Homework 2

Joshua Oswari - A14751270

4/20/2019

Problem 1

========== Part 1 ==========

```
x = rnorm(100)
q1 = quantile(x , type = 1)

plot(rep(2,100) + rnorm(100, sd = 0.1), x, xlim = c(0,4),ylim = c(-4,4))
segments(1, q1[1], 1, q1[5])
points(rep(1,5), q1, pch = 20)
q1
```

```
##          0%         25%         50%         75%        100%
## -2.66580302 -0.66909080 -0.03151049  0.55422237  2.60998218
```

```
q2 = quantile(x, type = 2)
segments(3, q2[1], 3, q2[5])
points(rep(3,5), q2, pch = 20)
q2
```

```
##          0%         25%         50%         75%        100%
## -2.66580302 -0.66852235 -0.02242965  0.58998786  2.60998218
```

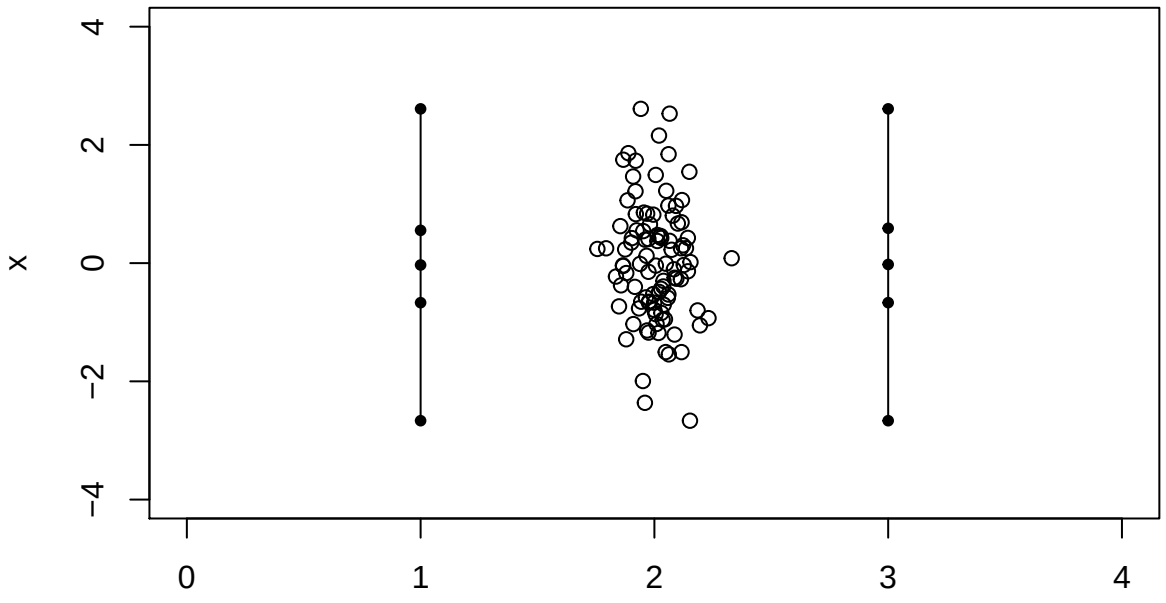
```
q3 = quantile(x, type = 3)
segments(5, q3[1], 5, q3[5])
points(rep(5,5), q2, pch = 20)
q3
```

```
##          0%         25%         50%         75%        100%
## -2.66580302 -0.66909080 -0.03151049  0.55422237  2.60998218
```

```
q4 = quantile(x, type = 4)
segments(7, q3[1], 7, q3[5])
points(rep(7,5), q2, pch = 20)
q4
```

```
##          0%         25%         50%         75%        100%
## -2.66580302 -0.66909080 -0.03151049  0.55422237  2.60998218
```

```
q5 = quantile(x, type = 5)
segments(9, q3[1], 9, q3[5])
points(rep(9,5), q2, pch = 20)
```




```
q5
```

```
##          0%         25%         50%         75%        100%
## -2.66580302 -0.66852235 -0.02242965  0.58998786  2.60998218
```

```
#install.packages("rmutil")
library(rmutil)
```

```
##
## Attaching package: 'rmutil'
```

```
## The following object is masked from 'package:stats':
##
##     nobs
```

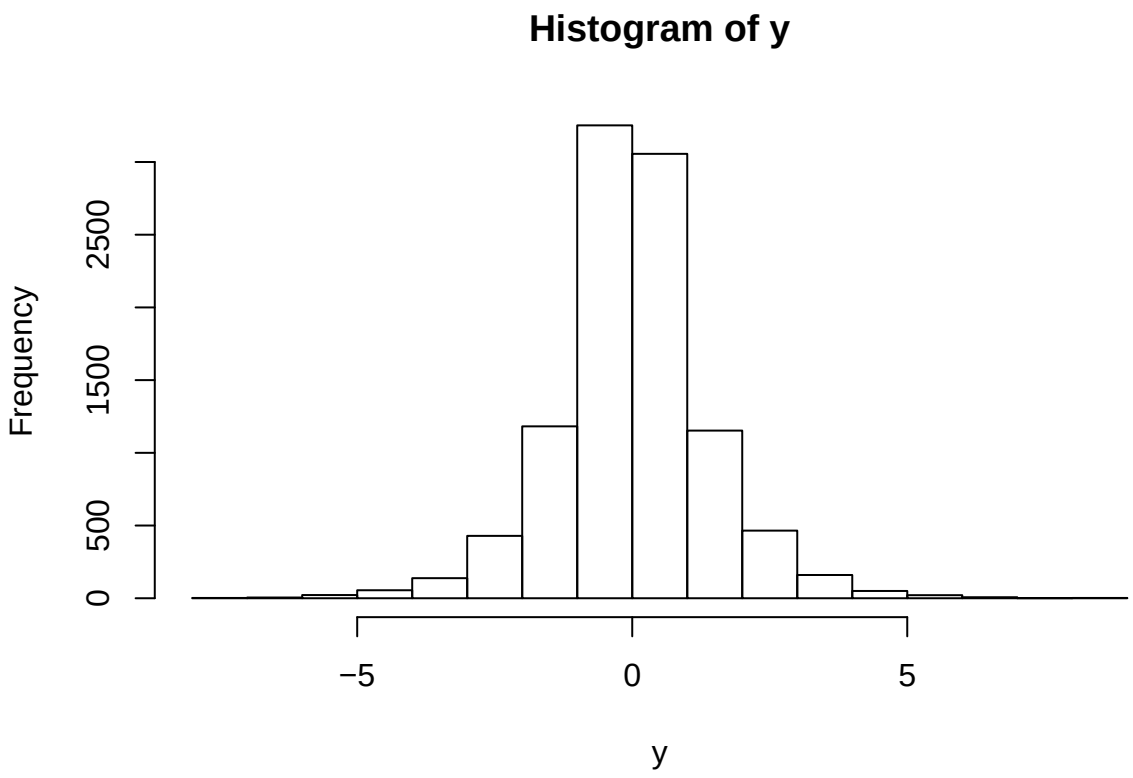
```
y = rlaplace(10000, m = 0, s = 1)
mean(y)
```

```
## [1] 0.001321424
```

```
var(y)
```

```
## [1] 1.945461
```

```
hist(y)
```



Problem 2

========== Part 1 ==========

```
x <- rnorm(100)
out = t.test(x, conf=0.90)
out
```

```
##
##  One Sample t-test
##
## data:  x
## t = -1.3788, df = 99, p-value = 0.1711
## alternative hypothesis: true mean is not equal to 0
## 90 percent confidence interval:
##  -0.31480286  0.02917214
## sample estimates:
##  mean of x
## -0.1428154
```

```
conf.lower = out$conf.int[1]
conf.upper = out$conf.int[2]
```

Bare bones visualization.

```
alpha = 0.10
n = 10
B = 20
mean_vec   = numeric(B)
conf_upper = numeric(B)
```

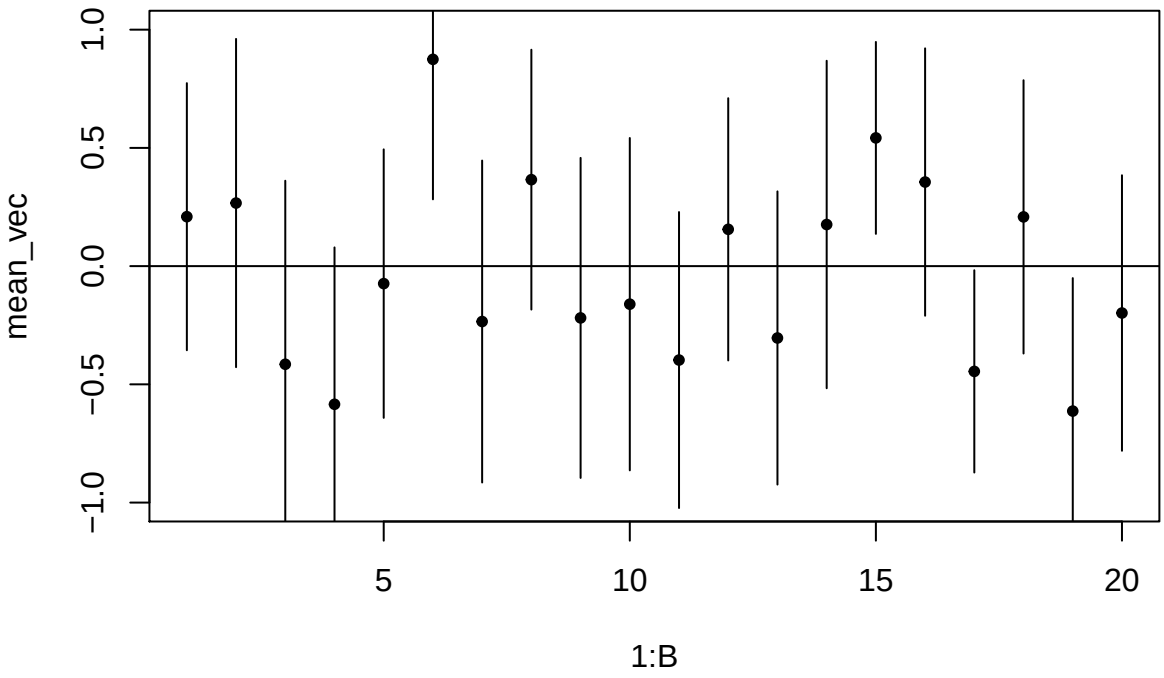
```
conf_lower = numeric(B)

t = 0
for (i in 1:B) {
  x <- rnorm(n, mean=0, sd=1)
  out = t.test(x, conf=1-alpha)
  conf_lower[i] = out$conf.int[1]
  conf_upper[i] = out$conf.int[2]
  mean_vec[i]   = out$estimate

  if ((conf_lower[i] <0) & (conf_upper[i]) >0) {
    t = t+1
  }
}

t
```

```
## [1] 16
```

```
t/B
```

```
## [1] 0.8
```

```
plot(1:B, mean_vec, pch=20, ylim = c(-1, 1))
segments(1:B, conf_lower, 1:B, conf_upper)
abline(h=0)
```



Fancier visualization

```
alpha = 0.10
mu    = 0
sigma = 1
n     = 10
B     = 20
mean_vec   = numeric(B)
```

```r
conf_upper = numeric(B)
conf_lower = numeric(B)

for (i in 1:B) {
  x <- rnorm(n, mean=mu, sd=sigma)
  out = t.test(x, conf=1-alpha)
  conf_lower[i] = out$conf.int[1]
  conf_upper[i] = out$conf.int[2]
  mean_vec[i]   = out$estimate
}

contains_mean = (conf_lower < mu) & (mu < conf_upper)
contains_mean
```

```
##  [1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [15] TRUE TRUE TRUE TRUE TRUE TRUE
```

```r
color_vec     = ifelse(contains_mean == TRUE, 1, 2)
color_vec
```

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

```r
plot(1:B, mean_vec, pch=20, ylim = c(-2, 2), col=color_vec)
segments(1:B, conf_lower, 1:B, conf_upper, col=color_vec)
abline(h=0, lty="dotted")
```