Tut3

```r
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.3.3     v purrr   0.3.4
## v tibble  3.0.4     v dplyr   1.0.2
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.0

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

PART A

```r
set.seed(1014)
effect_size <- read_csv('effect_a.csv')
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   Utility = col_double()
## )
```

```r
B = 1000 ## 1000 bootstrapped samples/ simulate 1000 times
n = 200 ## 200 observations per bootstrapped sample
boot_sample <- matrix(sample(effect_size$Utility, size = B * n, replace = TRUE), B, n)

sample_median <- median(effect_size$Utility)
boot_median <- apply(boot_sample, 1, median)
boot_dev <- boot_median - sample_median
boot_ci <- quantile(boot_dev, c(0.025, 0.975)) + sample_median
glimpse(boot_sample)
```

```
##  num [1:1000, 1:200] -1.25 2.7 2.1 1.75 3.23 ...
```

```r
glimpse(boot_dev)
```

```
##  num [1:1000] -0.01397 -0.05112 0.00781 -0.07682 -0.05994 ...
```

```r
glimpse(boot_median)
```

```
##  num [1:1000] 1.76 1.73 1.78 1.7 1.72 ...
```

```r
boot_ci
```

```
##     2.5%    97.5%
## 1.687255 1.892901
```

```r
median(sample_median) # Median of the original dataset
```

```
## [1] 1.776166
```

```r
# below is for the result of Mixture of Gaussian; not considered as part of solution
set.seed(1018)
#The number of samples from the true (mixture) distribution
N <- 200
#Sample N datapoints from a Uniform Distribution
U <- runif(N)
#Store the samples from the mixture distribution
result_median <- c()

for (x in 1:B) {
  samples <- array(NA, dim=N)
#Sampling from the mixture
for(i in 1:N){
if(U[i] < 0.8){
samples[i] = rnorm(1,2,0.5)
}else{
samples[i] = rnorm(1,-1,0.5)
}
}
  result_median <-c(result_median, median(samples))
}
mix_dev <- result_median - sample_median
mix_ci <- quantile(mix_dev, c(0.025, 0.975)) + sample_median
mix_ci
```

```
##     2.5%    97.5%
## 1.693801 1.904278
```

a): according to result printed from line 33, the confidence interval of bootstrapped sample has 95% confidence interval around 1.669 to 1.896.

b): The confidence interval is not too symmetric around true median of original sample, since 1.77-1.69=0.08, 1.89-1.77=0.12, where the left-right difference is differed by 0.04.

The result is reasonable to exist, since bootstrapped sample somehow resembles the distribution of population. This likely implies the population's distribution of data is not symmetric around median.

c): The bootstrapped sample's confidence interval includes the true median of original dataset, because "1.84", which is the population median, lies in [1.69, 1.89] (but barely contains the true median, it's near boundary of estimation).

d): Gaussian's method can generate data not exist in original sample (meaning median may not exist in sample), while empirical bootstrapped sample's median is always in original sample.

PART B: (a) code below:

```
set.seed(1013)
effect_size2 <- read_csv('effect_b.csv')
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   Utilities = col_double()
## )
```

```
B = 1500 ## 1500 bootstrapped samples/ simulate 1500 times
n = 30 ## 30 observations per bootstrapped sample
boot_sample2 <- matrix(sample(effect_size2$Utilities, size = B * n, replace = TRUE), B, n)

sample_mean2 <- mean(effect_size2$Utilities)
boot_mean2 <- apply(boot_sample2, 1, mean)
boot_dev2 <- boot_mean2 - sample_mean2
boot_ci2 <- quantile(boot_dev2, c(0.025, 0.975)) + sample_mean2
boot_ci2
```

```
##     2.5%    97.5%
## 1.792334 2.765355
```

(a) the bootstrapped CI for mean of utility is between 1.792 and 2.765.

(b) code below:

```
expectation <-mean(effect_size2$Utilities)
std.dev <- sd(effect_size2$Utilities)
expectation
```

```
## [1] 2.235561
```

```
std.dev
```

```
## [1] 1.414474
```

```
# qqnorm(effect_size2$Utilities)

# will use t-distribution since population variance is unknown
# refer to t-distribution table, t-value with degree 29 and alpha 0.025 is 2.05
CLTlower <- expectation - 2.05 * std.dev / (sqrt(30))
CLTupper <- expectation + 2.05 * std.dev / (sqrt(30))
CLTquant <- c(CLTlower, CLTupper)
CLTquant
```

```
## [1] 1.706156 2.764966
```

note: used t-distribution with degree of freedom 29 and alpha 0.025 (one-side), which is 2.05.

Thus interval constructed using CLT is between 1.706 and 2.765.

(c): code below:

```
set.seed(1013)
boot_ci <- matrix(NA, nrow=1000, ncol=2)
clt_ci <- matrix(NA, nrow=1000, ncol=2)

B=1500
n=30

for(i in 1:1000){
effect <- rgamma(n=30, shape=4, rate=2)
## Build a bootstrapped sample and calculate its mean
## Code goes in here
#########################
boot_sample3 <- matrix(sample(effect, size = B * n, replace = TRUE), B, n)

sample_mean3 <- mean(effect)
boot_mean3 <- apply(boot_sample3, 1, mean)
boot_dev3 <- boot_mean3 - sample_mean3
boot_ci3 <- quantile(boot_dev3, c(0.025, 0.975)) + sample_mean3


expectation2 <-mean(effect)
std.dev2 <- sd(effect)

CLTlower2 <- expectation2 - 2.05 * std.dev2 / (sqrt(30))
```

```r
CLTupper2 <- expectation2 + 2.05 * std.dev2 / (sqrt(30))
CLTquant2 <- c(CLTlower2, CLTupper2)

## Construct bootstrapped CI and CLT-based CI for each iteration
boot_ci[i, ] <- boot_ci3
clt_ci[i,] <-CLTquant2
########################

}
## Determine the proportion of CIs that cover the true population mean, mu=2.
boot_contain_mu <- sum(apply(boot_ci, 1, function(x){x[1] <2 & x[2] > 2}))
CLT_contain_mu <- sum(apply(clt_ci, 1, function(x){x[1] <2 & x[2] > 2}))
print(boot_contain_mu)
```

```
## [1] 940
```

```r
print(CLT_contain_mu)
```

```
## [1] 951
```

```r
print(boot_contain_mu/1000)
```

```
## [1] 0.94
```

```r
print(CLT_contain_mu/1000)
```

```
## [1] 0.951
```

Thus the proportion of bootstrapped sample's CI containing true population mean is 94.0%, while there are 95.1% CLT confidence intervals containing true population mean.

d): Since empirical bootstrapping is applied, the result is not as desirable as CLT's prediction result, with a 11% difference in prediction accuracy.

Thus even when sample size is as small as 30 for this question, empirical bootstrapping is still not as accurate as t-distribution's estimation (one possible reason is the shape of gamma distribution with scale=4, rate=2 resembles a translated-normal-distribution), since empirical bootstrapping cannot improve the estimate of original sample as bootstrapped-sample is drawn from it.

However bootstrapped sample is a relatively easier way of constructing CI since the procedure is easier than using CLT. It also does not require assumptions that the data is normally distributed. When the data is not distributed approximately normal, using bootstrapped method is also safer than CLT.