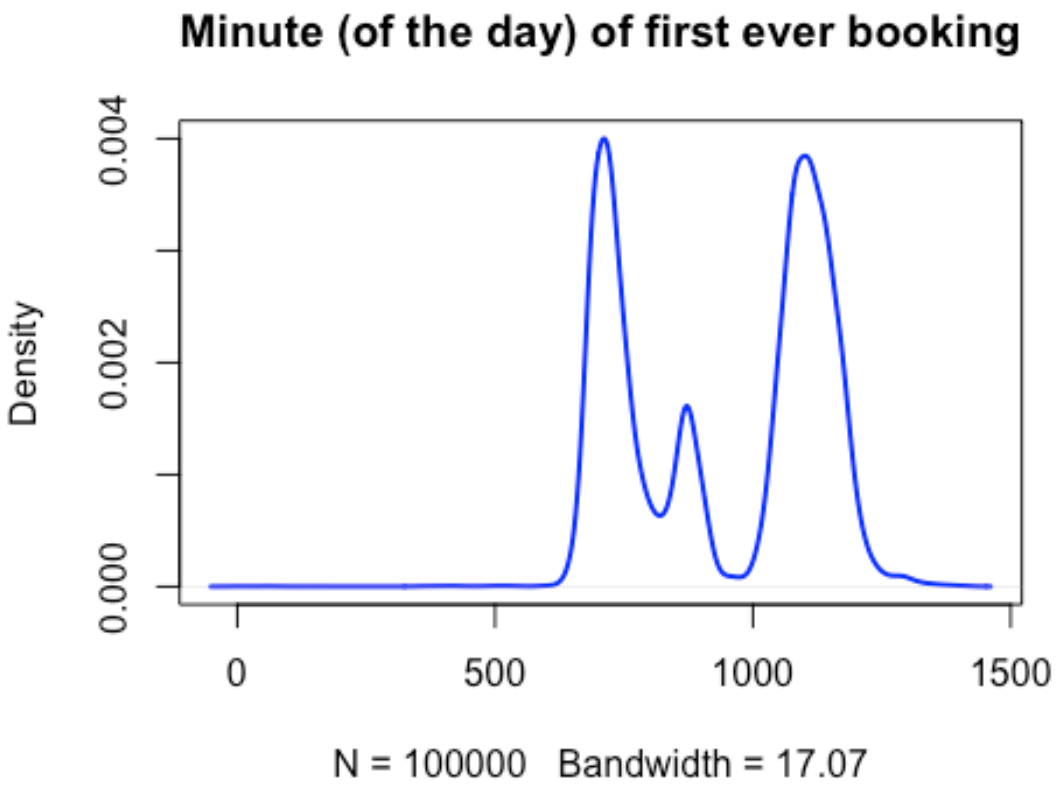


- The results in (a) and (b) will be the same when the distribution change to uniform distribution, because the mean of uniform distribution is still equal to median, which will lay in the center, obey central limit theorem. Although the variance is different between this 2 distribution, the expected samples that not include population mean will be the same as normal distribution.

Question 3) The company EZTABLE has an online restaurant reservation platform that is accessible by mobile and web. Imagine that EZTABLE would like to start a promotion for new members to make their bookings earlier in the day. We have a sample of data about their new members, in particular the date and time for which they make their first ever booking (i.e., the booked time for the restaurant) using the EZTABLE platform. Here is some sample code to explore the data:

(a) What is the “average” booking time for new members making their first restaurant booking?(use minday, which is the absolute minute of the day from 0-1440)

```
bookings <- read.table("first_bookings_datetime_sample.txt", header=TRU
E)
hours  <- as.POSIXlt(bookings$datetime, format="%m/%d/%Y %H:%M")$hour
mins   <- as.POSIXlt(bookings$datetime, format="%m/%d/%Y %H:%M")$min
minday <- hours*60 + mins
plot(density(minday), main="Minute (of the day) of first ever booking",
 col="blue", lwd=2)
```



i) Use traditional statistical methods to estimate the population mean of minday, its standard error, and the 95% confidence interval (CI) of the sampling means

```
mean(minday)
```

```
## [1] 942.4963
```

```
se = sd(minday)/sqrt(length(minday))

ci_left = mean(minday)- sd(minday)/sqrt(length(minday))
ci_right = mean(minday) + sd(minday)/sqrt(length(minday))

#95% CI
ci = c(ci_left, ci_right)
ci
```

```
## [1] 941.8966 943.0961
```

```
cat("mean:",mean(minday), "SE:", se, "95% Confident Interval:", ci)
```

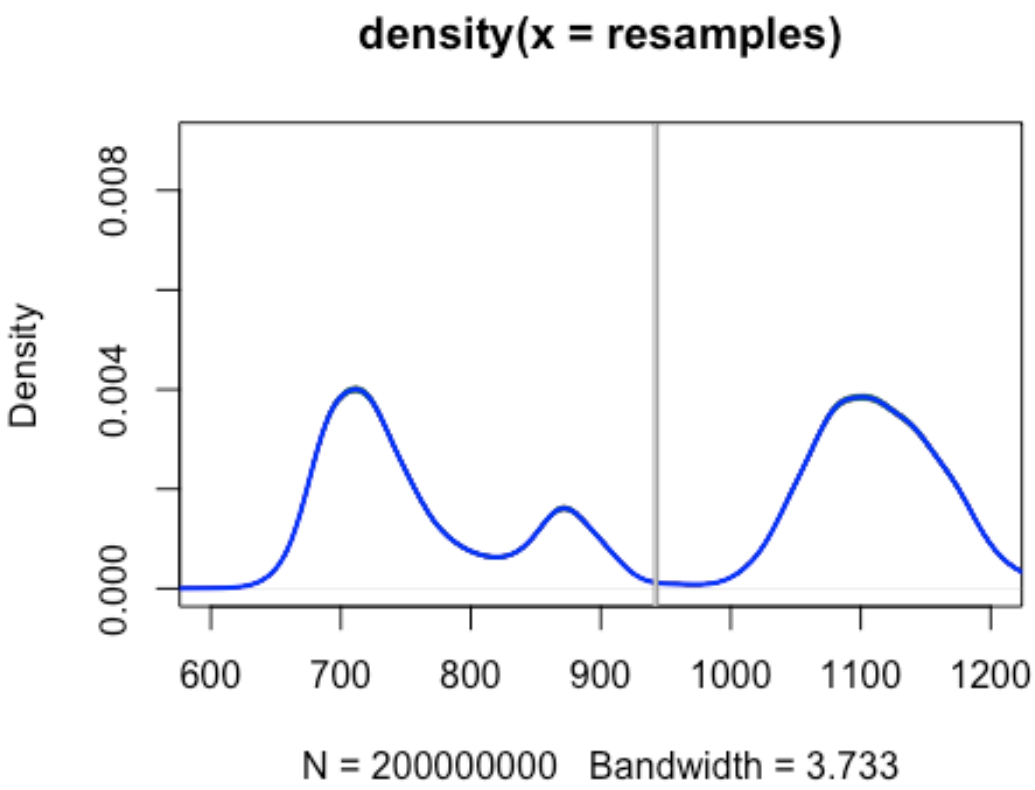
```
## mean: 942.4963 SE: 0.5997673 95% Confident Interval: 941.8966 943.09
61
```

ii) Bootstrap to produce 2000 new samples from the original sample

```
set.seed(1234)
resamples <- replicate(2000,
sample(minday, length(minday), replace=TRUE))
```

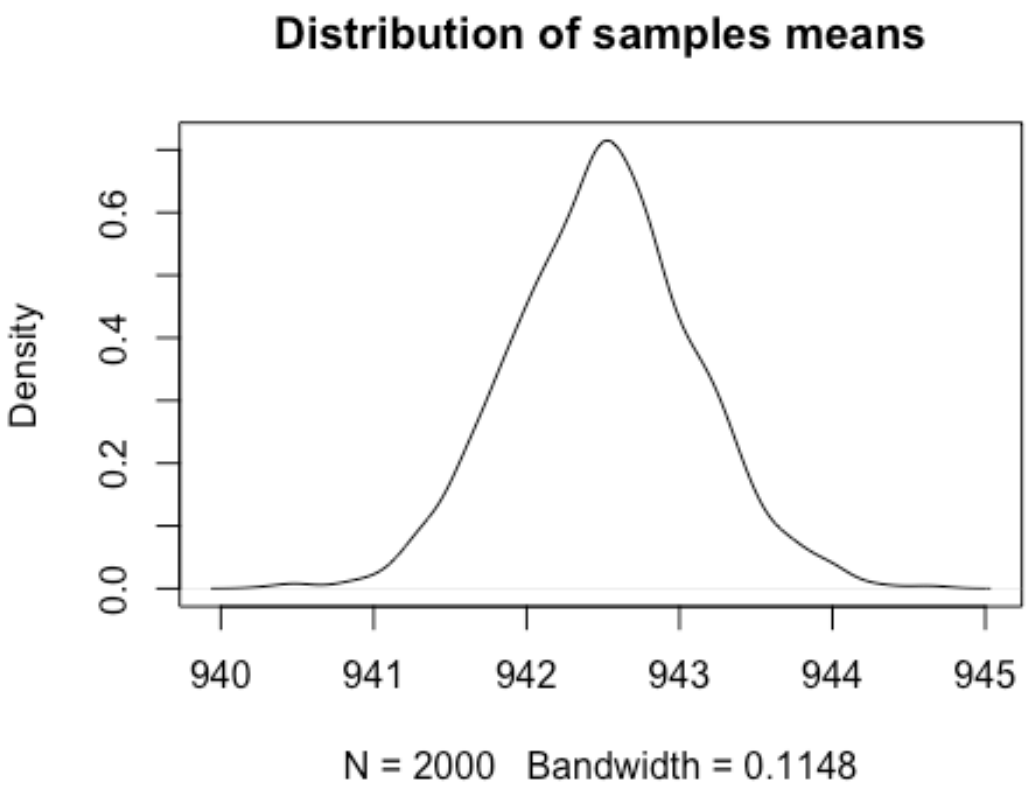
iii) Visualize the means of the 2000 bootstrapped samples

```
#Create an empty plotting space with axes
plot(density(resamples), lwd=0, ylim=c(0, 0.009), xlim=c(600,1200))

# A function to plot a single sample's distribution
plot_resample_density <- function(sample_i) {
lines(density(sample_i), col=rgb(0.0, 0.4, 0.0, 0.01))
return(mean(sample_i))
}

# Iteratively plot and get means of all bootstrapped samples
sample_means <- apply(resamples, 2, FUN=plot_resample_density)

# Plot original sample densities
lines(density(minday), col="blue", lwd=2)

# Draw light vertical lines for each sampling mean
abline(v=sample_means, col=rgb(0.7, 0.7, 0.7, 0.01))
```



- The light grey lines indicates that the sample means lay in the small range between approximately 940~950.

```
plot(density(sample_means), main="Distribution of samples means")
```



- The distribution of the sample means shows that the sample means range are from 940 to 945.

iv) Estimate the 95% CI of the bootstrapped means using the quantile function

```
#the 95% CI of the bootstrapped means
quantile(sample_means, probs=c(0.05, 0.95))
```

```
##       5%      95% 
## 941.5228 943.4768
```

b) By what time of day, have half the new members of the day already arrived at their restaurant?

i) Estimate the median of minday

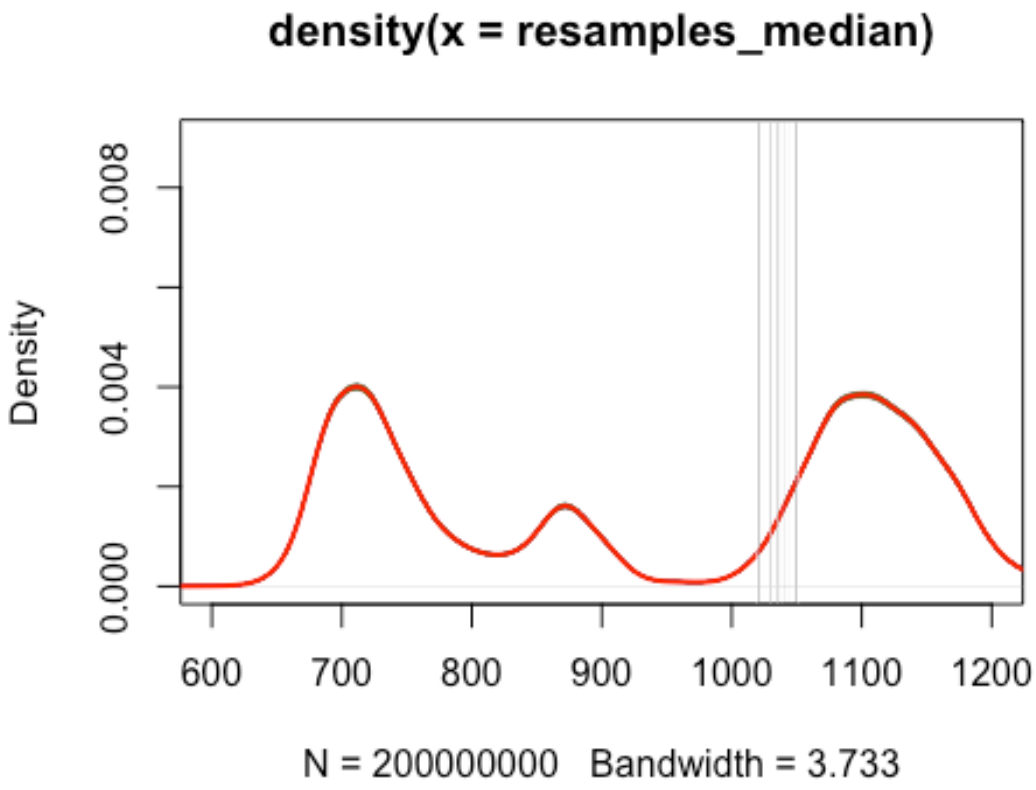
```
median(minday)
```

```
## [1] 1040
```

- The median is 1040 (minutes)

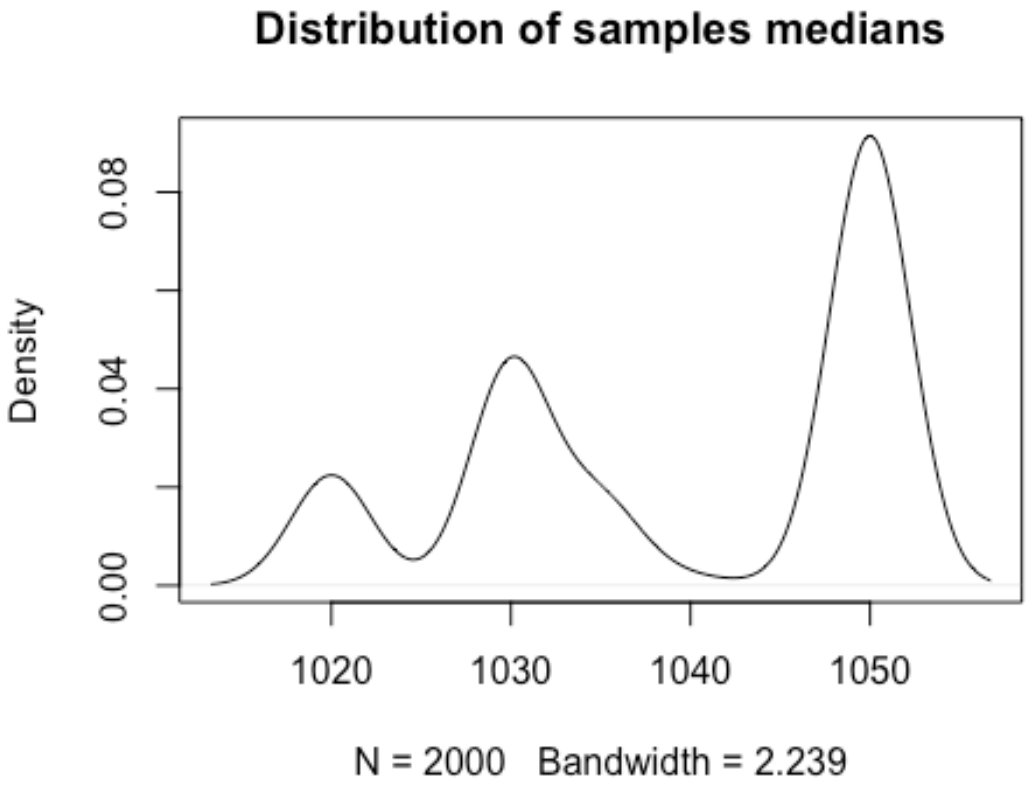
ii) Visualize the medians of the 2000 bootstrapped samples

```
set.seed(1234)
resamples_median <- replicate(2000,
sample(minday, length(minday), replace=TRUE))

plot(density(resamples_median), lwd=0, ylim=c(0, 0.009), xlim=c(600,120
0))

plot_resample_density_median <- function(sample_i) {
lines(density(sample_i), col=rgb(0.0, 0.4, 0.0, 0.01))
return(median(sample_i))
}

# Iteratively plot and get means of all bootstrapped samples
sample_medians <- apply(resamples_median, 2, FUN=plot_resample_density_
median)

# Plot original sample densities
lines(density(minday), col="red", lwd=2)

# Draw light vertical lines for each sampling mean
abline(v=sample_medians, col=rgb(0.7, 0.7, 0.7, 0.01))
```



- The medians are lat in the range of approximately 1010~1050.

```
plot(density(sample_medians), main="Distribution of samples medians")
```



- The distribution plot has shown that the sample medians are lying between the range 1010~1060 (minutes).

iii) Estimate the 95% CI of the bootstrapped medians using the quantile function

```
# the 95% CI of the bootstrapped medians
quantile(sample_medians, probs=c(0.05, 0.95))
```

```
##   5%  95%
## 1020 1050
```