BASC HW Week4 106071041

Question 1 | Standardize the data

(a) rnorm(mean=940, sd=190) and standardize it

```
# Create a data
nd <- rnorm(1000, 940, 190)

# Standardize
rnorm_std <- (nd-mean(nd))/sd(nd)
```

(i) Expected mean and standard deviation of rnorm_std and why

mean = 0 and standard deviation = 1.
“Standardized” means make the mean and standard deviation respectively become 0 and 1.

```
# make 1+15e -> 1000000000000...
options(scipen = 999)

mean(rnorm_std)

## [1] -0.00000000000000001950063

sd(rnorm_std)

## [1] 1
```

(ii) The look of the distribution of rnorm_std and why

Bell-shaped.
That is how standard normal distribution looks like.

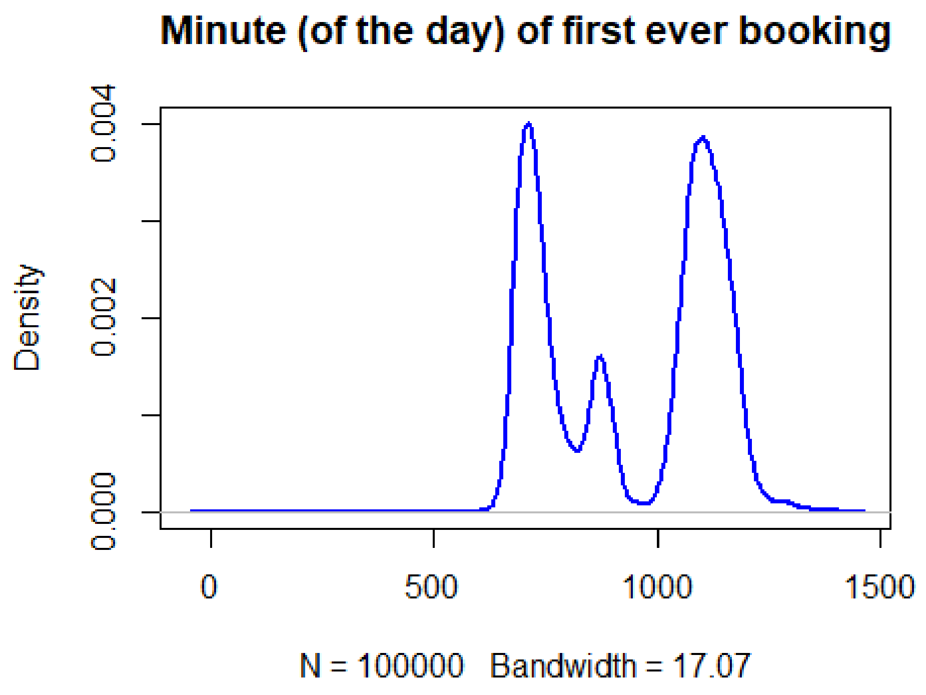
(iii) distributions that are normal and standardized?

Standard normal distribution.

(b) standardized version of minday

```
bookings <- read.table("first_bookings_datetime_sample.txt", header=TRU
E)
bookings$datetime[1:9]

## [1] "4/16/2014 17:30"  "1/11/2014 20:00"  "3/24/2013 12:00"  "8/8/20
13 12:00"
## [5] "2/16/2013 18:00"  "5/25/2014 15:00"  "12/18/2013 19:00" "12/23/
2012 12:00"
## [9] "10/18/2013 20:00"
```

```
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)
```



```
minday_std <- (minday - mean(minday))/sd(minday)
```

(i) expected mean and standard deviation of minday_std and why

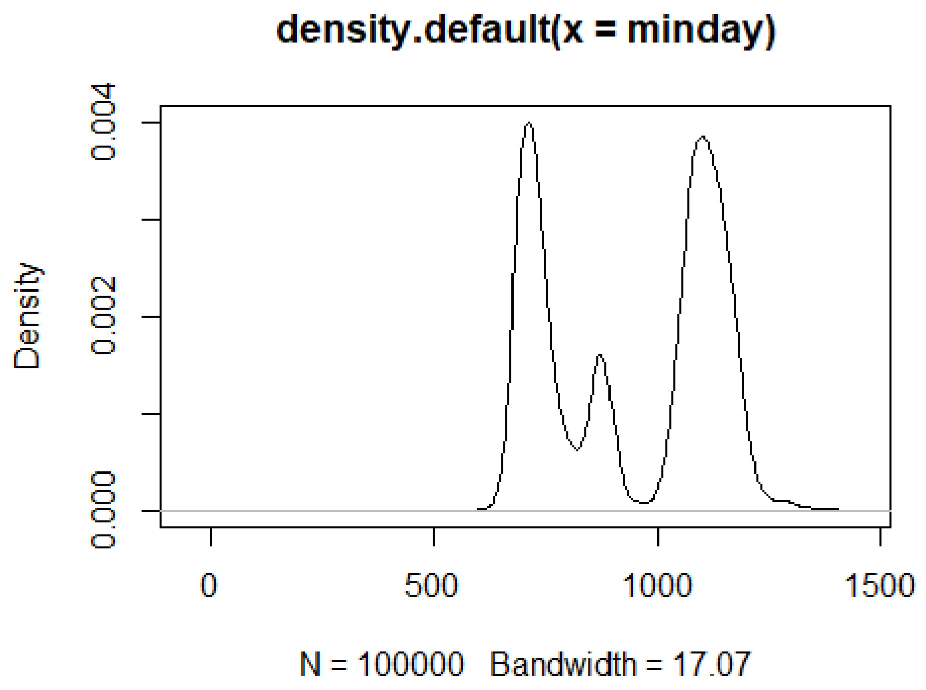
mean = 0 and standard deviation = 1.
"Standardized" means make the mean and standard deviation respectively become 0 and 1.

```
options(scipen = 999)

mean(minday_std)

## [1] -0.000000000000000425589

sd(minday_std)

## [1] 1
```

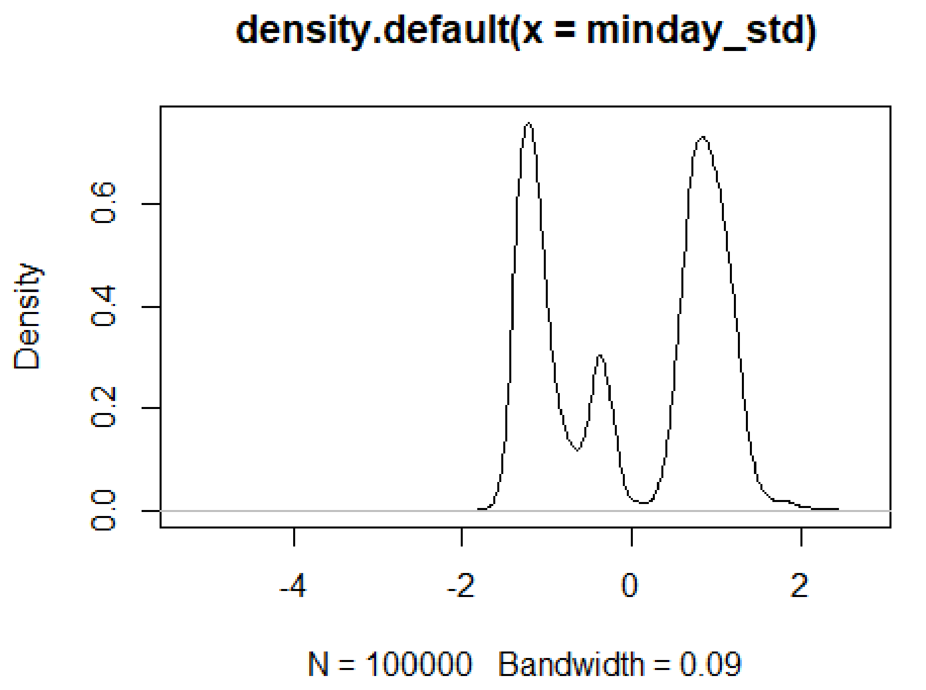
(ii) the look of the distribution of minday_std compared to minday, and why

looks similar.
minday is not normally distributed so will minday_std.

```
plot(density(minday))
```



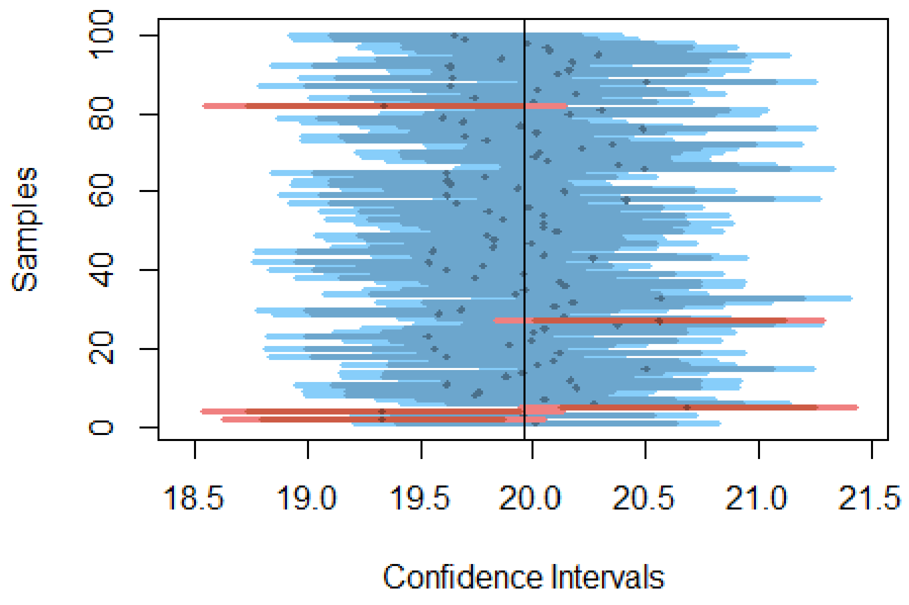
```
plot(density(minday_std))
```



Question 2 | Simulations

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

```r
visualize_sample_ci <- function(num_samples = 100, sample_size = 100,
                                pop_size=10000, distr_func, ...) {
  # Simulate a large population
  population_data <- distr_func(pop_size, ...)
  pop_mean <- mean(population_data)
  pop_sd <- sd(population_data)

  # Simulate samples
  samples <- replicate(num_samples,
                       sample(population_data, sample_size, replace=FAL
SE))

  # Calculate descriptives of samples
  sample_means = apply(samples, 2, FUN=mean)
  sample_stdevs = apply(samples, 2, FUN=sd)
  sample_stderrs <- sample_stdevs/sqrt(sample_size)
  ci95_low  <- sample_means - sample_stderrs*1.96
  ci95_high <- sample_means + sample_stderrs*1.96
  ci99_low  <- sample_means - sample_stderrs*2.58
  ci99_high <- sample_means + sample_stderrs*2.58

  # Visualize confidence intervals of all samples
  plot(NULL, xlim=c(pop_mean-(pop_sd/2), pop_mean+(pop_sd/2)),
       ylim=c(1,num_samples), ylab="Samples", xlab="Confidence Interval
s")
  add_ci_segment(ci95_low, ci95_high, ci99_low, ci99_high,
                 sample_means, 1:num_samples, good=TRUE)

  # Visualize samples with CIs that don't include population mean
  bad = which(((ci95_low > pop_mean) | (ci95_high < pop_mean)) |
              ((ci99_low > pop_mean) | (ci99_high < pop_mean)))
  add_ci_segment(ci95_low[bad], ci95_high[bad], ci99_low[bad], ci99_hig
h[bad],
                 sample_means[bad], bad, good=FALSE)

  # Draw true population mean
  abline(v=mean(population_data))
}

add_ci_segment <- function(ci95_low, ci95_high, ci99_low, ci99_high,
                           sample_means, indices, good=TRUE) {
  segment_colors <- list(c("lightcoral", "coral3", "coral4"),
                         c("lightskyblue", "skyblue3", "skyblue4"))
  color <- segment_colors[[as.integer(good)+1]]
  segments(ci99_low, indices, ci99_high, indices, lwd=3, col=color[1])
  segments(ci95_low, indices, ci95_high, indices, lwd=3, col=color[2])
  points(sample_means, indices, pch=18, cex=0.6, col=color[3])
}

visualize_sample_ci(num_samples = 100, sample_size = 100, pop_size=1000
0, distr_func=rnorm, mean=20, sd=3)
```



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

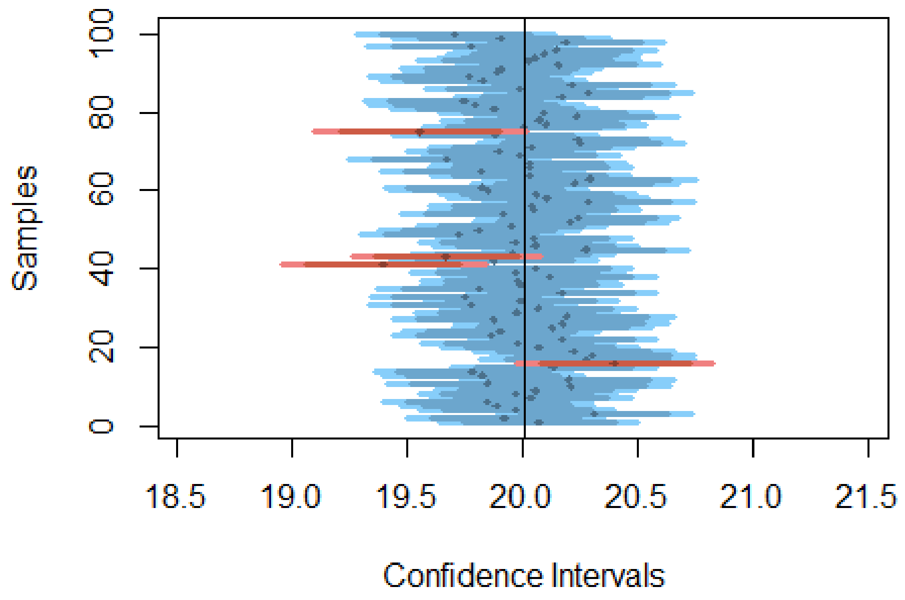
5

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

1

(b) Rerun the previous simulation with larger samples (sample_size=300)

```
visualize_sample_ci(num_samples = 100, sample_size = 300, pop_size=1000
0, distr_func=rnorm, mean=20, sd=3)
```



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

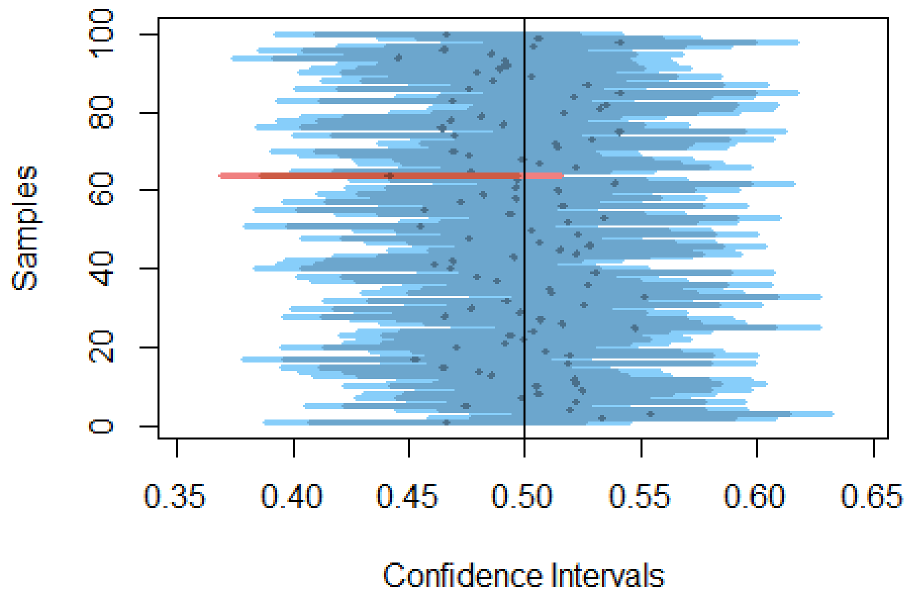
narrower.

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

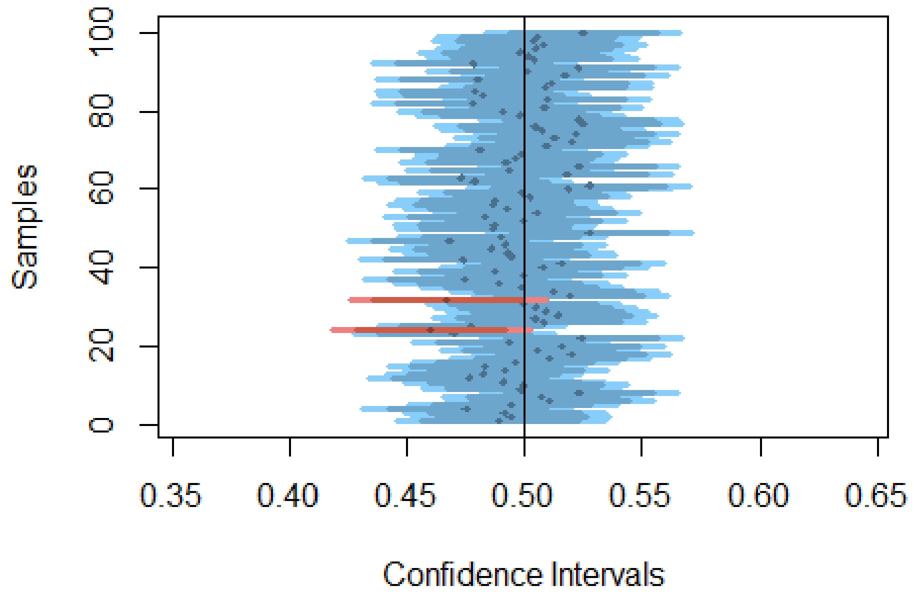
(c) ran the above two examples (a and b) using a uniformly distributed population, the answers to (a) and (b)?

the number of samples expected to not include the population mean will more than (a) and (b).
The population is more concentrated, around mean(bell-shaped), in normal distribution, so it's easier to get a sample within one range including mean.

```
visualize_sample_ci(num_samples = 100, sample_size = 100, pop_size=1000
0, distr_func=runif)
```



```
visualize_sample_ci(num_samples = 100, sample_size = 300, pop_size=1000
0, distr_func=runif)
```



Question 3

(a) “average” booking time

```
mean(minday)

## [1] 942.4964
```

(i) Estimate the population mean of minday, its standard error, and the 95% confidence interval (CI) of the sampling means

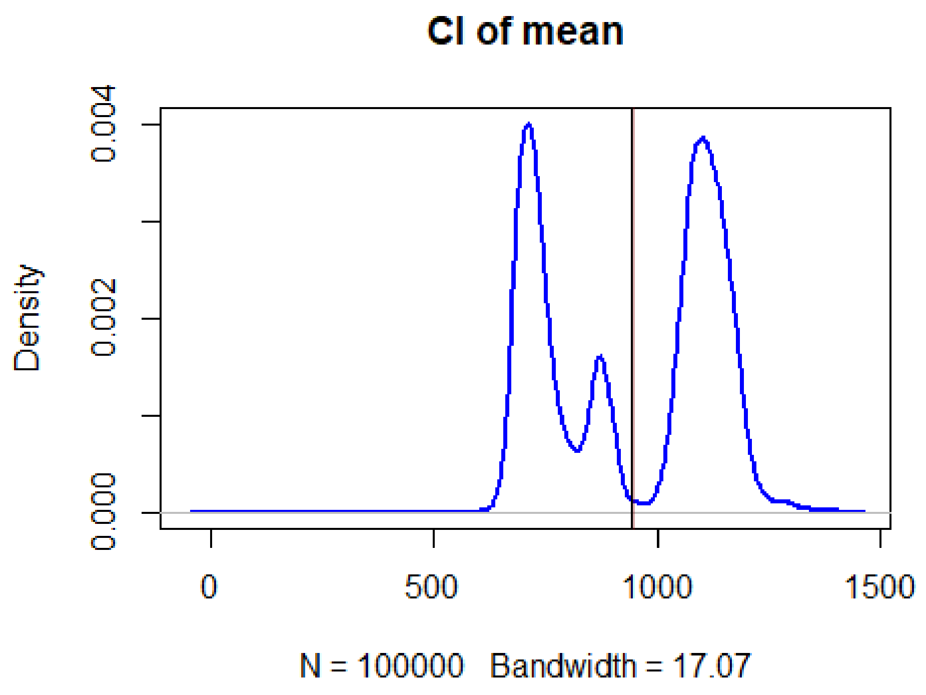
```
#mean
mean_minday <- sum(minday)/length(minday)
mean_minday

## [1] 942.4964

#standard deviation
std_minday <- (sum((minday - mean_minday) ** 2) / length(minday)-1)**(1
/2)
std_minday

## [1] 189.6595

#standard error
std_error_minday <- std_minday/sqrt(length(minday))
std_error_minday

## [1] 0.599756

#CI
quantiles <- unname(quantile(minday, c(0.025, 0.975)))
CI <- c(quantiles[1], quantiles[2])
CI

## [1]  690 1200
```

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

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

(iii) Visualize the means of the 2000 bootstrapped samples

```
plot_resample_mean <- function(sample_i){
  abline(v=mean(sample_i), col=rgb(0.5, 0.0, 0.0, 0.02))
}

plot(density(minday), col="blue", lwd=2, main="CI of mean")
apply(resample, 2, FUN=plot_resample_mean)

## NULL

abline(v=mean(minday), lwd=1)
```



(iv) Estimate the 95% CI of the bootstrapped means.

```
boots_means <- apply(resample, 2, mean)

# the 95% CI of the bootstrapped means
quantile(boots_means, c(0.025, 0.975))

##     2.5%    97.5% 
## 941.2880 943.6974
```

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

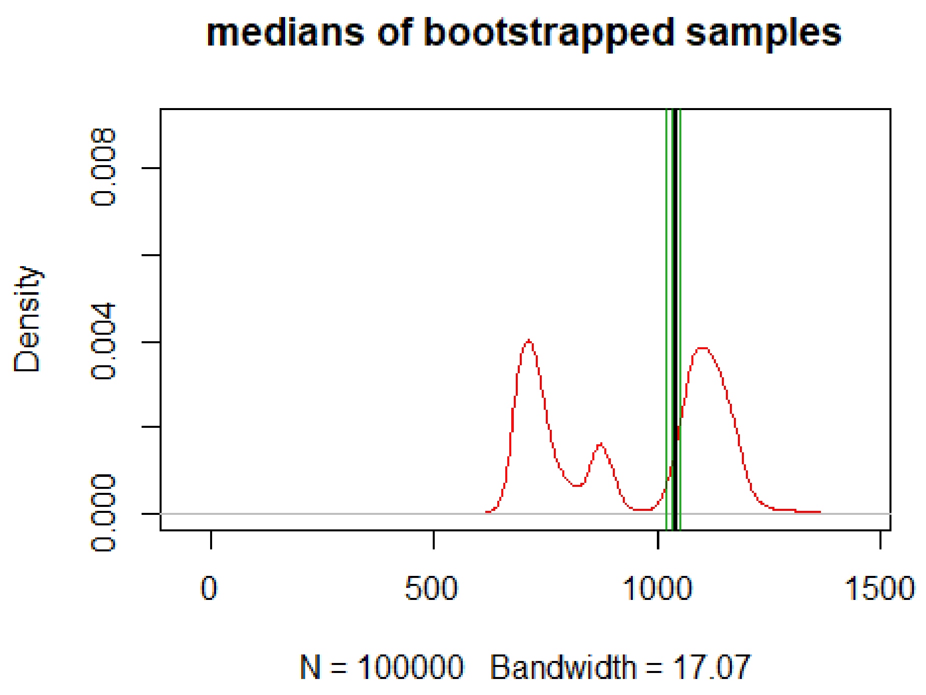
(i) Estimate the median of minday

```
median(minday)

## [1] 1040
```

(ii) Visualize the medians of the 2000 bootstrapped samples

```
plot_resample_median <- function(sample_i){
  abline(v=median(sample_i), col=rgb(0.0, 0.5, 0.0, 0.01))
}

plot(density(minday), lwd=1, ylim=c(0, 0.009), main="medians of bootstr
apped samples",col="red")

apply(resample, 2, plot_resample_median)
```

```
## NULL

abline(v=median(minday), lwd=2)
```



(iii) Estimate the 95% CI of the bootstrapped medians.

```
medians_boots <- apply(resample, 2, median)

quantile(medians_boots, c(0.025, 0.975))

##  2.5% 97.5%
##  1020  1050
```