q1a.R

mirrien

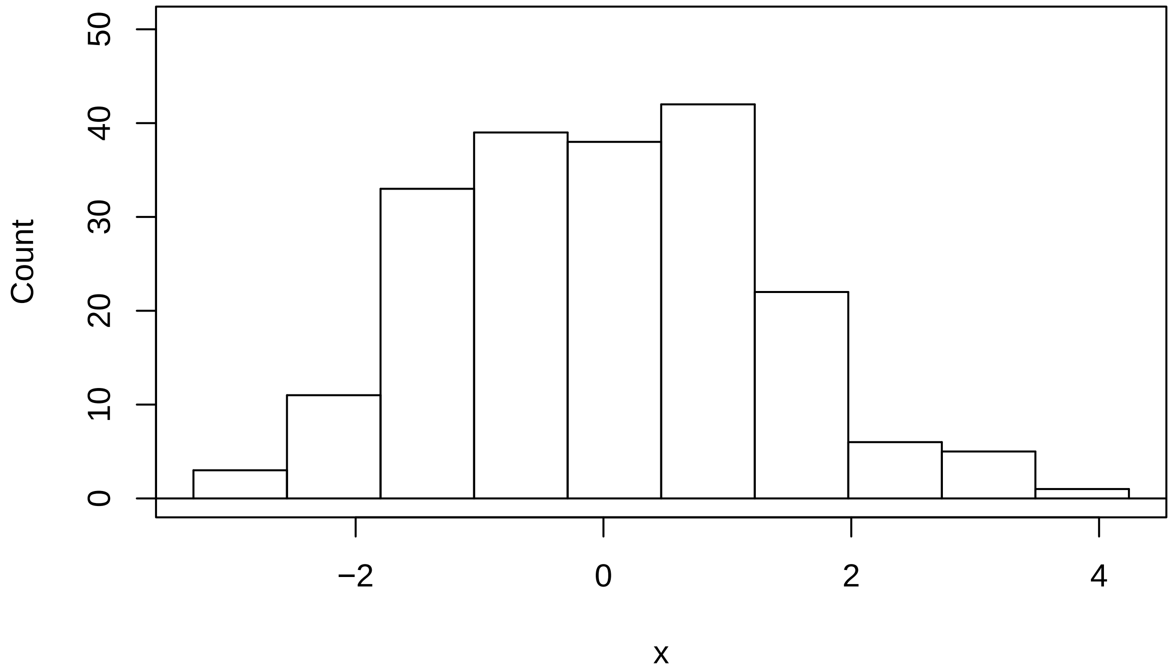
2022-02-07

```
library(tidyverse)
```

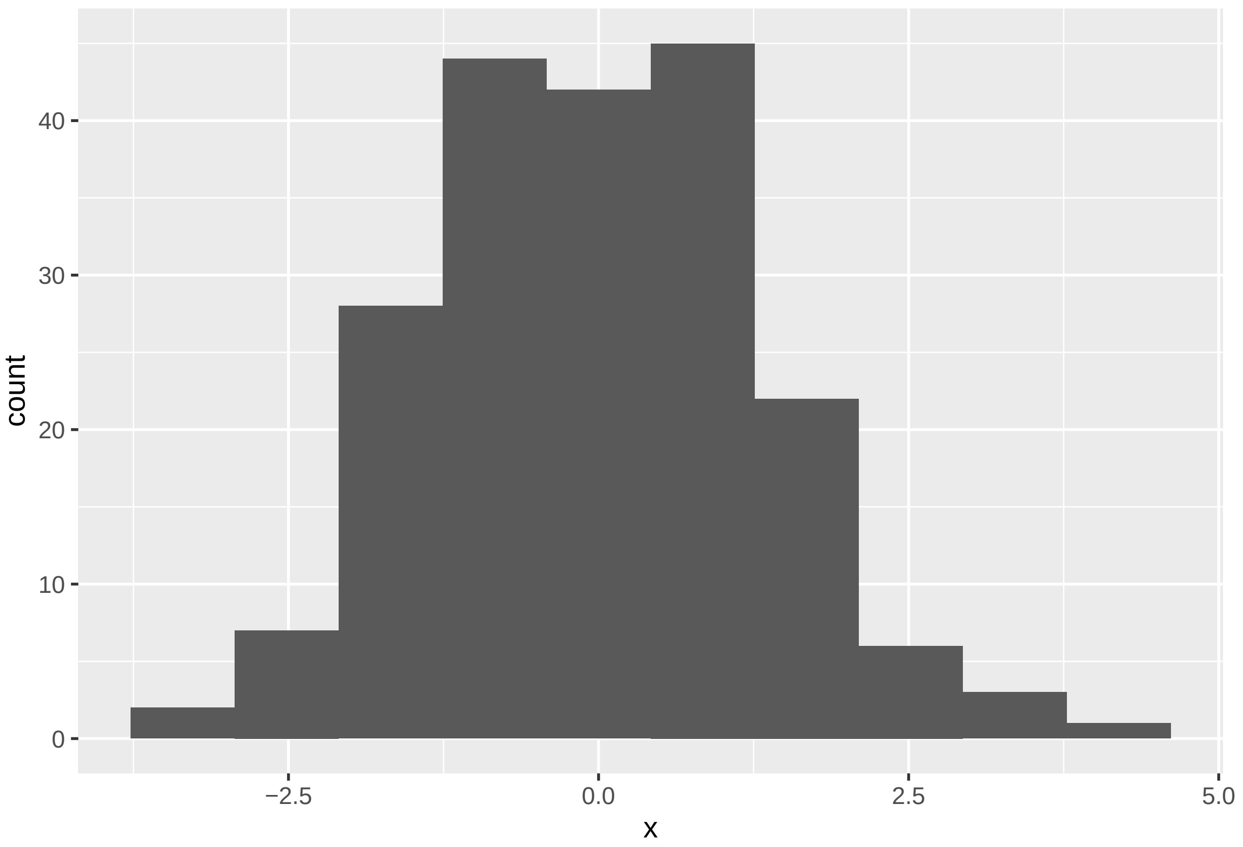
```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.4     v dplyr   1.0.7
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   2.0.1     v forcats 0.5.1

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

```
#################################################

# Q1a

counts <- function(x,n){

  v <- c() # Empty vector to store the divided checkpoints b_n
  d <- (max(x)-min(x))/n # The length of one interval

  for (i in (0:n)){
    v[i+1] <- min(x)+i*d
    # e.g., v[1]=min(x), v[2] = b_1 = min(x)+1*d, v[3] = b_2 = min(x)+2*d
    # v[n+1] = b_n = min(x) + n*d = min(x) + [max(x) - min(x)] = max(x)
  }

  ret <- vector(mode = "numeric", length = n)
  # Empty vector to store counters that count # of elements that lie in
  # each interval
  # e.g., ret[j] stores # of elements lie in the j-th interval

  for (j in (1:length(x))) { # j is the counter that loops elements in x

    i=1 # initiate and re-initiate counter i that loops elements in v

    while(!between(x[j],v[i],v[i+1])){i=i+1}
    # as long as x[j] is not the i-th interval, i++ and examine next interval
    # if x[j] is found in the i-th interval, the corresponding i-th counter in
```

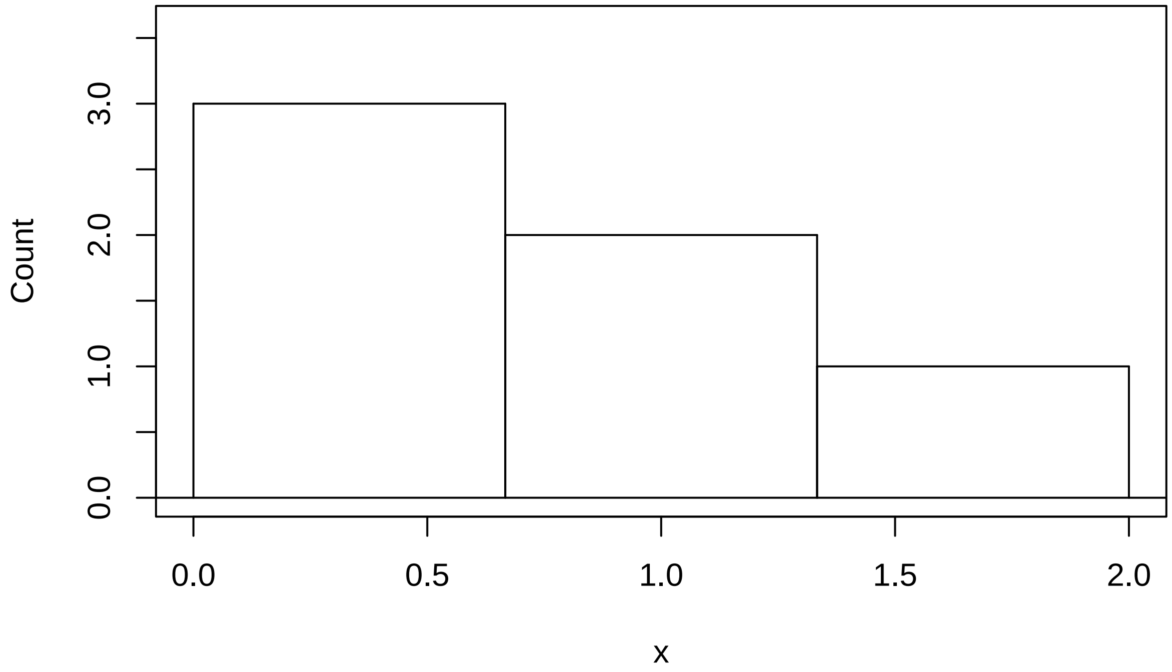
```r
    # ret will increase by 1.
    ret[i]=ret[i]+1

  }

  # once every element in x has been determined which interval it belongs,
  # return ret
  return(ret)
}


# test case (PASSED):

# x <- seq(1,10)
# counts(x,3) == c(4,3,3)
# # [1,4], (4,7], (7,10]
# counts(x,2) == c(5,5)
# # [1,5.5], (5.5,10]
#
# set.seed(123)
# y <- rnorm(10,10,5)
# y
# range(y) # [3.674694,18.575325]
#
#
# counts(y,2) == c(7,3)
# # [3.674694,11.1250095], (11.1250095,18.575325]



###################################################

# Q1b

histo <- function(x,n){

  d <- (max(x)-min(x))/n # The length of one interval (width of bars)
  res <- counts(x,n) # number of counts in each interval (height of bars)

  # Empty plot
  plot(1,
       type = 'n',
       xlab = 'x',
       ylab = 'Count',
       xlim = range(x),
       ylim = c(0,(1.2*max(res)))
  )

  # x-axis
  abline(h=0)


  # draw bars
  for (i in (1:n)) {
    segments(min(x)+(i-1)*d,0,min(x)+(i-1)*d,res[i]) # left line
    segments(min(x)+i*d,0,min(x)+i*d,res[i]) # right line
    segments(min(x)+(i-1)*d,res[i],min(x)+i*d,res[i]) # top line
  }

}


# histo(x,3)


##################################################

# Q1c

set.seed(123)

z <- c(rnorm(100,-1,1),rnorm(100,1,1))

histo(z,10)
```



```
# Compared to the graph made via geom_histogram:
data <- data.frame(x=z)
ggplot(data,aes(x)) + geom_histogram(bins=10)
```



```
#################################################

# Q1d


histo(c(0,0,0,1,1,2),3)
```



```
# ggplot(data.frame(x=c(0,0,0,1,1,2)),aes(x))
#     + geom_histogram(bins = 10)
```