Crimtab Data for Simulation of T-Distribution

coop711

2025-10-08

Data Loading

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

```
## [1] "crimtab_2"       "crimtab_df"      "crimtab_long"    "crimtab_long_df"
```

```
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 ...
```

```
head(crimtab_long_df,
     n = 20)
```

```
##    finger height
## 1    10.0     56
## 2    10.3     57
## 3     9.9     58
## 4    10.2     58
## 5    10.2     58
## 6    10.3     58
## 7    10.4     58
## 8    10.7     58
## 9    10.0     59
## 10   10.1     59
## 11   10.2     59
## 12   10.2     59
## 13   10.3     59
## 14   10.3     59
## 15   10.3     59
```

```
## 16   10.4     59
## 17   10.5     59
## 18   10.6     59
## 19   10.7     59
## 20   10.7     59
```

Student 의 Simulation 재현

Sample t-values

3,000장의 카드를 잘 섞는 것은 `sample()` 이용.

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

```
##      finger height
## 51     10.7     60
## 2230   11.9     67
## 2574   12.1     68
## 2423   11.5     68
## 2035   11.3     67
## 1118   11.0     65
## 1201   11.3     65
## 2494   11.7     68
## 1145   11.2     65
## 757    11.0     64
```

각 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 23 ... 750
```

각 표본의 평균과 표준편차 계산에는 `tapply()` 이용.

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

```
##  num [1:750(1d)] 11.6 11.3 11.1 11.4 11.6 ...
##  - attr(*, "dimnames")=List of 1
##   ..$ : chr [1:750] "1" "2" "3" "4" ...
```

```r
str(finger_sample_sd)
```

```
##  num [1:750(1d)] 0.619 0.287 0.15 0.538 0.85 ...
##  - 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)

```r
sample_t <- (finger_sample_mean - mean(crimtab_long_df[, "finger"])) / (finger_sample_sd/sqrt(4))
str(sample_t)
```

```
##  num [1:750(1d)] 0.00851 -1.54836 -5.63156 -0.64108 0.18267 ...
##  - 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)

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

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

```r
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.

```r
mean(sample_t)
```

```
## [1] -0.04779039
```

```r
sd(sample_t)
```

```
## [1] 1.6343
```

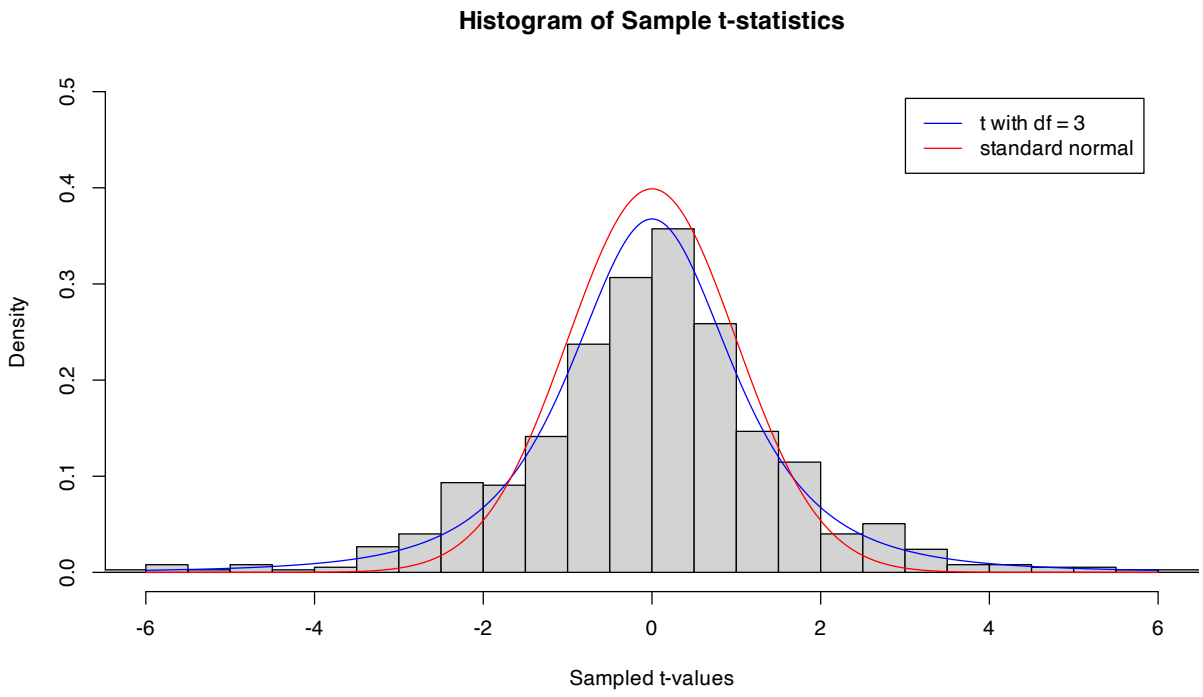
```r
summary(sample_t)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -8.822952 -0.850822  0.008208 -0.047790  0.816831  6.764549
```

Histogram and Density Curves

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

```r
# 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 Samp
# hist(sample_t, prob = TRUE, nclass = 50, xlim = c(-6, 6), ylim = c(0, 0.5), main = "Histogram of Samp
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"))
```

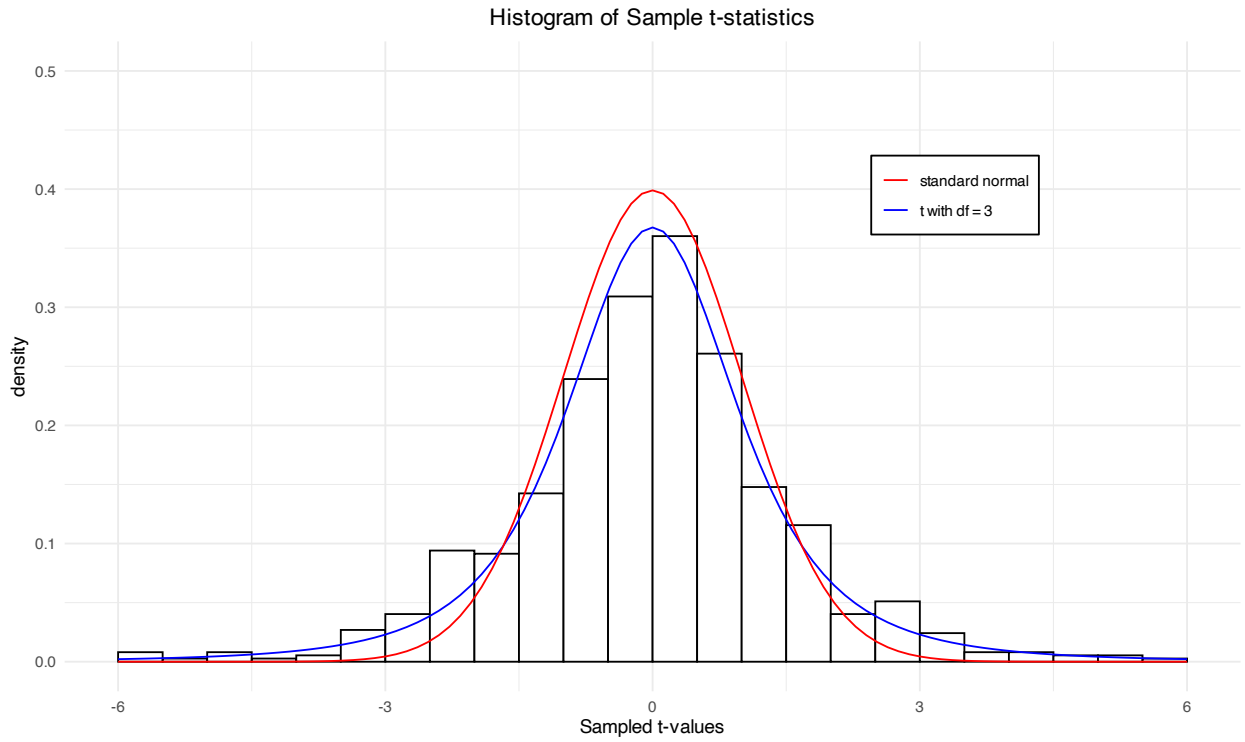


ggplot

```r
library(ggplot2)

#
```

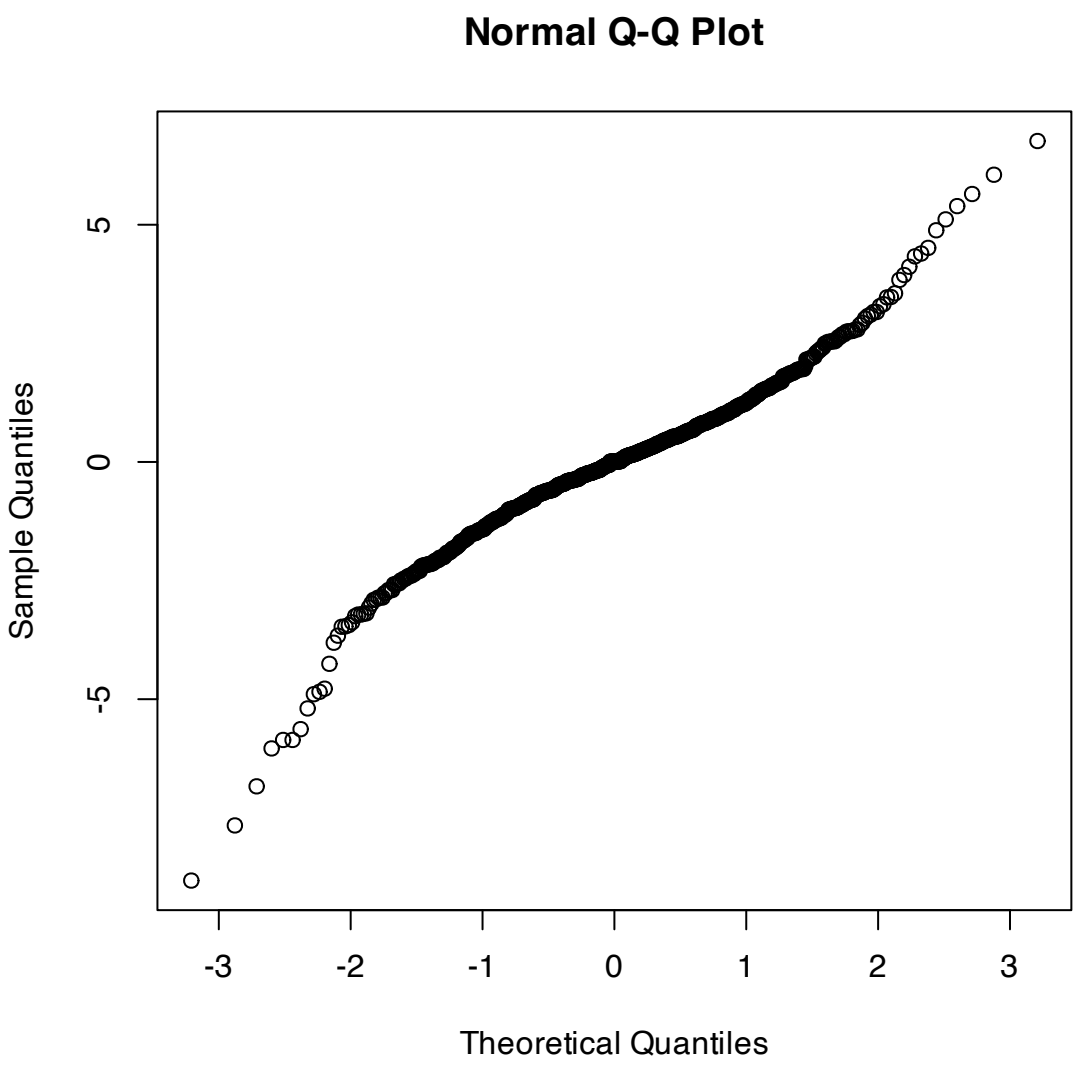
```r
ggplot(data.frame(x = sample_t), aes(x = x)) +
  geom_histogram(aes(y = after_stat(density)),
                 breaks = seq(-20, 20, by = 0.5),
                 color = "black", fill = "white") +
  xlim(c(-6, 6)) +
  ylim(c(0, 0.5)) +
  stat_function(fun = dt, args = list(df = 3), aes(color = "t with df = 3"), linetype = "solid") +
  stat_function(fun = dnorm, aes(color = "standard normal"), linetype = "solid") +
  labs(title = "Histogram of Sample t-statistics",
       x = "Sampled t-values",
       color = NULL) +  #
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5)) +
  scale_color_manual(values = c("t with df = 3" = "blue", "standard normal" = "red")) +
  guides(color = guide_legend(override.aes = list(linetype = c("solid", "solid"), size = 1.5))) +
  theme(legend.position = "inside",
        legend.justification = c(0.8, 0.8), #
        legend.background = element_rect(fill = "white", color = "black"))
```



QQnorm

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

```r
qqnorm(sample_t)
```


Normal Q-Q Plot
Sample Quantiles
Theoretical Quantiles
5
0
-5
-3
-2
-1
0
1
2
3