HW (Week4)

106022103

2021/3/21

- Helped by: 106070038

Set up

import libary

```
library(ggplot2)
```

Useful function

Before go on the question, just write some useful function to make the answer more pretty.

```
# print the statistics property of a distritbution
dist_print <- function(dist, Mean=FALSE, Median=FALSE, Sd=FALSE){
  if (Mean){
    print(sprintf("Mean: %.3f",mean(dist)))
  }
  if (Median){
    print(sprintf("Median: %.3f",median(dist)))
  }
  if (Sd){
    print(sprintf("Standard deviation: %.3f",sd(dist)))
  }
}
```

```
# plot the distribution
dist_ggplot <- function(dist,plot_dist=TRUE,plot_density=TRUE){
  p <- ggplot(mapping = aes(dist))
  p + geom_histogram(bins=nclass.Sturges(dist))
  p + geom_area(stat = "density", fill = "cadetblue2")
}
```

Read data

```
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
```

Q1

> Let's reexamine what it means to standardize data. To standardize a vector, subtract the mean of the vector from all its values, and then divide them by the standard deviation.

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

```
rnorm_std <- rnorm(n=1e4, mean=940, sd=190)
```

i) What should we expect the mean and standard deviation of rnorm_std to be, and why? **ANSWER:** In this question, I choose $n = 1 \times 10^4$, and it's a sample from population. The sample distribution should be in the range of $\mu \approx \bar{x} \pm z_{\alpha/2}\frac{\sigma}{\sqrt{n}}$.
So we **expect the mean should be** 940 ± 3.724 **under the 95% confidence level**.
The standard deviation should follow the Chi-Square distribution, so as the equation of $\sigma^2 \approx (LCL, UCL)$
$LCL = \frac{(n-1)s^2}{\chi^2_{n-1,\alpha/2}} \approx 35119.859$
$UCL = \frac{(n-1)s^2}{\chi^2_{n-1,1-\alpha/2}} \approx 37121.958$
So we **expect the standard deviation should be** $\sigma \approx (187.4, 192.7)$ **under the 95% confidence level**.

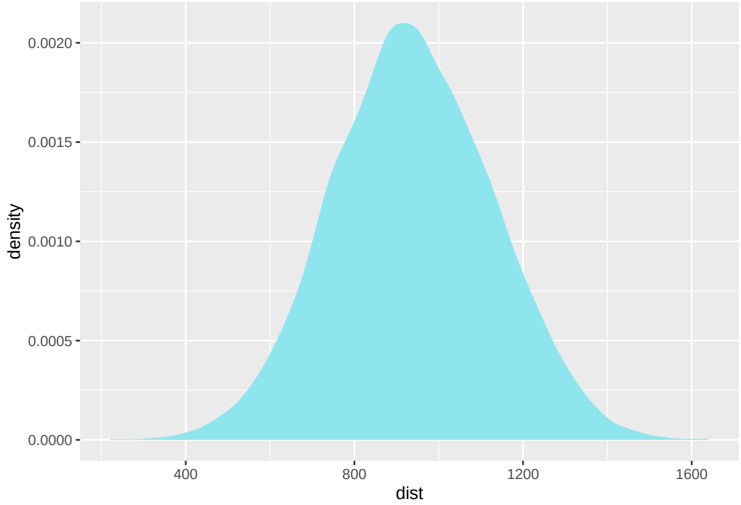
Let's check the value with our exception.

```
dist_print(rnorm_std,Mean=TRUE, Sd=TRUE)
```

```
## [1] "Mean: 937.884"
## [1] "Standard deviation: 190.347"
```

ii) What should the distribution (shape) of rnorm_std look like, and why? **ANSWER:** We can except that the shape should be in bell shape (normal distribution), because a sample from normal distribution should still follow the normal distribution. Let's check it.

```
dist_ggplot(rnorm_std)
```



iii) What do we generally call distributions that are normal and standardied? **ANSWER:** We have many different names to call it, such as
+ Gaussian distribution + Z distribution

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

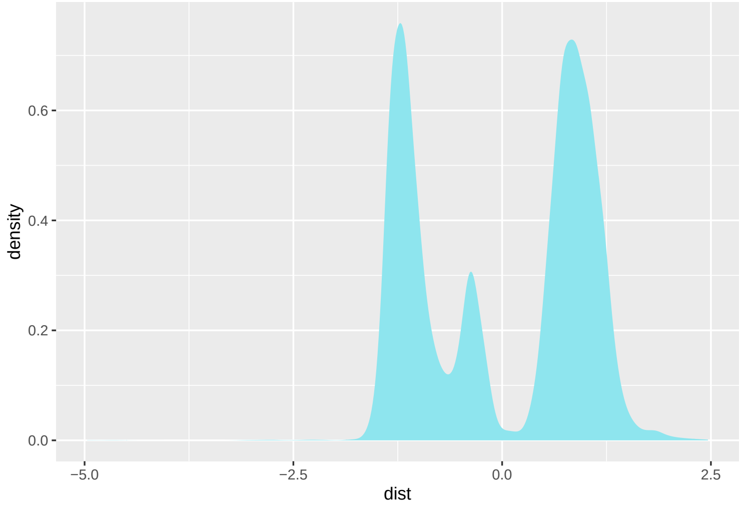
```
minday_std <- scale(minday)
```

(i) What should we expect the mean and standard deviation of minday_std to be, and why? **ANSWER:** After standardized, it's equal the distribution with `rnorm(n,mean=0,sd=1)`. The sample

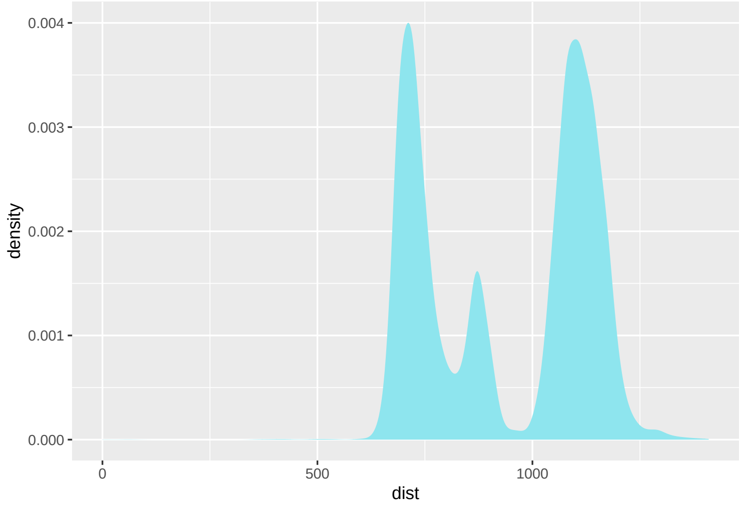
distribution should be in the range of $\mu \approx \bar{x} \pm z_{\alpha/2}\frac{\sigma}{\sqrt{n}}$.
So we **expect the mean should be** 0 ± 0.0196 **under the 95% confidence level**.
The standard deviation should follow the Chi-Square distribution, so as the equation of $\sigma^2 \approx (LCL, UCL)$
$LCL = \frac{(n-1)s^2}{\chi^2_{n-1,\alpha/2}} \approx 0.973$
$UCL = \frac{(n-1)s^2}{\chi^2_{n-1,1-\alpha/2}} \approx 1.028$
So we **expect the standard deviation should be** $\sigma \approx (0.973, 1.028)$ **under the 95% confidence level**.

(ii) What should the distribution of minday_std look like compared to minday, and why?
ANSWER: The shape is the same. Because the standardized process have only extension and translation variations. $Z = \frac{X-\mu}{\sigma}$

```
dist_ggplot(minday_std)
```



```
dist_ggplot(minday)
```



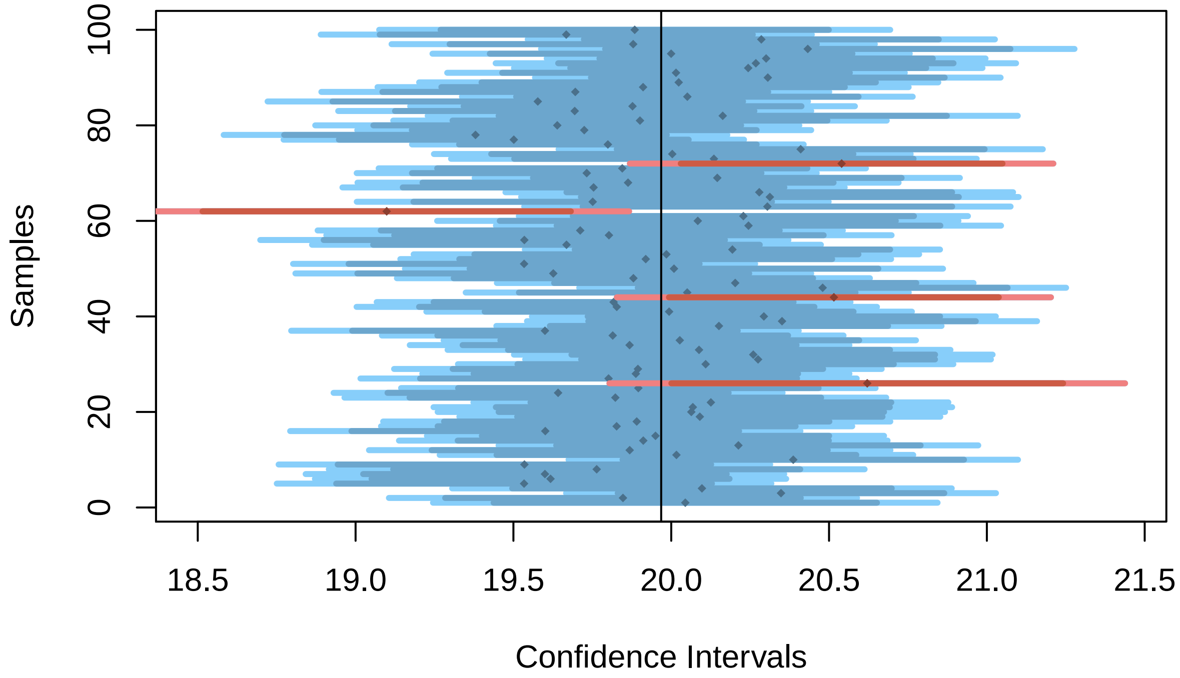
Q2

Copy and run the code we used in class to create simulations of confidence intervals. Run visualize_sample_ci(), which 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.

```
# Visualize the confidence intervals of samples drawn from a population
#   e.g.,
#     visualize_sample_ci(sample_size=300, distr_func=rnorm, mean=50, sd=10)
#     visualize_sample_ci(sample_size=300, distr_func=runif, min=17, max=35)
visualize_sample_ci <- function(num_samples = 100, sample_size = 100,
                                pop_size=10000, distr_func=rnorm, ...) {
  # 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=FALSE))

  # 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 Intervals")
  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_high[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])
}
```

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

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

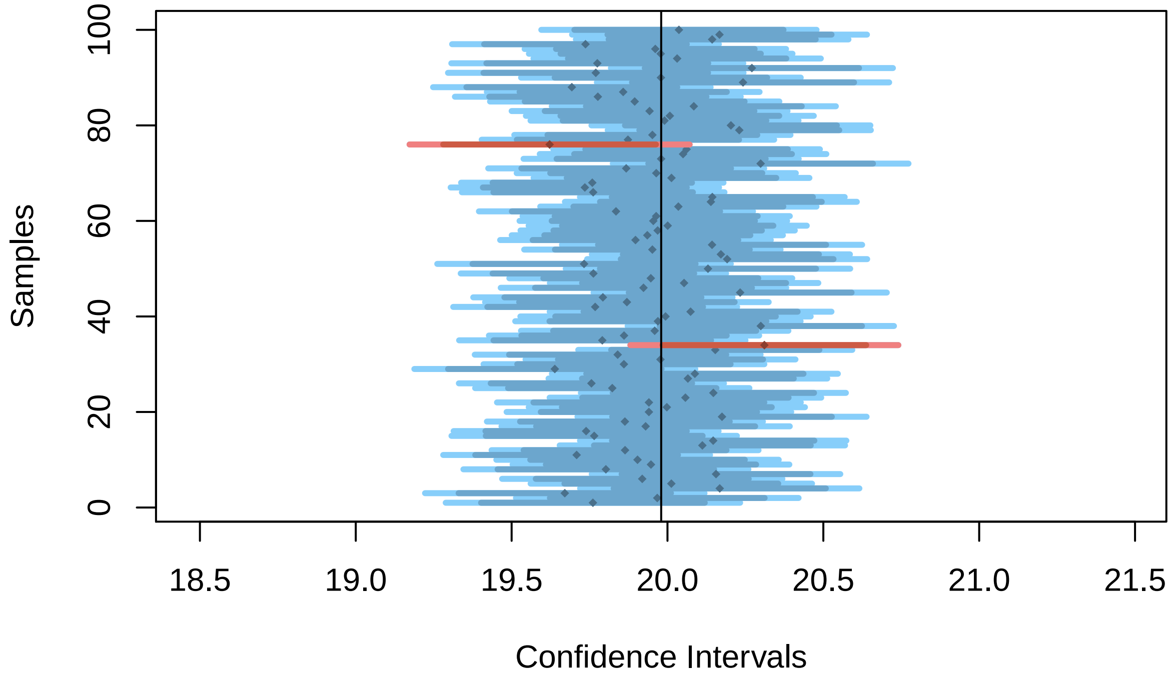


(i) How many samples do we expect to NOT include the population mean in its 95% CI?
ANSWER: Since the population is finite, we can use $\sigma_{\bar{x}} = \sqrt{\frac{N-n}{N-1}}\frac{\sigma}{\sqrt{n}}$ to get the standard deviation of samples. Take $n = 100$, I except there will be $100 \times (1 - 95\%) \times \sqrt{\frac{10000-100}{10000-1}} \approx 4.975$ not include the population in its 95% CI.

(ii) How many samples do we expect to NOT include the population mean in their 99% CI?
ANSWER: Since the population is finite, we can use $\sigma_{\bar{x}} = \sqrt{\frac{N-n}{N-1}}\frac{\sigma}{\sqrt{n}}$ to get the standard deviation of samples. Take $n = 100$, I except there will be $100 \times (1 - 99\%) \times \sqrt{\frac{10000-100}{10000-1}} \approx 0.995$ not include the population in its 99% CI.

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

```
visualize_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? ANSWER: Since the population is finite, we can use $\sigma_{\bar{x}} = \sqrt{\frac{N-n}{N-1}}\frac{\sigma}{\sqrt{n}}$ to get the standard deviation of samples.

Take $n = 300$, I except there will be $100 \times (1 - 95\%) \times \sqrt{\frac{10000-300}{10000-1}} \approx 14.925$ not include the population in its 95% CI, and $100 \times (1 - 99\%) \times \sqrt{\frac{10000-300}{10000-1}} \approx 0.985$ not include the population in its 95% CI

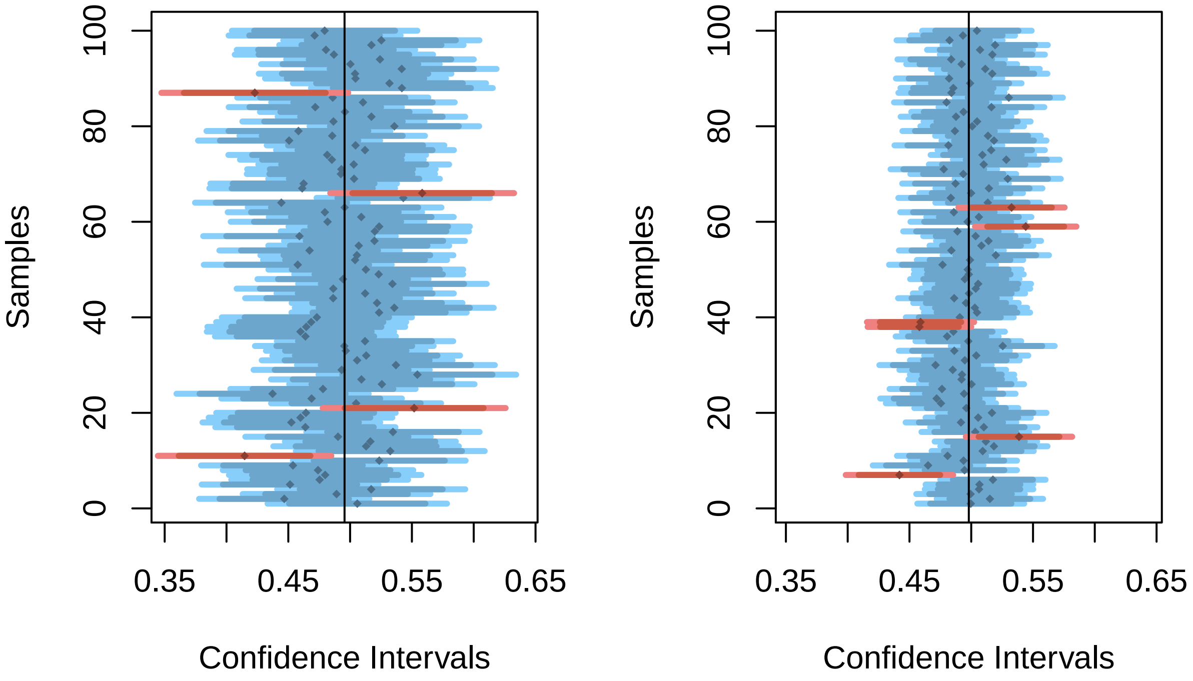
(ii) This time, how many samples (out of the 100) would we expect to NOT include the population mean in its 95% CI? ANSWER: As the results of the calculation show, we can find that the larger n is, the less the result of exceeding the CI. Further, we compare the number of results exceeding the CI for n from 1 to 100000 respectively.

n	out of 95%CI	out of 99%CI
1	5	1
100	4.975	0.995
300	4.925	0.985
1000	4.744	0.949
5000	3.536	0.707
8000	2.236	0.447
10000	0	0

It is worth to be noted that when n=10000 there will not be any results over CI. It make sense because we use all population.

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

```
par(mfcol=c(1,2))
visualize_sample_ci(num_samples = 100, sample_size = 100, pop_size=10000,
                    distr_func=runif)
visualize_sample_ci(num_samples = 100, sample_size = 300, pop_size=10000,
                    distr_func=runif)
```

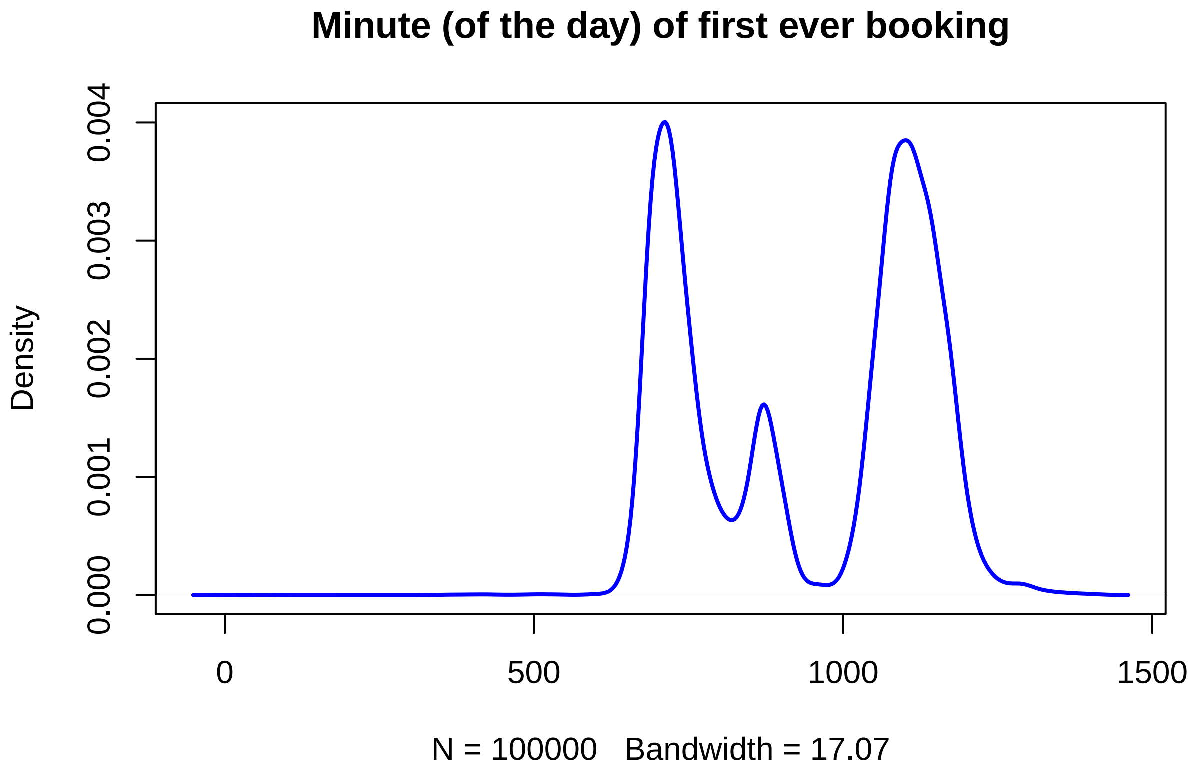


ANSWER: I except the answer in (a) and (b) will not change. Because even the distribution functions are different, the 95% CI will still contains 95% of values (99% CI is the same). The result is still the same, the larger `sample_size` is, the less number out of 95% or 99% CI will be.

Q3

> The startup company EZTABLE has an online restaurant reservation system 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:

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



(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)

```
dist_print(minday,Mean=TRUE,Sd=TRUE)
```

```
## [1] "Mean: 942.496"
## [1] "Standard deviation: 189.663"
```

```
CI <- quantile(minday, probs=c(0.025, 0.975))
sprintf("The 95%% of CI should be (%.2f,%.2f)",CI[1],CI[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

```
## [1] "The 95% of CI should be (690.00,1200.00)"
```

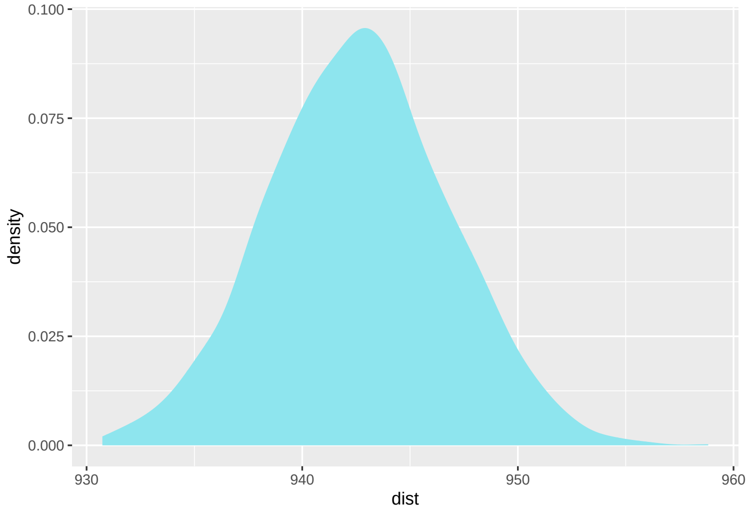
(ii) Bootstrap to produce 2000 new samples from the original sample

> The question is a little unclear, so I decide to take 2000 as the number of sample size each time,and bootstrap 2000 times.

```
resample <- lapply(1:2000,function(x){sample(minday,2000)})
```

(iii) Visualize the means of the 2000 bootstrapped samples ANSWER:

```
resample_mean <- sapply(resample,mean)
dist_ggplot(resample_mean)
```



```
CI <- quantile(resample_mean, probs=c(0.025, 0.975))
sprintf("The 95%% of CI should be (%.2f,%.2f)",CI[1],CI[2])
```

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

```
## [1] "The 95% of CI should be (934.34,950.92)"
```

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

ANSWER: That's the median(50% quantile) of `minday`.

```
sprintf("At %d minutes of day, have half the new members of the day already arrived at their restaurant
```

```
## [1] "At 1040 minutes of day, have half the new members of the day already arrived at their restauran
```

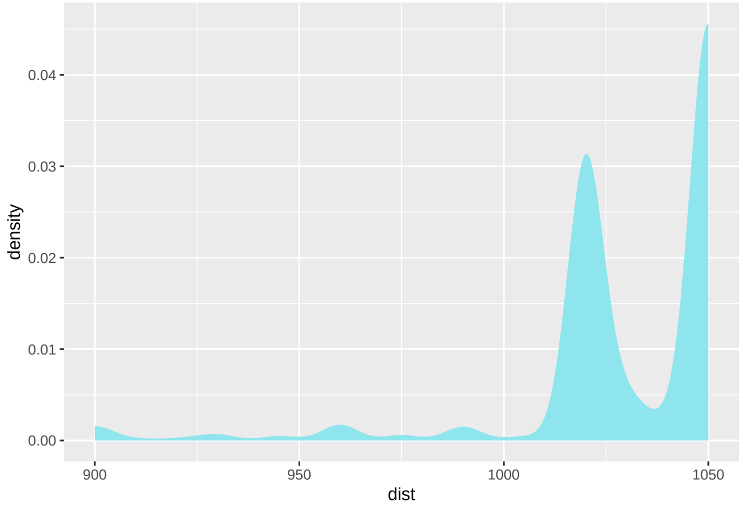
```
dist_print(minday, Median=TRUE)
```

(i) Estimate the median of minday

```
## [1] "Median: 1040.000"
```

(ii) Visualize the medians of the 2000 bootstrapped samples ANSWER:

```
resample_median <- sapply(resample,median)
dist_ggplot(resample_median)
```



```
CI <- quantile(resample_median, probs=c(0.025, 0.975))
sprintf("The 95%% of CI should be (%.2f,%.2f)",CI[1],CI[2])
```

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

```
## [1] "The 95% of CI should be (930.00,1050.00)"
```

Reference

- Sample and distribution
- ggplot
-