Assignment 2

Fredrik Nyström

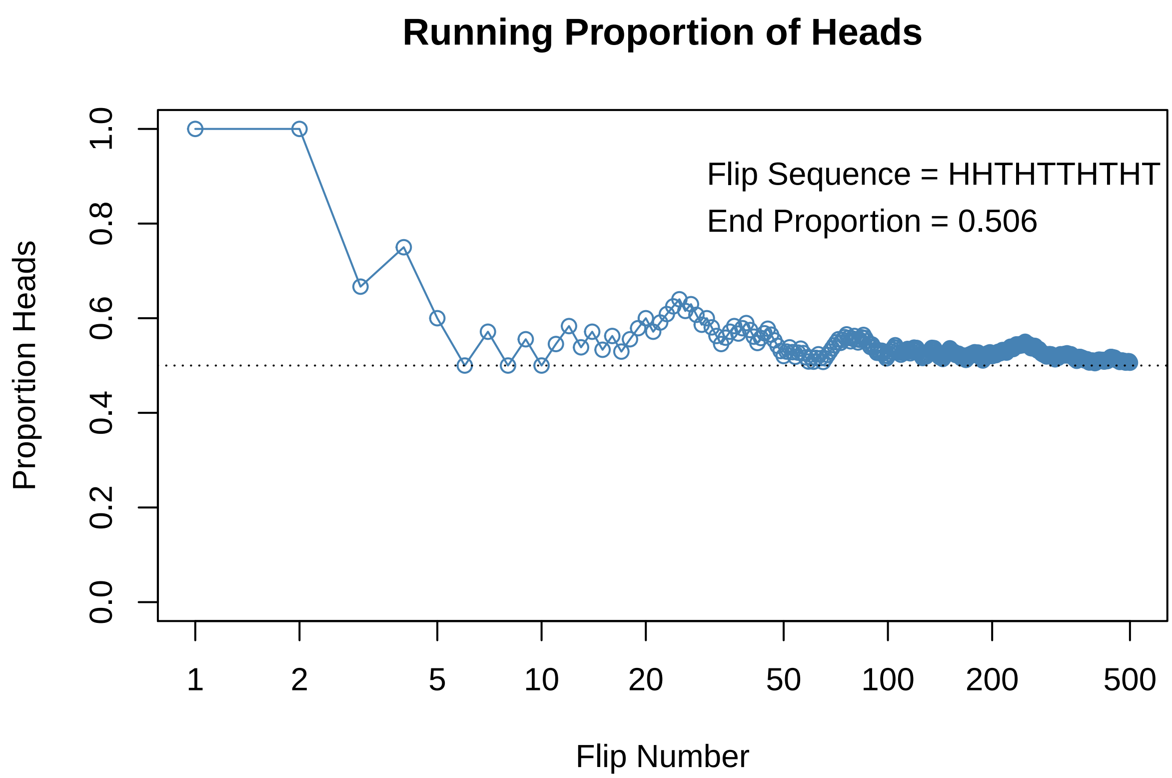
2019-09-11

Task A

1. Discrete random variable

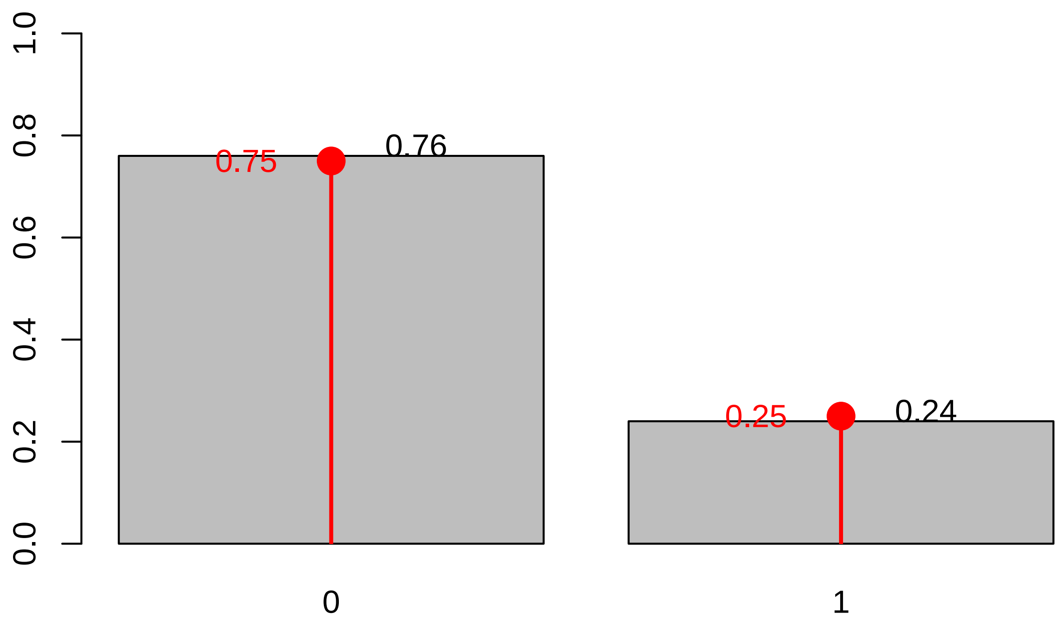
Recreate the 4.1 figure from the book simulating coin tosses.

```
N <- 500

coin_toss <- function(n) {
  fair <- function(theta) { if (theta < 0.5) 1 else 0 }

  sapply(runif(n), fair)
}


toss_data <- coin_toss(n = N)
running_proportion <- cumsum(toss_data)/(1:N)

plot(x = 1:N, y = running_proportion, type = "o", col = "steelblue", log = "x",
     ylim = c(0.0, 1.0),
     main = "Running Proportion of Heads",
     ylab = "Proportion Heads", xlab = "Flip Number")
abline(h = 0.5, lty = "dotted")

letter_sequence <- paste(c("T", "H")[toss_data[1:10] + 1], collapse = "")
sequence_text <- paste("Flip Sequence = ", letter_sequence, " ... ", sep = "")

text(x = 30, y = 0.9, sequence_text, adj = 0)
text(x = 30, y = 0.8, paste("End Proportion =", running_proportion[N]), adj = 0)
```



Modify the function so that the coin is biased with $\theta = 0.25$. Sample 100 biased coin tosses and plot a histogram with the true PMF overlaid.

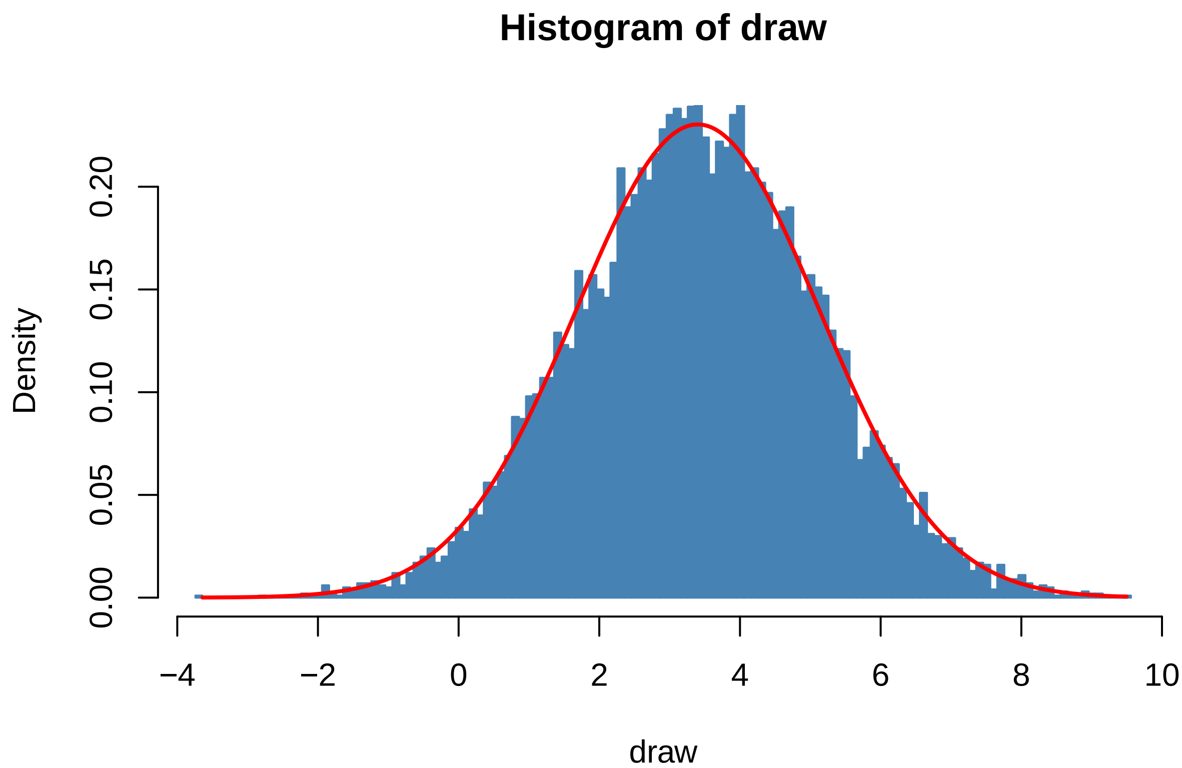
```
biased_coin_toss <- function(n) {
  bias <- function(theta) { if (theta < 0.25) 1 else 0 }

  sapply(runif(n), bias)
}

bias_toss_data <- biased_coin_toss(n = 100)
#Calculate frequency table
bias_toss_data <- table(bias_toss_data)/length(bias_toss_data)
bp <- barplot(bias_toss_data, ylim = c(0, 1))
text(x = bp + 0.2, y = bias_toss_data + 0.02, labels = bias_toss_data)

# Probability mass calculated from the binomial distribution function
pmf <- dbinom(x = 0:1, size = 1, prob = 0.25)
lines(x = bp, y = pmf, type = "h", col = "red", lwd = 2)
points(x = bp, y = pmf, col = "red", pch = 16, lwd = 2, cex = 2)
text(x = bp - 0.2, y = pmf, col = "red", labels = pmf)
```



2. Continuous random

Make 10000 draws from $\mathcal{N}(\mu = 3.4, \sigma^2 = 3)$. Plot them as a normalized frequency histogram and overlay the normal PDF.

```r
pdf_normal <- function(x, mu = 3.4, sigma_sq = 3) {
  (1 / sqrt(2 * pi * sigma_sq)) * exp(-(x - mu)^2/(2 * sigma_sq))
}

draw <- rnorm(n = 1e4, mean = 3.4, sd = sqrt(3))

bin_breaks <- seq(from = min(draw) - 0.1, to = max(draw) + 0.1, by = 0.1)

hist(draw, breaks = bin_breaks, freq = FALSE, ylim = c(0, pdf_normal(3.4)),
     col = "steelblue", border = "steelblue")
curve(pdf_normal, from = min(draw), to = max(draw), n = 1e4, add = TRUE, col = "red", lwd = 2)
```



Calculate the expected value and the variance of x using a Riemann sum.

```
dx <- 0.1
xs <- seq(from = -10, to = 20, by = dx)

ex <- sum(pdf_normal(xs) * xs * dx)
vx <- sum(pdf_normal(xs) * (xs - ex)^2 * dx)
```

```
ex
```

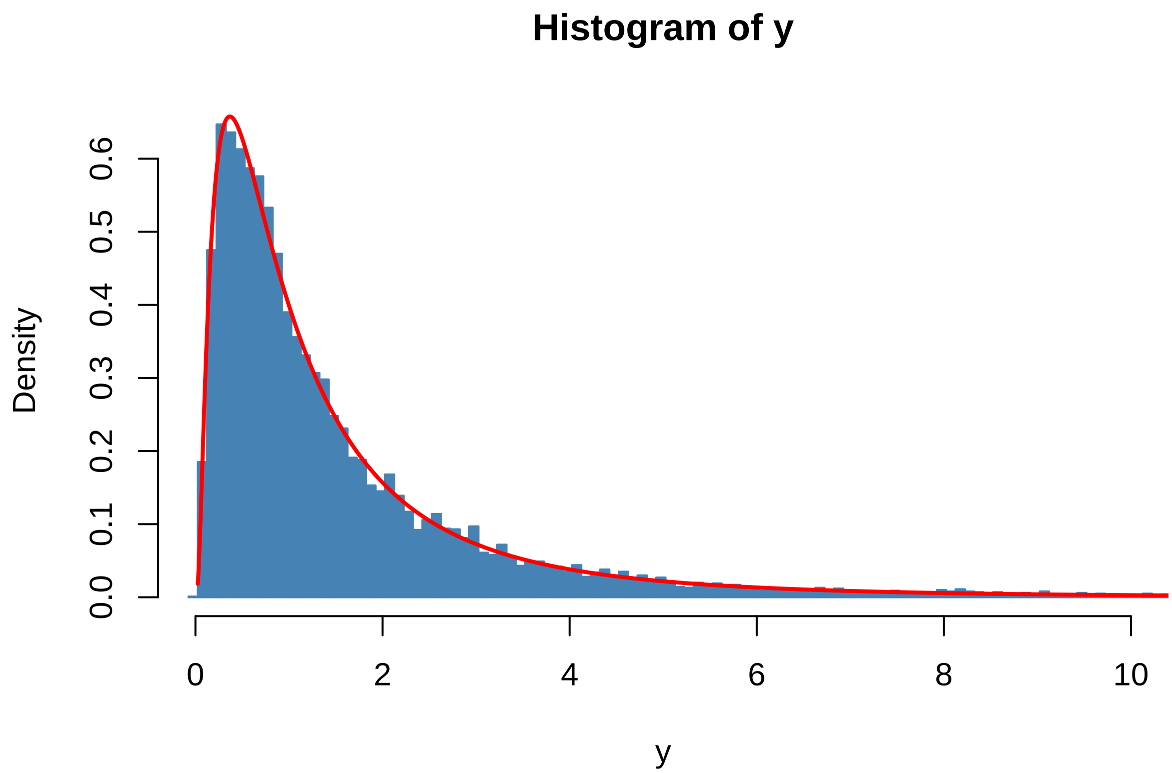
```
## [1] 3.4
```

```
vx
```

```
## [1] 3
```

Plot the histogram and PDF of $y = \exp x$ where $x \sim \mathcal{N}(\mu = 0, \sigma^2 = 1)$.

```
pdf_lognormal <- function(x, mu = 0, sigma_sq = 1) {
  (1 / (x * sqrt(2 * pi * sigma_sq))) * exp(-(log(x - mu))^2/(2 * sigma_sq))
}

x <- rnorm(n = 1e4)
y <- exp(x)

bin_breaks <- seq(from = min(y) - 0.1, to = max(y) + 0.1, by = 0.1)

hist(y, breaks = bin_breaks, freq = FALSE, xlim = c(0, 10), col = "steelblue", border = "steelblue")
curve(pdf_lognormal, from = min(y), to = max(y), n = 1e4, add = TRUE, col = "red", lwd = 2)
```



Find the mode using samples $z = p(y, \mu, \sigma)$, and using optimization.

```
z <- pdf_lognormal(y)

i <- which.max(z) # which(z == max(z))
z[i] # max(z)
```

```
## [1] 0.6577446
```

```
zmax <- optimize(pdf_lognormal, interval = c(0, 1e4), maximum = TRUE)
zmax$objective
```

```
## [1] 0.6577446
```

Task B

Reading a CSV file (long formated), pivoting it to a wider format. Recalculate the data from frequency data to proportion.

```
library(here)
library(tidyverse)

HEC <- read_csv(here("data", "HairEyeColor.csv"))

HEC
```

```
## # A tibble: 16 x 3
##    Hair  Eye   Count
##    <chr> <chr> <dbl>
```

```
##   1 Black Blue     20
##   2 Black Brown    68
##   3 Black Green     5
##   4 Black Hazel    15
##   5 Blond Blue     94
##   6 Blond Brown     7
##   7 Blond Green    16
##   8 Blond Hazel    10
##   9 Brown Blue     84
##  10 Brown Brown   119
##  11 Brown Green    29
##  12 Brown Hazel    54
##  13 Red   Blue     17
##  14 Red   Brown    26
##  15 Red   Green    14
##  16 Red   Hazel    14
```

```r
HEC <- HEC %>% pivot_wider(names_from = Hair, values_from = Count, names_prefix = "Hair.") %>%
  column_to_rownames(var = "Eye")
rownames(HEC) <- paste0("Eye.", rownames(HEC))

HEC
```

```
##           Hair.Black Hair.Blond Hair.Brown Hair.Red
## Eye.Blue          20         94         84       17
## Eye.Brown         68          7        119       26
## Eye.Green          5         16         29       14
## Eye.Hazel         15         10         54       14
```

```r
HEC <- HEC/sum(HEC)
```

Verify the correctness.

```r
HEC
```

```
##            Hair.Black Hair.Blond Hair.Brown   Hair.Red
## Eye.Blue  0.033783784 0.15878378 0.14189189 0.02871622
## Eye.Brown 0.114864865 0.01182432 0.20101351 0.04391892
## Eye.Green 0.008445946 0.02702703 0.04898649 0.02364865
## Eye.Hazel 0.025337838 0.01689189 0.09121622 0.02364865
```

Sum over columns, sum over rows and sum over all elements.

```r
colSums(HEC)
```

```
## Hair.Black Hair.Blond Hair.Brown   Hair.Red
##  0.1824324  0.2145270  0.4831081  0.1199324
```

```r
rowSums(HEC)
```

```
##  Eye.Blue Eye.Brown Eye.Green Eye.Hazel
## 0.3631757 0.3716216 0.1081081 0.1570946
```

```r
sum(HEC)
```

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

$p(Blue\ Eyes \cap Blond\ Hair)$

```r
HEC["Eye.Blue", "Hair.Blond"]
```

```
## [1] 0.1587838
```

$p(Brown\ Hair)$

```r
sum(HEC["Eye.Brown",])
```

```
## [1] 0.3716216
```

$p(Red\ Hair|Brown\ Eyes)$

```r
HEC["Eye.Brown","Hair.Red"]/sum(HEC["Eye.Brown",])
```

```
## [1] 0.1181818
```

$p((Red\ Hair \cup Blond\ Hair) \cap (Brown\ Eyes \cup Blue\ Eyes))$

```r
sum(HEC[c("Eye.Brown", "Eye.Blue"),c("Hair.Red", "Hair.Blond")])
```

```
## [1] 0.2432432
```

$p((Red\ Hair \cup Blond\ Hair) \cup (Brown\ Eyes \cup Blue\ Eyes))$

```r
sum(HEC[c("Eye.Brown", "Eye.Blue"),]) + sum(HEC[,c("Hair.Red", "Hair.Blond")]) - sum(HEC[c("Eye.Brown",
```

```
## [1] 0.8260135
```

If the attributes are independent the following relation should hold: $p(Blue\ Eyes \cap Blond\ Hair) = p(Blue\ Eyes)p(Blond\ Hair)$

```r
HEC["Eye.Blue", "Hair.Blond"] == sum(HEC["Eye.Blue",]) * sum(HEC[,"Hair.Blond"])
```

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

Proof by contradiction.