Crimtab Data (with noise) for Simulation of T-Distribution

coop711

2018-03-31

Data Loading

```
load("./crimtab_test.RData")
ls()
```

```
## [1] "crimtab_2"             "crimtab_df"            "crimtab_long"
## [4] "crimtab_long_df"       "crimtab_long_df_noise" "r_noise"
```

```
ls.str()
```

```
## crimtab_2 :  'table' int [1:42, 1:22] 0 0 0 0 0 0 1 0 0 0 ...
## crimtab_df : 'data.frame':   924 obs. of  3 variables:
##  $ finger: num  9.4 9.5 9.6 9.7 9.8 9.9 10 10.1 10.2 10.3 ...
##  $ height: num  56 56 56 56 56 56 56 56 56 56 ...
##  $ Freq  : int  0 0 0 0 0 0 1 0 0 0 ...
## crimtab_long :  num [1:3000, 1:2] 10 10.3 9.9 10.2 10.2 10.3 10.4 10.7 10 10.1 ...
## crimtab_long_df : 'data.frame':  3000 obs. of  2 variables:
##  $ finger: num  10 10.3 9.9 10.2 10.2 10.3 10.4 10.7 10 10.1 ...
##  $ height: num  56 57 58 58 58 58 58 58 59 59 ...
## crimtab_long_df_noise : 'data.frame':    3000 obs. of  2 variables:
##  $ finger: num  9.98 10.29 9.91 10.24 10.17 ...
##  $ height: num  55.8 56.9 58.1 58.4 57.7 ...
## r_noise :  num [1:3000] -0.2345 -0.1279 0.0729 0.4082 -0.2983 ...
```

```
head(crimtab_long_df_noise,
     n = 10)
```

```
##       finger   height
## 1   9.976551 55.76551
## 2  10.287212 56.87212
## 3   9.907285 58.07285
## 4  10.240821 58.40821
## 5  10.170168 57.70168
## 6  10.339839 58.39839
## 7  10.444468 58.44468
## 8  10.716080 58.16080
## 9  10.012911 59.12911
## 10 10.056179 58.56179
```

Student 의 Simulation 재현

Sample t-values

3,000장의 카드를 잘 섞는 것은 `sample()` 이용.

```
# set.seed(113)
crimtab_shuffle_noise <- crimtab_long_df_noise[sample(1:3000), ]
head(crimtab_shuffle_noise,
     n = 10)
```

```
##        finger   height
## 2408 11.32767 68.27667
## 1439 11.93634 65.36344
## 532  11.25294 62.52938
## 1191 11.32788 65.27884
## 2440 11.54257 68.42574
## 2820 12.66086 68.60859
## 520  11.21379 63.13791
## 1063 10.63788 65.37884
## 2110 11.61868 67.18683
## 2870 12.05588 69.55877
```

표본의 크기가 4인 750개의 표본을 만드는 작업은 `rep()` 이용.

```
sample_id <- as.factor(rep(1:750, each = 4))
head(sample_id, n = 10)
```

```
##  [1] 1 1 1 1 2 2 2 2 3 3
## 750 Levels: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 ... 750
```

각 표본의 평균과 표준편차 계산에는 `tapply()` 이용.

```
finger_sample_mean <- tapply(crimtab_shuffle_noise[, "finger"],
                             INDEX = sample_id,
                             FUN = mean)
finger_sample_sd <- tapply(crimtab_shuffle_noise[, "finger"],
                           INDEX = sample_id,
                           FUN = sd)
str(finger_sample_mean)
```

```
##  num [1:750(1d)] 11.5 11.5 11.5 11.6 11.7 ...
##  - attr(*, "dimnames")=List of 1
##   ..$ : chr [1:750] "1" "2" "3" "4" ...
```

```
str(finger_sample_sd)
```

```
##  num [1:750(1d)] 0.319 0.851 0.527 0.226 0.532 ...
##  - attr(*, "dimnames")=List of 1
##   ..$ : chr [1:750] "1" "2" "3" "4" ...
```

t-통계량 계산. Student는 표준편차 계산에서 분모에 n을 사용하고 히스토그램을 그려 비교하였으나 자유도 3인 t-분포와 비교하기 위하여 $t = \frac{\bar{X}_n - \mu}{\hat{SD}/\sqrt{n}}$을 계산함. (여기서 $\hat{SD}$는 표본 표준편차)

```
sample_t <- (finger_sample_mean - mean(crimtab_long_df_noise[, "finger"]))/(finger_sa
mple_sd/sqrt(4))
str(sample_t)
```

```
##  num [1:750(1d)] -0.5364 -0.0773 -0.1729 0.3091 0.4304 ...
##  - attr(*, "dimnames")=List of 1
##   ..$ : chr [1:750] "1" "2" "3" "4" ...
```

계산한 t-통계량 값들의 평균과 표준편차, 히스토그램을 그리고 자유도 3인 t-분포의 밀도함수 및 표준정규곡선과 비교. 우선 모두 같은 값들이 나와서 분모가 0인 경우가 있는지 파악. 있으면 모평균과 비교하여 양수인 경우 +6, 음수인 경우 -6 값 부여 (Student가 한 일)

```
t_inf <- is.infinite(sample_t)
sample_t[t_inf]
```

```
## named numeric(0)
```

```
sample_t[t_inf] <- 6 * sign(sample_t[t_inf])
```

문제되는 값이 없는 것을 확인하고, 평균과 표준편차 계산. 자유도 n인 t-분포의 평균과 표준편차는 각각 0과 $\sqrt{\frac{n}{n-2}}$임을 상기할 것. -6이나 +6보다 큰 값이 상당히 자주 나온다는 점에 유의.

```
mean(sample_t)
```

```
## [1] 0.04320298
```

```
sd(sample_t)
```

```
## [1] 1.587887
```

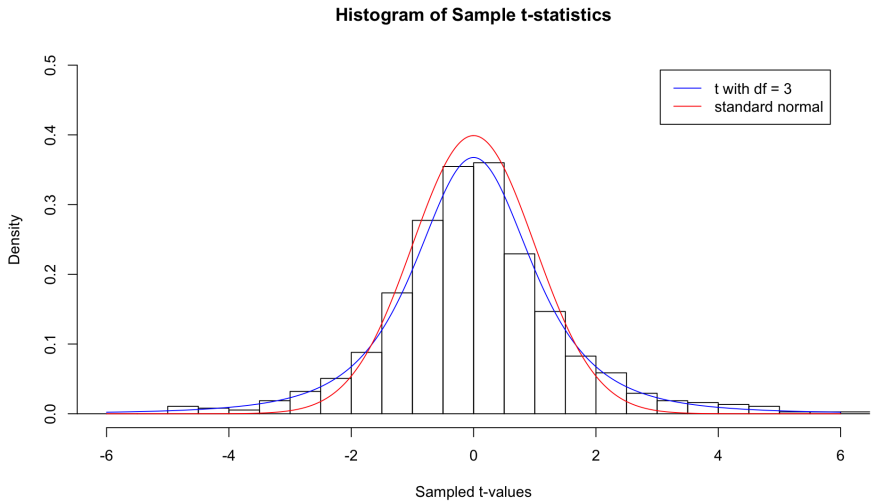
```
summary(sample_t)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -8.18669 -0.73848 -0.05273  0.04320  0.71279 13.05629
```

Histogram and Density Curves

t-통계량들의 히스토그램을 그리고, 자유도 3인 t의 밀도함수, 표준정규분포 밀도함수와 비교.

```
# hist(sample_t, prob = TRUE, ylim = c(0, 0.5))
# hist(sample_t, prob = TRUE, nclass = 20, xlim = c(-6, 6), ylim = c(0, 0.5), main =
 "Histogram of Sample t-statistics", xlab = "Sampled t-values")
# hist(sample_t, prob = TRUE, nclass = 50, xlim = c(-6, 6), ylim = c(0, 0.5), main =
 "Histogram of Sample t-statistics", xlab = "Sampled t-values")
hist(sample_t,
     prob = TRUE,
     breaks = seq(-20, 20, by = 0.5),
     xlim = c(-6, 6),
     ylim = c(0, 0.5),
     main = "Histogram of Sample t-statistics",
     xlab = "Sampled t-values")
lines(seq(-6, 6, by = 0.01),
      dt(seq(-6, 6, by = 0.01), df = 3),
      col = "blue")
lines(seq(-6, 6, by = 0.01),
      dnorm(seq(-6, 6, by = 0.01)),
      col = "red")
legend("topright",
       inset = 0.05,
       legend = c("t with df = 3", "standard normal"),
       lty = 1,
       col = c("blue", "red"))
```



QQnorm

`qqnorm()` 을 그려보면 정규분포와 꼬리에서 큰 차이가 난다는 것을 알 수 있음.

```
qqnorm(sample_t)
abline(a = 0, b = 1, col = "blue")
```