Homework 20

Natalia Baymacheva

31 03 2020

```
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 3.6.2
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 3.6.2
```

```
library(ggpubr)
```

```
## Warning: package 'ggpubr' was built under R version 3.6.2
```

```
## Loading required package: magrittr
```

1.Measures of center 1.0 create own sample or use given vector and write mode, median, mean functions/one-liners

```
x <- c(175, 176, 180, 165, 167, 172, 175, 146, 158, 178)

mode_func <- function(x){
  y <- as.data.frame(sort(table(x), decreasing = T))
  return(as.numeric(as.vector(y[1,1])))
}

median_func <- function(x) {
  x <- sort(x)
  l <- length(x)
  if (l %% 2 != 0) {
    return(x[ceiling(l/2)])
  } else {
    return((x[l/2] + x[l/2+1]) / 2)
  }
```

```
}

mean_func <- function(x) {
  return(sum(x)/length(x))
}
```

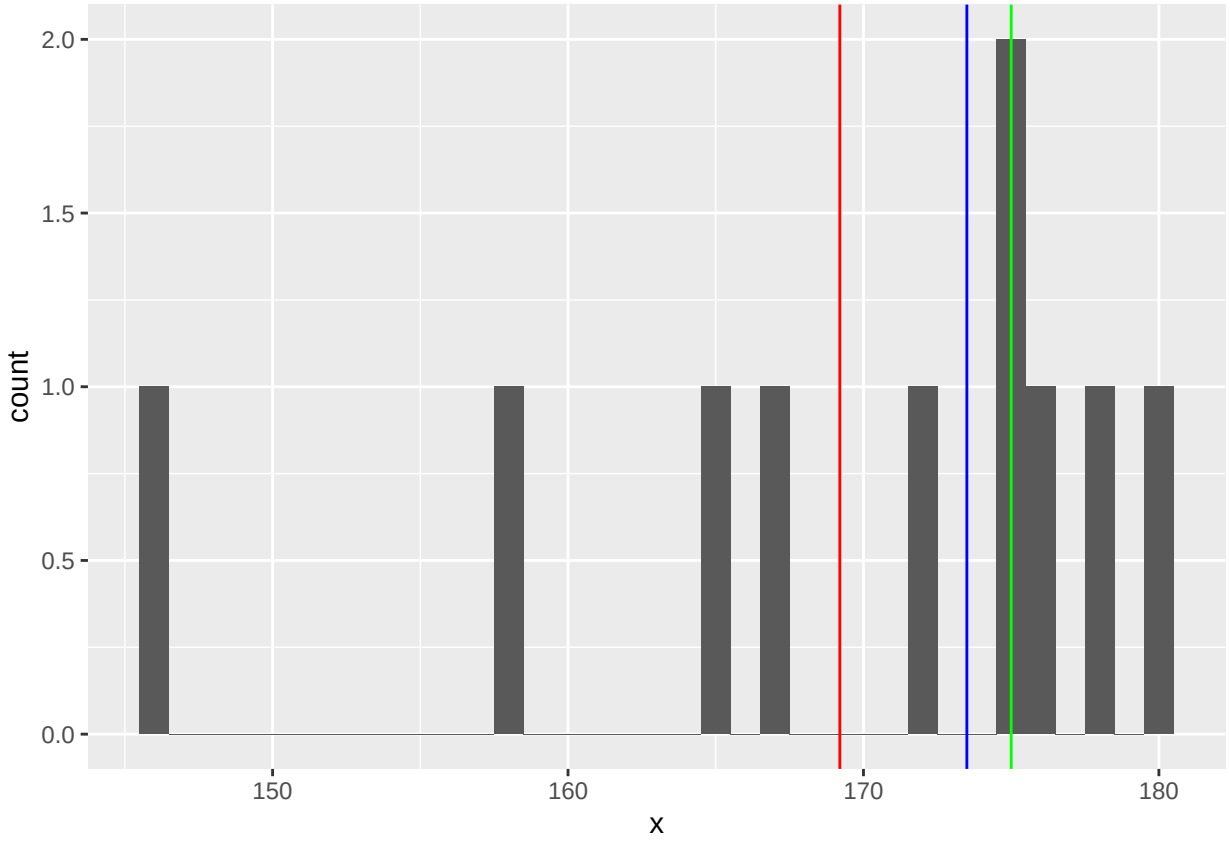
1.1 calculate mode, median and mean for the sample. Compare results for own and built-ins for median and mean

```
functions_table <- function(x) {
  func_names <- c('Mode', 'Median', 'Mean')
  built_in <- c('', median(x), mean(x))
  calculated <- c(mode_func(x), median_func(x), mean_func(x))
  return(data.frame(func_names, built_in, calculated))
}

functions_table(x)
```

```
##   func_names built_in calculated
## 1       Mode               175.0
## 2     Median    173.5      173.5
## 3       Mean    169.2      169.2
```

1.2 visualize histogram with 3 vertical lines for measures of center Histogram + add mode, median, mean

```
ggplot(as.data.frame(x), aes(x = x)) +
  geom_histogram(binwidth = 1) +
  geom_vline(xintercept = mean(x), color = 'red') +
  geom_vline(xintercept = median(x), color = 'blue') +
  geom_vline(xintercept = mode_func(x), color = 'green')
```



1.3 spoil your sample with the outlier - repeat steps 1.1 and 1.2

```
x_spoiled <- c(x, 15)
```

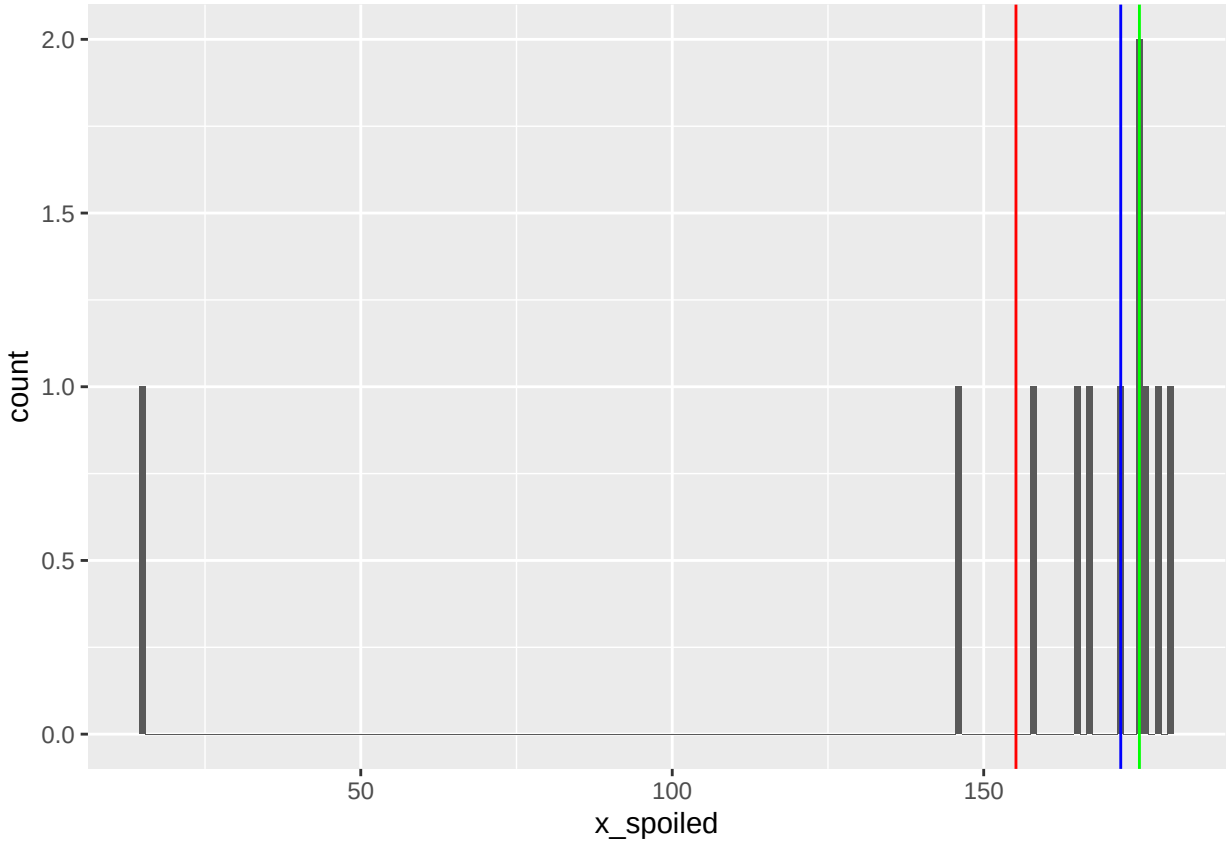
1.3.1. repeat table

```
functions_table(x_spoiled)
```

```
##   func_names         built_in calculated
## 1       Mode                     175.0000
## 2     Median              172   172.0000
## 3       Mean 155.181818181818   155.1818
```

1.3.2 repeat histogram Histogram + add mode, median, mean

```
ggplot(as.data.frame(x_spoiled), aes(x = x_spoiled)) +
  geom_histogram(binwidth = 1) +
  geom_vline(xintercept = mean(x_spoiled), color = 'red') +
  geom_vline(xintercept = median(x_spoiled), color = 'blue') +
  geom_vline(xintercept = mode_func(x_spoiled), color = 'green')
```



2. Measures of spread 2.0 write the functions/one-liners for variance and sd.

```
var_func <- function(x){
  N <- length(x)
  m <- mean(x)
  return(sum((x-m)^2)/N)
}

sd_func <- function(x) {
  return(var_func(x)^(.5))
}

var_sd_table <- function(x) {
  func_names <- c('var', 'sd')
  built_in <- c(var(x), sd(x))
  calculated <- c(var_func(x), sd_func(x))
  return(data.frame(func_names, built_in, calculated))
}
```

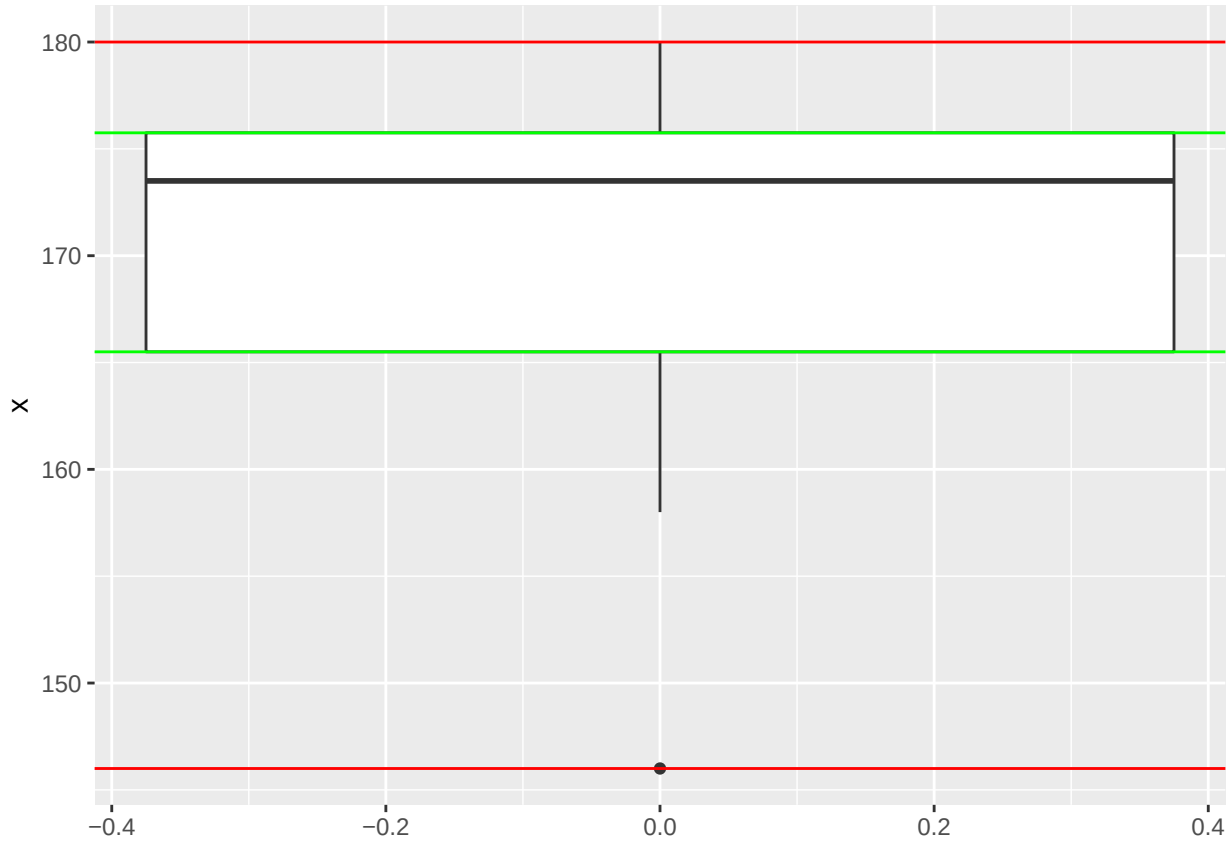
Calculate result, compare with the built-ins

```
var_sd_table(x)
```

```
##   func_names  built_in calculated
## 1        var 111.28889    100.160
## 2         sd  10.54935     10.008
```

2.1 visualize with the box plot. Add horizontal lines for range, IQR, 1-sd borders (use built-ins)

```
ggplot(as.data.frame(x), aes(y = x)) +
  geom_boxplot() +
  geom_hline(yintercept = min(x), color = 'red') +
  geom_hline(yintercept = max(x), color = 'red') +
  geom_hline(yintercept = quantile(x, 3/4), color = 'green') +
  geom_hline(yintercept = quantile(x, 1/4), color = 'green')
```



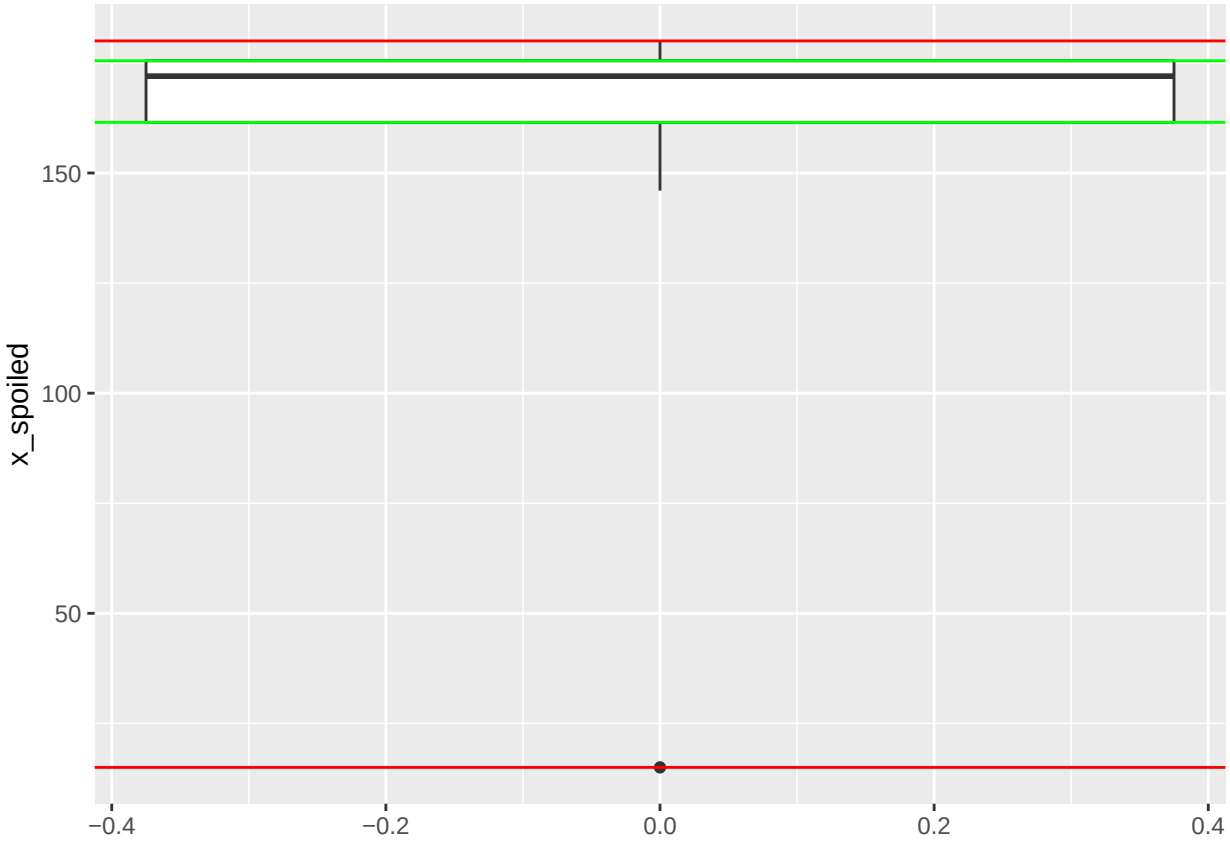
2.2 spoil your sample with the outlier, repeat step 2.1

```
var_sd_table(x_spoiled)
```

```
##   func_names built_in calculated
## 1        var 2261.764 2056.14876
## 2         sd   47.558   45.34478
```

2.2.1 repeat histogramm

```
ggplot(as.data.frame(x_spoiled), aes(y = x_spoiled)) +
  geom_boxplot() +
  geom_hline(yintercept = min(x_spoiled), color = 'red') +
  geom_hline(yintercept = max(x_spoiled), color = 'red') +
  geom_hline(yintercept = quantile(x_spoiled, 3/4), color = 'green') +
  geom_hline(yintercept = quantile(x_spoiled, 1/4), color = 'green')
```



3. Properties mean(x-100) == mean(x) - 100 mean(x / 100) == mean(x) / 100 abs(sum(x - mean(x)) - 0) < 0.000000001 var(x - 100) == var(x) var(x / 100) == var(x) / 10000 sd(x / 100) == sd(x) / 100

3.0 check the properties for mean and sd for your sample 3.1 visualize result tabularly and graphically (maybe with facetting free scales?)

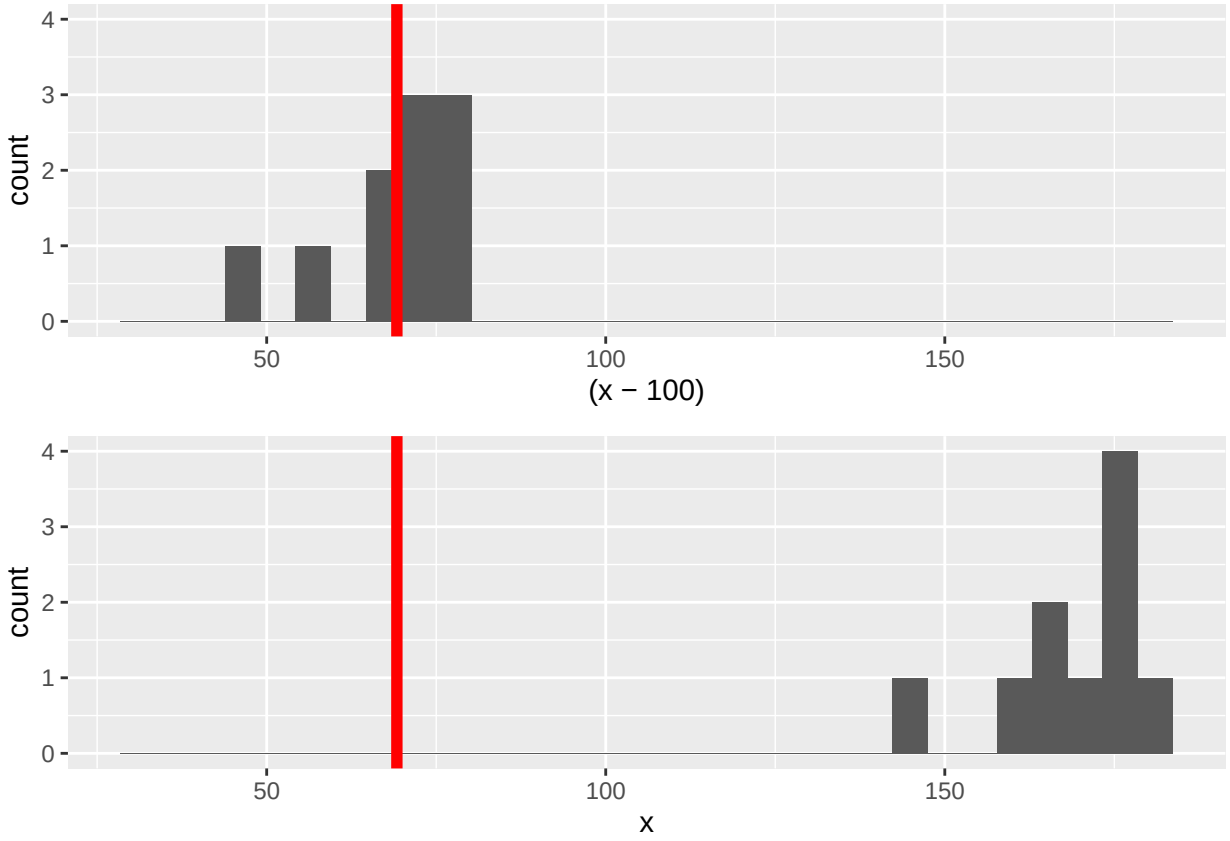
```
property <- c('mean(x-100) = mean(x) - 100',
              'mean(x / 100) = mean(x) / 100',
              'sd(x / 100) = sd(x) / 100')

left <- c(mean(x-100),
          mean(x / 100),
          sd(x / 100))

right <- c(mean(x) - 100,
           mean(x) / 100,
           sd(x) / 100)

properties <- data.frame(property, left, right)

properties
```

```
##                       property       left      right
## 1   mean(x-100) = mean(x) - 100 69.2000000 69.2000000
## 2 mean(x / 100) = mean(x) / 100  1.6920000  1.6920000
## 3     sd(x / 100) = sd(x) / 100  0.1054935  0.1054935
```

```
h_mean_sub <- ggplot(as.data.frame(x-100), aes(x = (x-100))) +
  geom_histogram() +
  geom_vline(xintercept = mean(x-100), color = 'red', size = 2) +
  expand_limits(x = c(30, 180), y = c(0, 4))

H_mean_sub <- ggplot(as.data.frame(x), aes(x = x)) +
  geom_histogram() +
  geom_vline(xintercept = (mean(x) - 100), color = 'red', size = 2) +
  expand_limits(x = c(30, 180), y = c(0, 4))

graphs <- ggarrange(h_mean_sub, H_mean_sub, nrow = 2)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
graphs
```

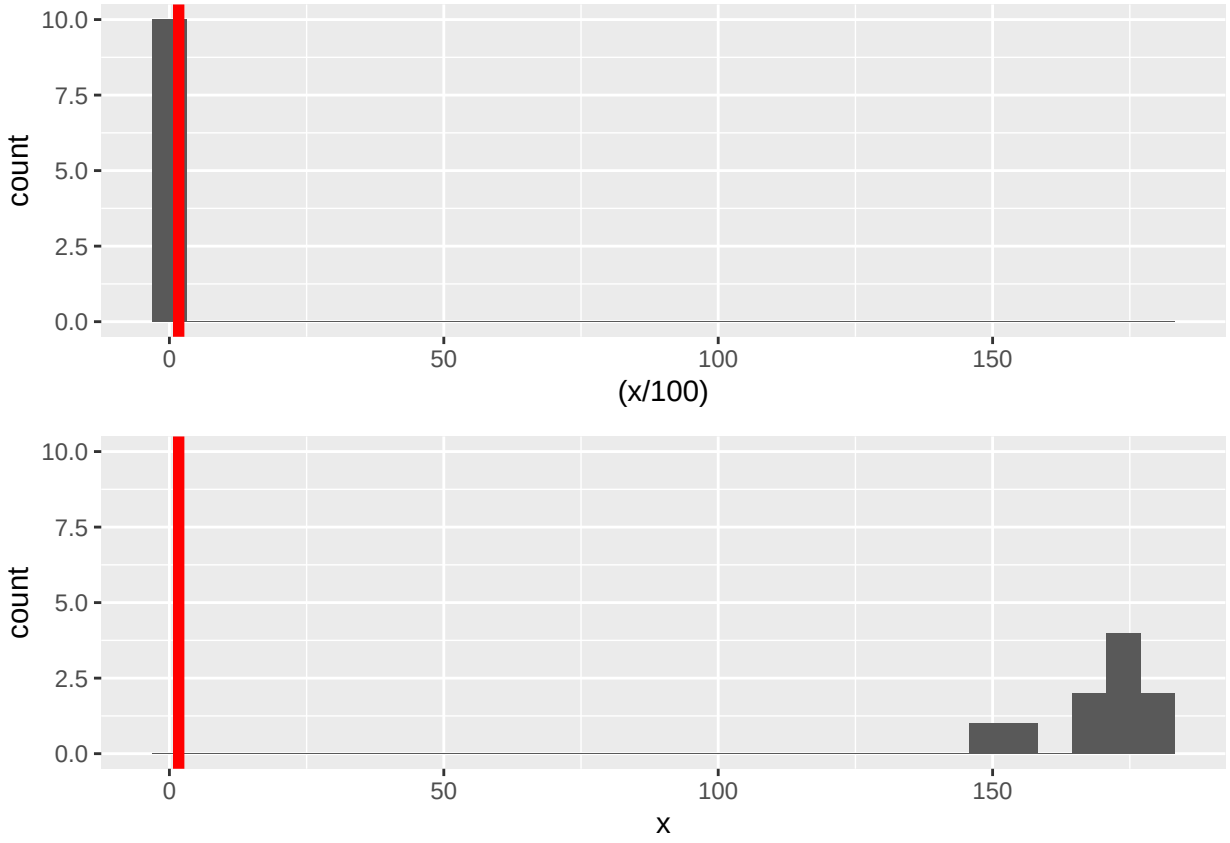



```
h_mean_div <- ggplot(as.data.frame(x/100), aes(x = (x/100))) +
  geom_histogram() +
  geom_vline(xintercept = mean(x/100), color = 'red', size = 2) +
  expand_limits(x = c(0, 180), y = c(0, 10))

H_mean_div <- ggplot(as.data.frame(x), aes(x = x)) +
  geom_histogram() +
```

```
  geom_vline(xintercept = (mean(x)/100), color = 'red', size = 2) +
  expand_limits(x = c(0, 180), y = c(0, 10))

graphs <- ggarrange(h_mean_div, H_mean_div, nrow = 2)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

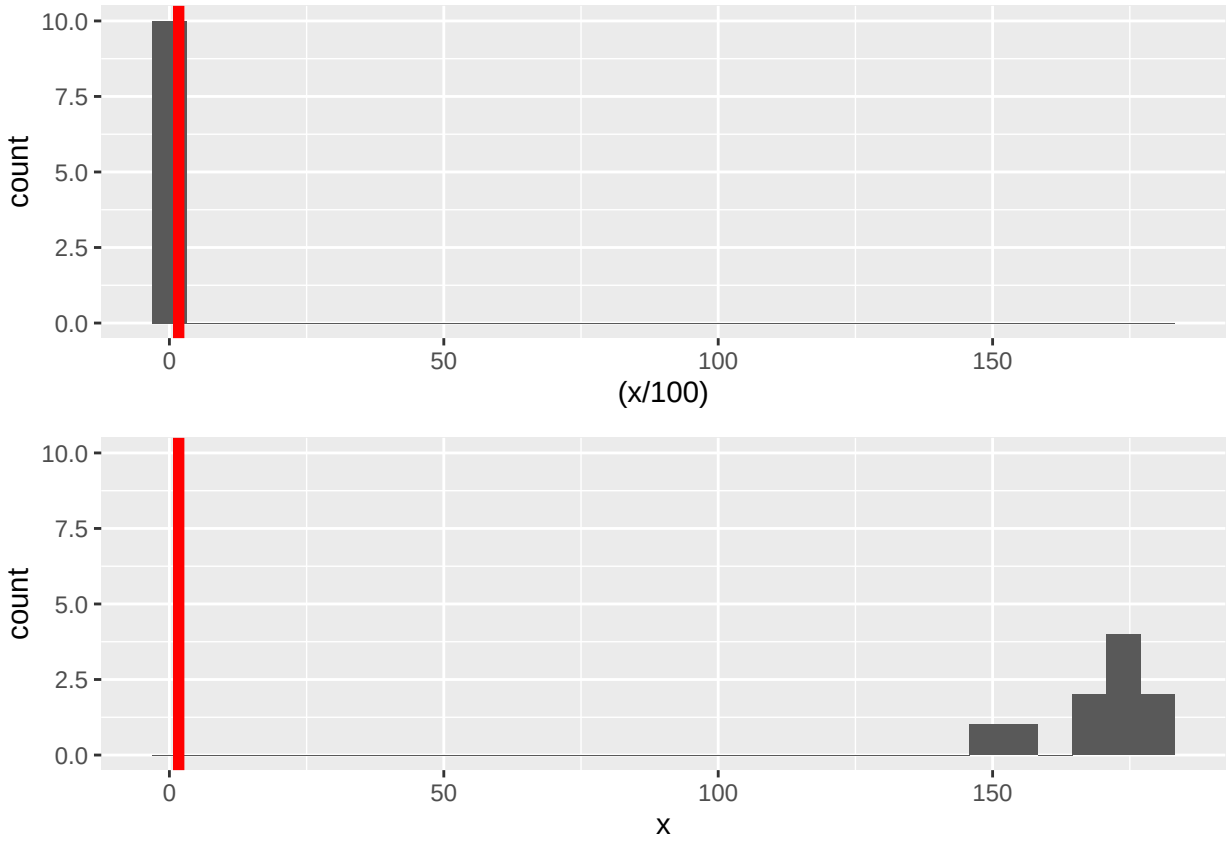
```
graphs
```




```
h_sd <- ggplot(as.data.frame(x/100), aes(x = (x/100))) +
  geom_histogram() +
  geom_vline(xintercept = mean(x/100), color = 'red', size = 2) +
  expand_limits(x = c(0, 180), y = c(0, 10))

H_sd <- ggplot(as.data.frame(x), aes(x = x)) +
  geom_histogram() +
  geom_vline(xintercept = (mean(x)/100), color = 'red', size = 2) +
  expand_limits(x = c(0, 180), y = c(0, 10))

graphs <- ggarrange(h_sd, H_sd, nrow = 2)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
graphs
```



4. Normal distribution pnorm()

4.0 for the population N(175, 10) find the probability to be: less than 156cm,

```
pnorm(156, mean = 175, sd = 10)
```

```
## [1] 0.02871656
```

more than 198,

```
pnorm(198, mean = 175, sd = 10, lower.tail = F)
```

```
## [1] 0.01072411
```

between 168 and 172 cm

```
pnorm(172, mean = 175, sd = 10) - pnorm(168, mean = 175, sd = 10)
```

```
## [1] 0.1401249
```

Standard normal distribution 4.1 check the properties of 1-2-3-sd's (68-95-99,7 rule) for standard normal distribution using pnorm()

```
pnorm(1) - pnorm(-1)
```

```
## [1] 0.6826895
```

```
pnorm(2) - pnorm(-2)
```

```
## [1] 0.9544997
```

```
pnorm(3) - pnorm(-3)
```

```
## [1] 0.9973002
```

Standardization 4.2 generate sample using rnorm() from N(175, 10), find mean and sd;

```
s <- rnorm(10000, mean = 175, sd = 10)
print(c('mean is', mean(s)))
```

```
## [1] "mean is"          "175.210931097638"
```

```
print(c('sd is', sd(s)))
```

```
## [1] "sd is"            "10.0159957214468"
```

4.3 standardize, find the same (x-mean)/sd

```
cols <- c('function', 'built-ins', 'calculated')
s_stand <- (s-mean(s))/sd(s)
s_stand_calc <- (s-175)/10
functions <- c('mean', 'sd')
built_ins <- c(mean(s_stand), sd(s_stand))
calculated <- c(mean(s_stand_calc), sd(s_stand_calc))
data.frame(functions, built_ins, calculated)
```

```
##   functions     built_ins calculated
## 1      mean -3.072293e-16 0.02109311
## 2        sd  1.000000e+00 1.00159957
```

5.0 Generate large population (n ~ 100 000 - 1 000 000) distributed as N(0, 1)

```
population <- rnorm(100000, mean = 0, sd = 1)
```

Sample from population k observations for 30 times - you will have set of 30 samples. 5.1 k = 10 5.2 k = 50 5.3 k = 100 5.4 k = 500

```r
k_10 <- data.frame(replicate(30, sample(population, 10)))
k_50 <- data.frame(replicate(30, sample(population, 50)))
k_100 <- data.frame(replicate(30, sample(population, 100)))
k_500 <- data.frame(replicate(30, sample(population, 500)))
```

For each sample calculate mean.

```r
k_10$Mean <- rowMeans(k_10)
k_50$Mean <- rowMeans(k_50)
k_100$Mean <- rowMeans(k_100)
k_500$Mean <- rowMeans(k_500)
```

For the set calculate mean of means, sd of means, SE. Create table with k, mean of means, sd of means, SE.

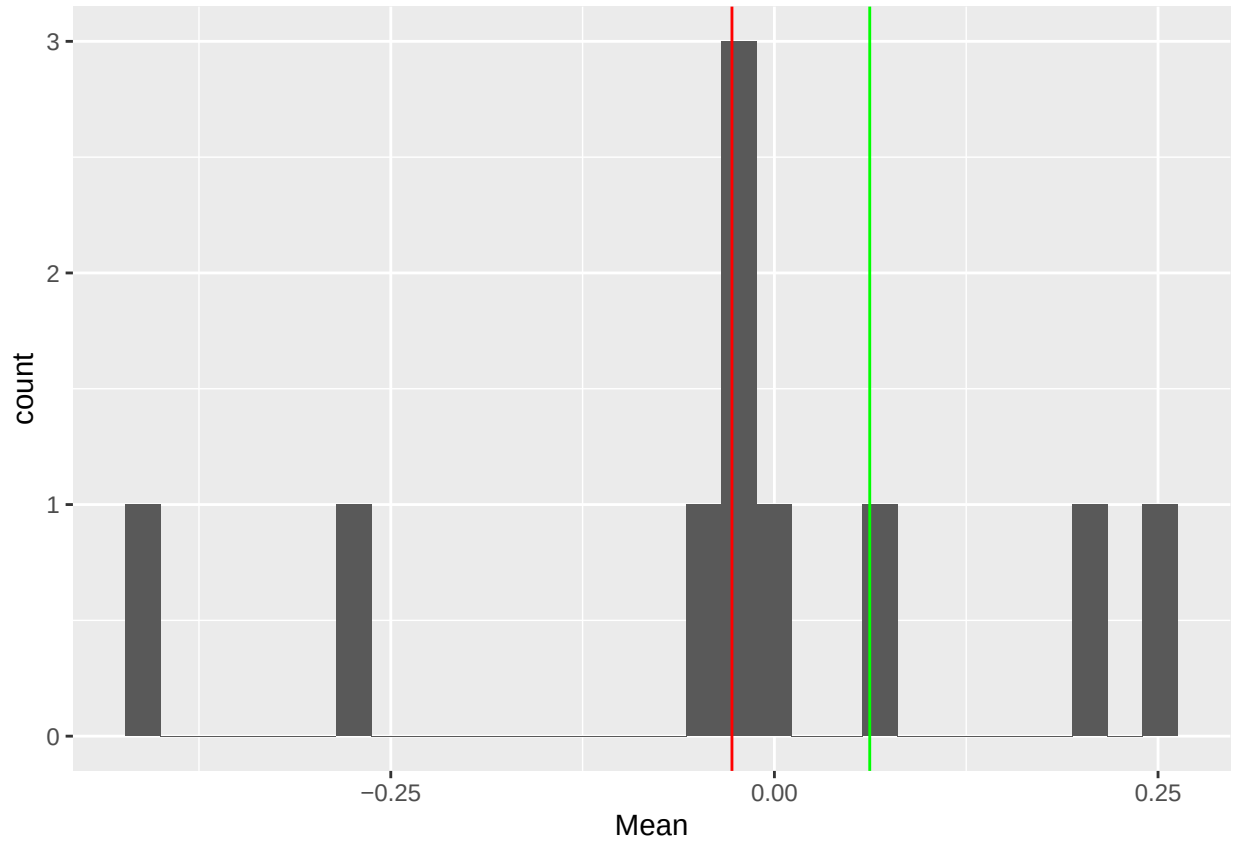
```r
k <- c(10, 50, 100, 500)
mean_of_means <- c(mean(k_10$Mean), mean(k_50$Mean), mean(k_100$Mean), mean(k_500$Mean))
sd_of_means <- c(sd(k_10$Mean), sd(k_50$Mean), sd(k_100$Mean), sd(k_500$Mean))
SE <- sd_of_means/sqrt(k)
data.frame(k, mean_of_means, sd_of_means, SE)
```

```
##     k mean_of_means sd_of_means          SE
## 1  10  -0.027667114   0.1964411 0.062120134
## 2  50  -0.076501347   0.1645574 0.023271927
## 3 100  -0.025050645   0.1759127 0.017591265
## 4 500   0.006294827   0.1919357 0.008583624
```

Visualize distribution of means with histogram and lines for mean of means and SE. 5.5 Compare results

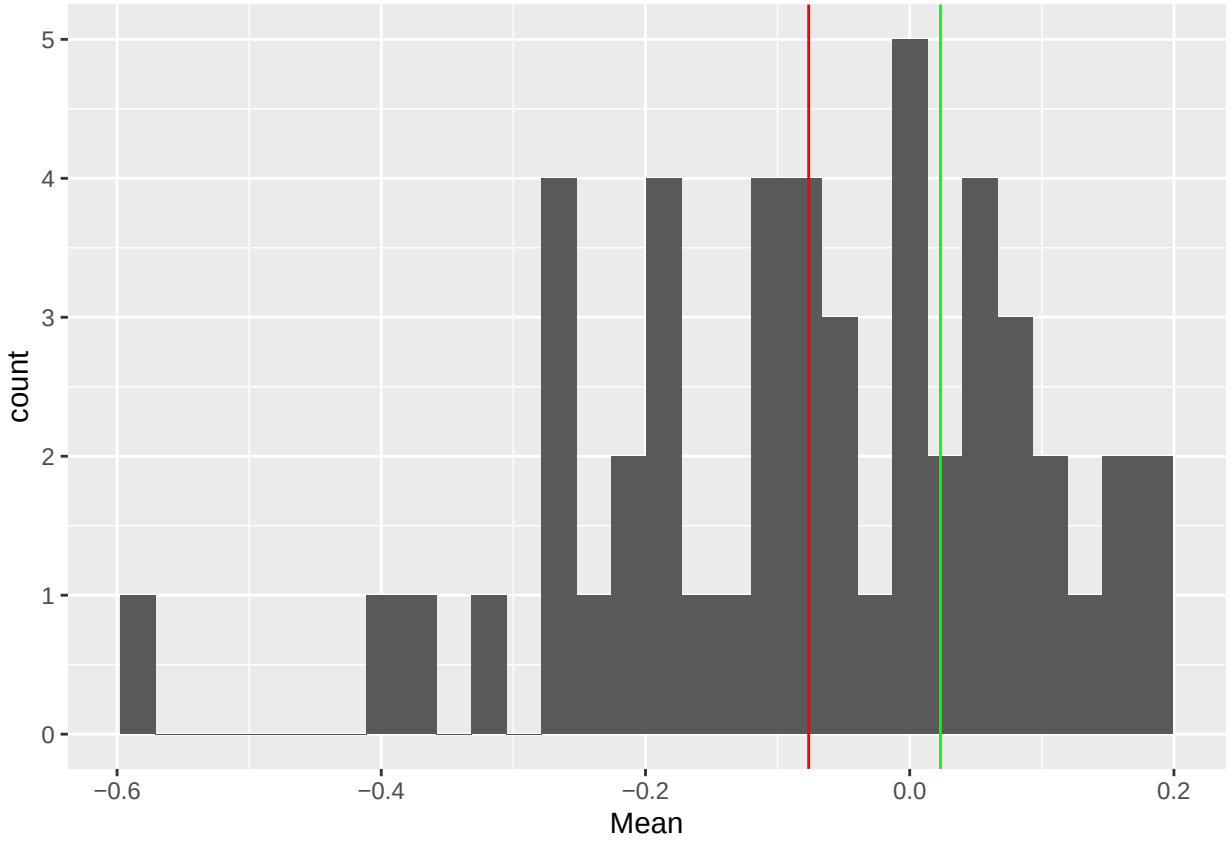
```r
ggplot(k_10, aes(x = Mean)) +
  geom_histogram() +
  geom_vline(xintercept = mean_of_means[1], color = 'red') +
  geom_vline(xintercept = SE[1], color = 'green')
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



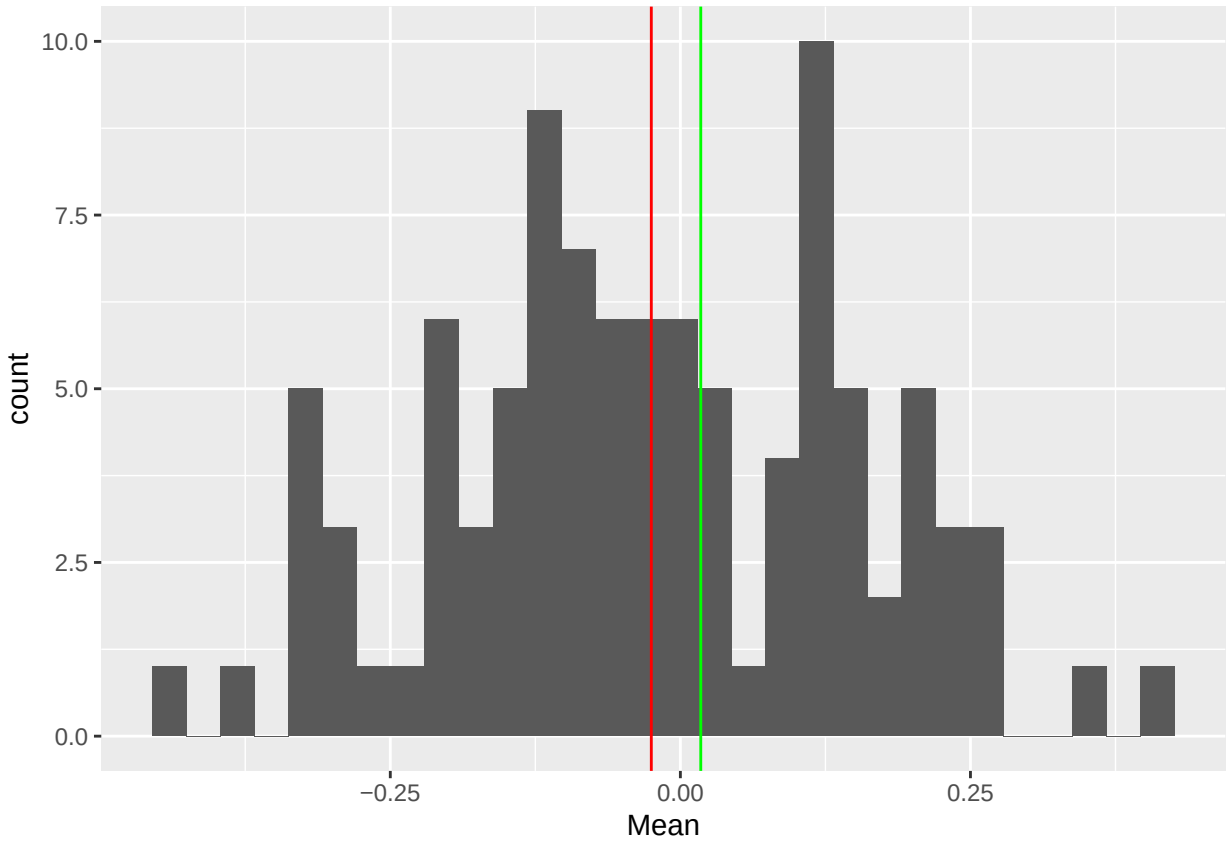

```
ggplot(k_50, aes(x = Mean)) +
  geom_histogram() +
  geom_vline(xintercept = mean_of_means[2], color = 'red') +
  geom_vline(xintercept = SE[2], color = 'green')
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



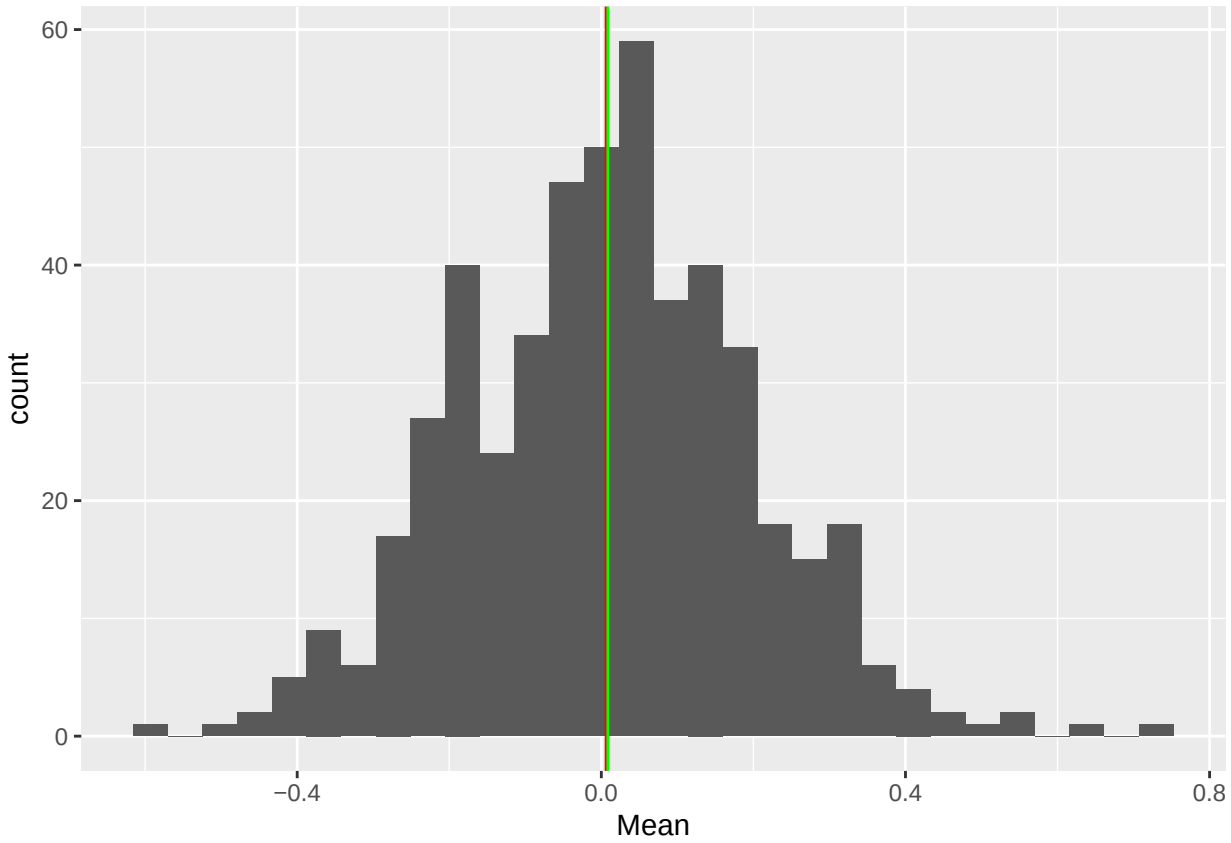

```
ggplot(k_100, aes(x = Mean)) +
  geom_histogram() +
  geom_vline(xintercept = mean_of_means[3], color = 'red') +
  geom_vline(xintercept = SE[3], color = 'green')
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
ggplot(k_500, aes(x = Mean)) +
  geom_histogram() +
  geom_vline(xintercept = mean_of_means[4], color = 'red') +
  geom_vline(xintercept = SE[4], color = 'green')
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```


60
40
20
0
count
−0.4
0.0
0.4
0.8
Mean