## Student ID: 112077423

```
library(ggplot2)
library(dplyr)
library(compstatslib)
```

## Question 1(a)

*Create a normal distribution (mean=940, sd=190) and standardize it.*

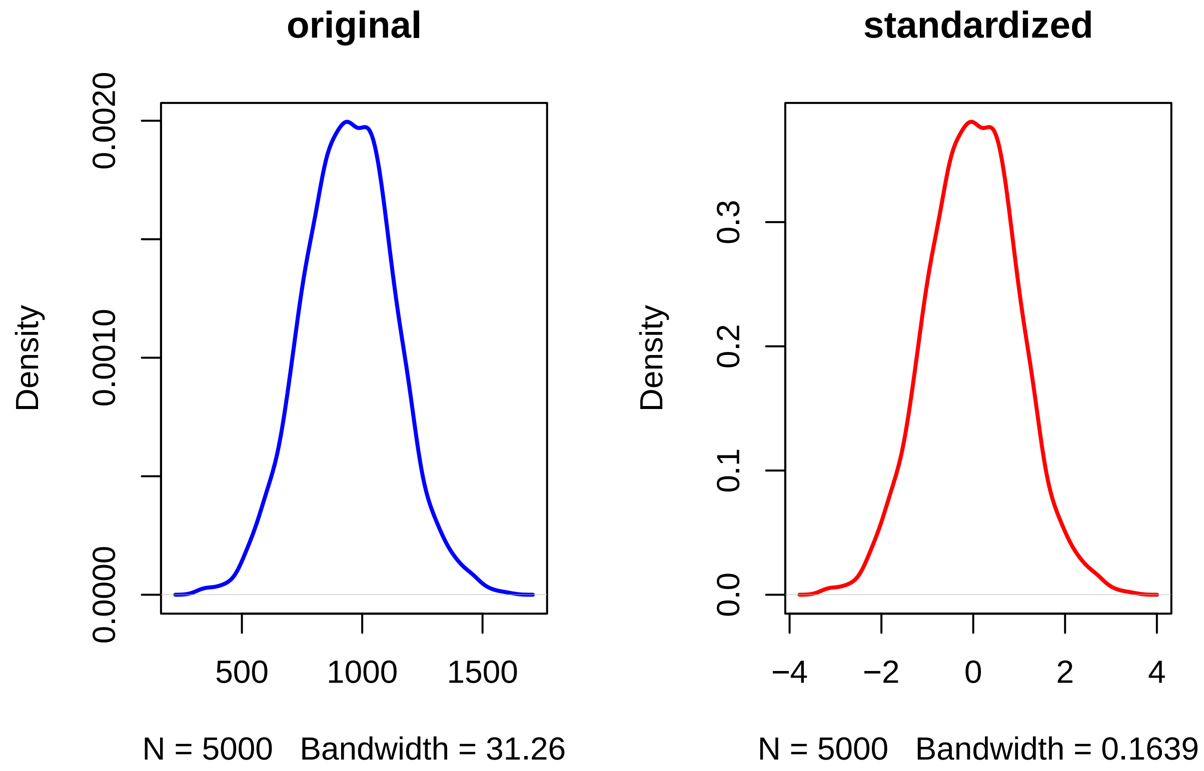
```
# function to standardize
standardize <- function(numbers) {
  numbers <- (numbers - mean(numbers)) / sd(numbers)
  return(numbers)
}

normal <- rnorm(5000, mean=940, sd=190)
rnorm_std <- standardize(normal)

print(paste('mean:', mean(rnorm_std), 'sd:', sd(rnorm_std)))
```

```
## [1] "mean: 1.89917114153115e-16 sd: 1"
```

```
par(mfrow=c(1,2))
plot(density(normal), main='original', col="blue", lwd=2)
plot(density(rnorm_std), main='standardized', col="red", lwd=2)
```



(i) We expect the mean and standard deviation of rnorm_std to be 0 and 1 respectively. By subtracting the original mean from each element of the vector, we are centering data around 0. And by dividing each element of the vector by sd, we are scaling the data so it has a sd of 1;

(ii) rnorm_std should be bell-shaped because standardization doesn't affect the original shape. Data is simply centered around 0 with a standard deviation of 1;

(iii) Standard Normal Distribution.

### Question 1(b)

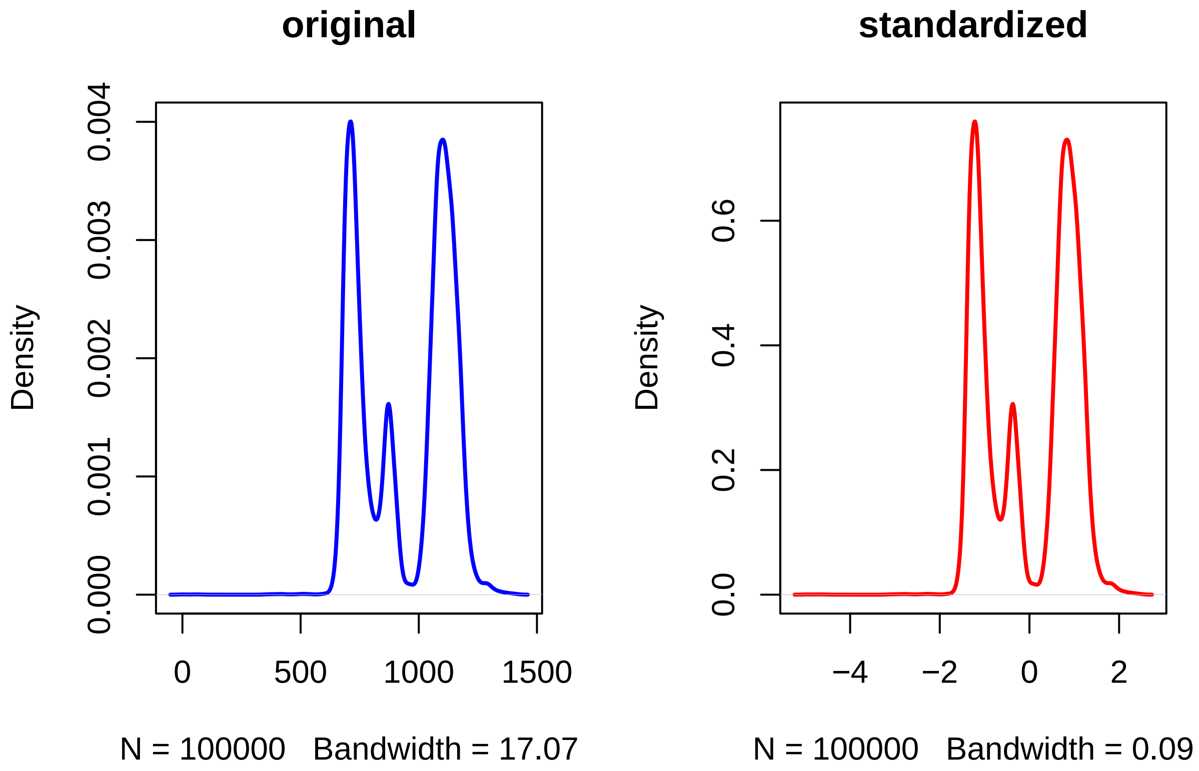
*Create a standardized version of minday.*

```
bookings <- read.table("first_bookings_datetime_sample.txt", header=TRUE)
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

minday_std <- standardize(minday)
print(paste('mean:', mean(minday_std), 'sd:', sd(minday_std)))
```

```
## [1] "mean: -4.25589034500073e-17 sd: 1"
```

```
par(mfrow=c(1,2))
plot(density(minday), main='original', col="blue", lwd=2)
plot(density(minday_std), main='standardized', col="red", lwd=2)
```



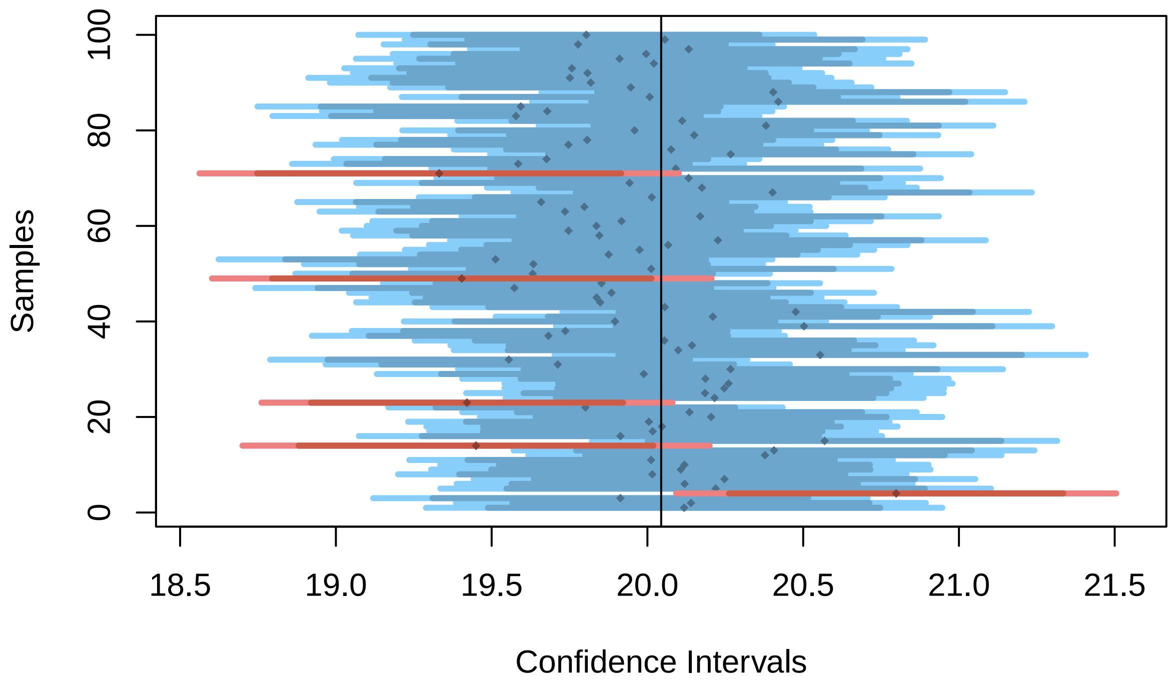
(i) We expect the mean and standard deviation of minday_std to be 0 and 1 respectively. By subtracting the original mean from each element of the vector, we are centering data around 0. And by dividing each element of the vector by sd, we are scaling the data so it has a sd of 1;

(ii) minday_std should have the same distribution as minday because standardization doesn't affect the original shape.

### Question 2(a)

*Simulate 100 samples (each of size 100), from a normally distributed population of 10000.*

```
plot_sample_ci(num_samples = 100,
               sample_size = 100,
               pop_size=10000,
               distr_func=rnorm,
               mean=20,
               sd=3)
```



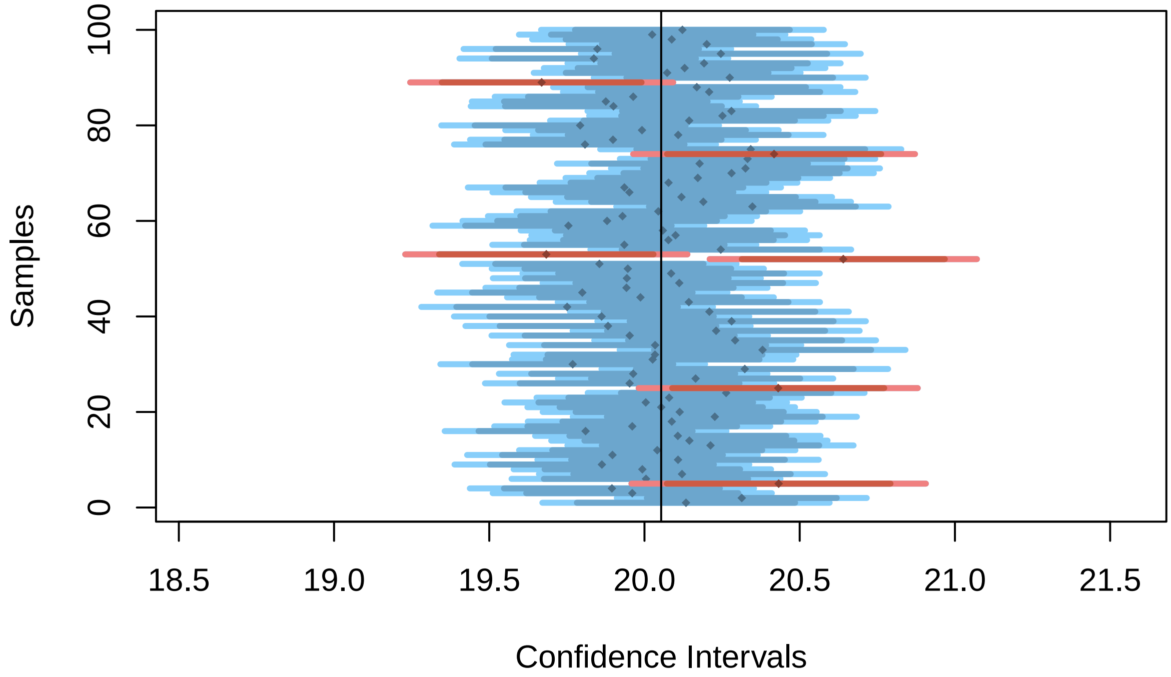
(i) We expect ~5% of samples to not include the population mean in its 95% CI;

(ii) We expect ~1% of samples to not include the population mean in their 99% CI.

## Question 2(b)

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

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



(i) Narrower;

(ii) We expect ~5% of samples to not include the population mean in its 95% CI. However, since the sample size is bigger, results have to be more precise.

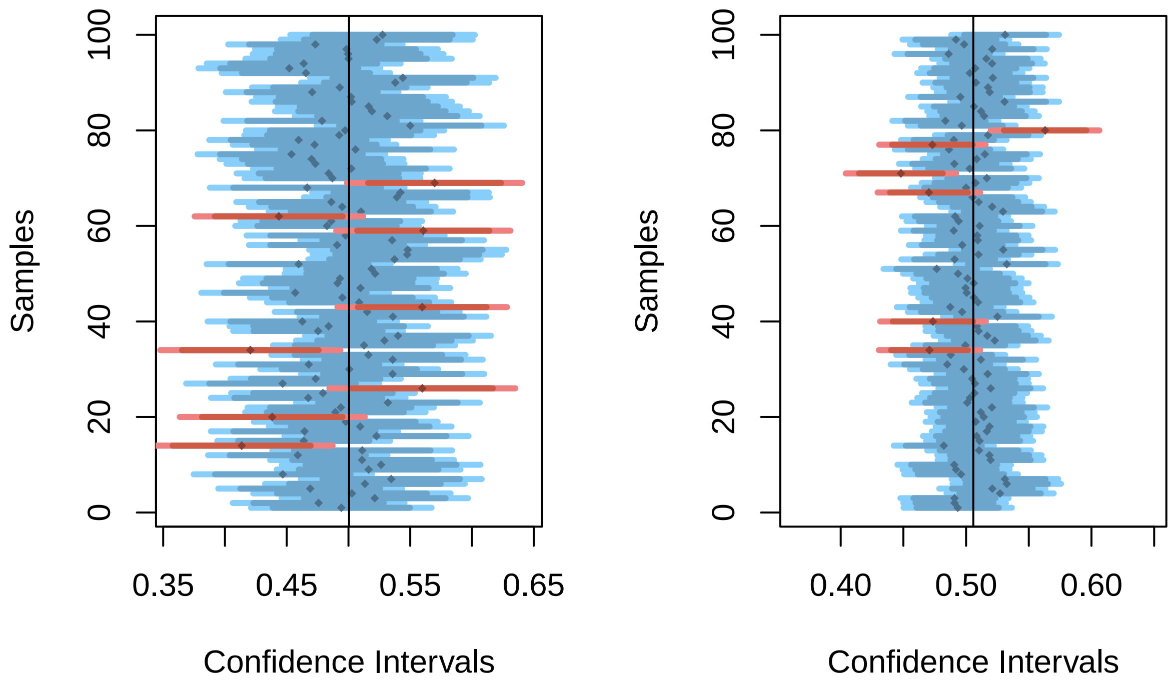
## Question 2(c)

*Rerun (a) and (b) using a uniformly distributed population.*

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

plot_sample_ci(num_samples = 100,
               sample_size = 100,
               pop_size=10000,
               distr_func=runif)

plot_sample_ci(num_samples = 100,
               sample_size = 300,
               pop_size=10000,
               distr_func=runif)
```



Expect to get similar results. Even if a population is non-normally distributed, as long as it is randomly drawn into independent samples, its sample means will be almost normally distributed. In other words, the explanation lies in the Central Limit Theorem.

## Question 3(a)

*What is the "average" booking time for new members making their first restaurant booking?*

```
mean_ <- mean(minday)
se <- sd(minday)/sqrt(length(minday))
CI_95 <- mean(minday) + c(-1.96*se, 1.96*se)

print(paste('mean:', mean_))
```

```
## [1] "mean: 942.49635"
```

```
print(paste('standard error:', se))
```

```
## [1] "standard error: 0.599767314943967"
```

```
print(paste('95% confidence interval:', CI_95[1], '-', CI_95[2]))
```

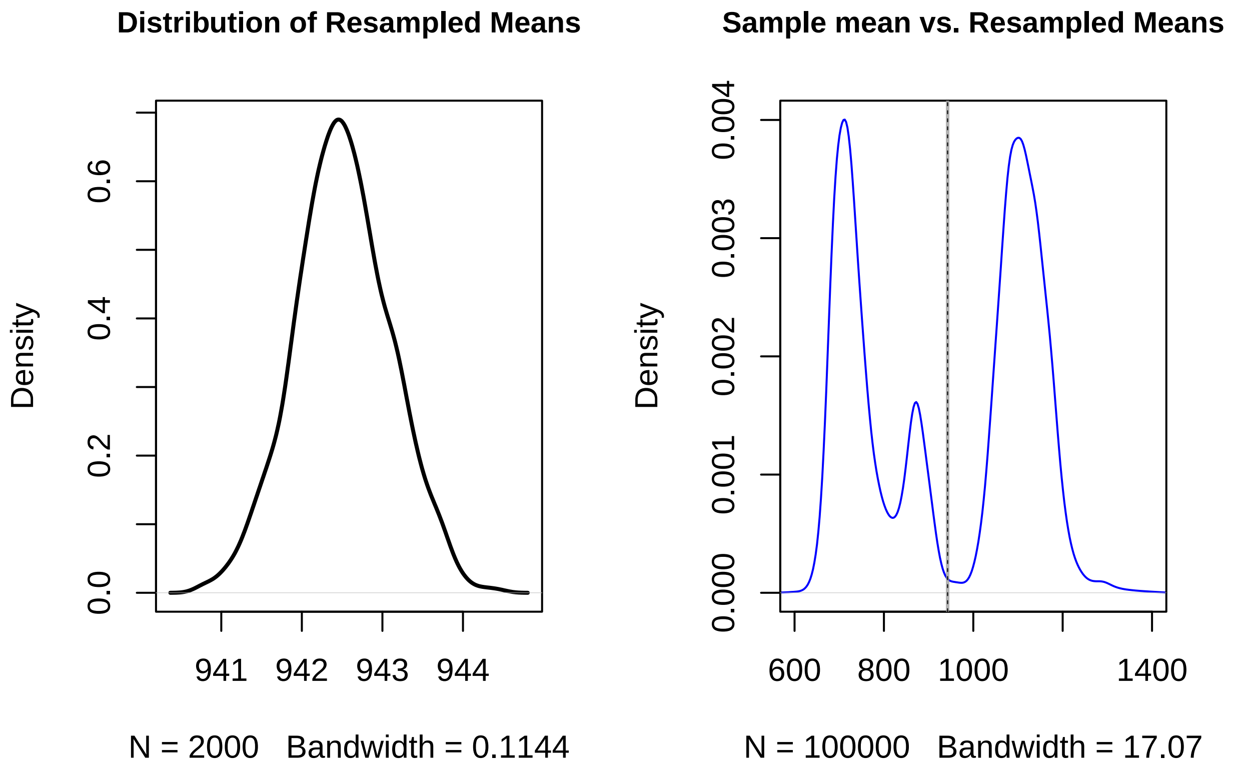
```
## [1] "95% confidence interval: 941.32080606271 - 943.67189393729"
```

```
resamples <- replicate(2000, sample(minday, length(minday), replace=TRUE))

sample_means <- c()

# calculate mean for each sample
for(i in 1:ncol(resamples)) {
  sample_means[i] <- mean(resamples[,i])
}
```

```
par(mfrow=c(1,2))
plot(density(sample_means), lwd=2,
     main='Distribution of Resampled Means',
     cex.main=0.9)
plot(density(minday), col="blue",
     xlim=c(600, 1400),
     main='Sample mean vs. Resampled Means',
     cex.main=0.9)
abline(v=sample_means, col=rgb(0.7, 0.7, 0.7, 0.02))
abline(v=mean_, lty="dashed",lw=0.5)
```



```
CI_95_ <- quantile(sample_means, probs=c(0.025, 0.975))
print(paste('95% CI of the bootstrapped means:', CI_95_[1], '-', CI_95_[2]))
```

```
## [1] "95% CI of the bootstrapped means: 941.338475 - 943.69184125"
```

We are 95% sure the average booking time is between 941.3 and 943.6

## Question 3(b)

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

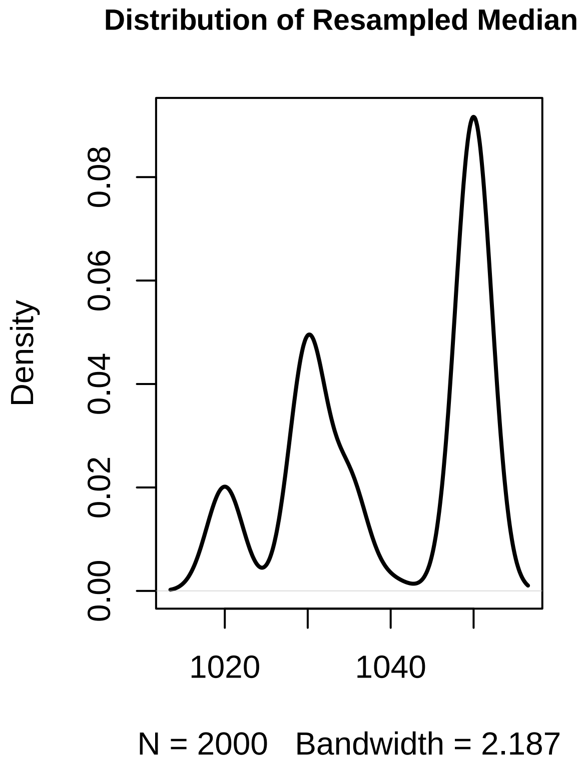
```
median_ <- median(minday)
print(paste('Median:', median_))
```

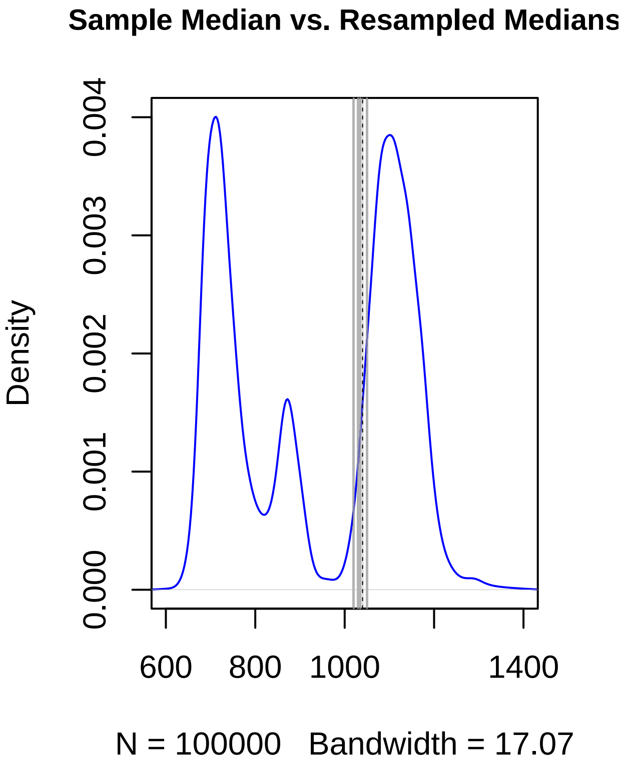
```
## [1] "Median: 1040"
```

```
sample_medians <- c()

# calculate median for each sample
for(i in 1:ncol(resamples)) {
  sample_medians[i] <- median(resamples[,i])
}

par(mfrow=c(1,2))
plot(density(sample_medians), lwd=2,
     main='Distribution of Resampled Medians',
     cex.main=0.9)
plot(density(minday), col="blue", xlim=c(600, 1400),
     main='Sample Median vs. Resampled Medians',
     cex.main=0.9)
abline(v=sample_medians, col=rgb(0.7, 0.7, 0.7, 0.02))
abline(v=median_, lty="dashed",lw=0.5)
```





```
CI_95_median <- quantile(sample_medians, probs=c(0.025, 0.975))
print(paste('95% CI of the bootstrapped medians in minutes:',
            CI_95_median[1], '-', CI_95_median[2]))
```

```
## [1] "95% CI of the bootstrapped medians in minutes: 1020 - 1050"
```

```
print(paste('95% CI of the bootstrapped medians in hours:',
            unname(CI_95_median[1]/60), '-', unname(CI_95_median[2]/60)))
```

```
## [1] "95% CI of the bootstrapped medians in hours: 17 - 17.5"
```

We are 95% confident that by 17.00-17.30, the half of new members have already arrived at the restaurant.