Assignment 2

Fredrik Nyström

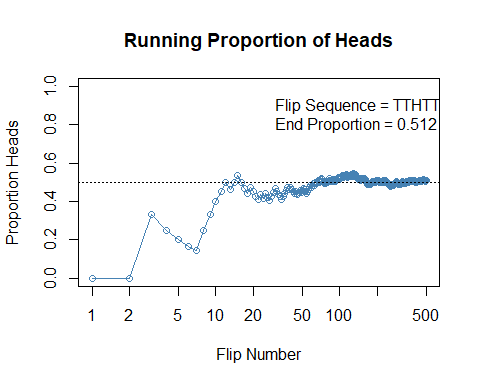
2019-09-11

## Task A

### 1. Discrete random variable

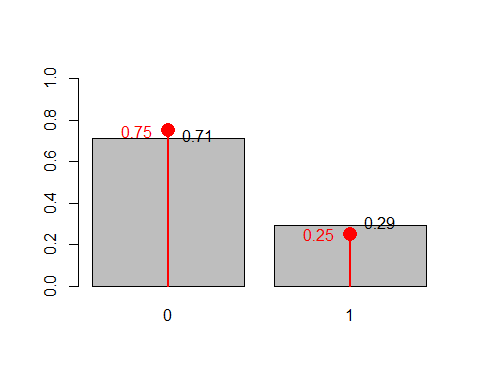
Recreate the 4.1 figure from the book simulating coin tosses.

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 . Sample 100 biased coin tosses and plot a histogram with the true PMF overlaid.

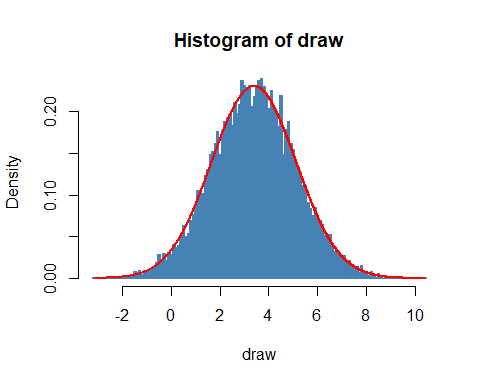
biased\_coin\_toss <- function(n) {  
 bias <- function(theta) { if (theta < 0.25) 1 else 0 }  
   
 sapply(runif(n), bias)  
}  
  
bias\_toss\_data <- biased\_coin\_toss(n = 100)  
#Calculate frequency table   
bias\_toss\_data <- table(bias\_toss\_data)/length(bias\_toss\_data)  
bp <- barplot(bias\_toss\_data, 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 . Plot them as a normalized frequency histogram and overlay the normal PDF.

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

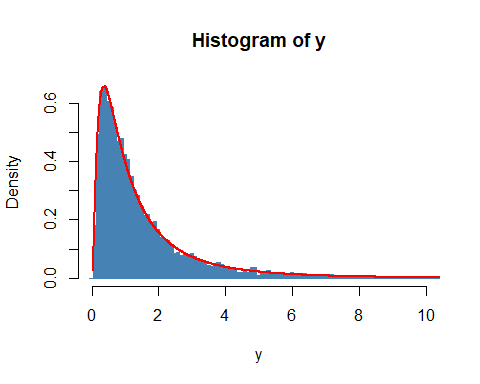
## [1] 3.4

vx

## [1] 3

Plot the histogram and PDF of where .

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 , 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 <- HEC %>% pivot\_wider(names\_from = Hair, values\_from = Count, names\_prefix = "Hair.") %>%  
 column\_to\_rownames(var = "Eye")   
rownames(HEC) <- paste0("Eye.", rownames(HEC))  
  
HEC <- HEC/sum(HEC)

Verify the correctnes.

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.

colSums(HEC)

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

rowSums(HEC)

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

sum(HEC)

## [1] 1

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

## [1] 0.1587838

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

## [1] 0.3716216

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

## [1] 0.1181818

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

## [1] 0.2432432

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

## [1] 0.8260135

If the attributes are independent the following relation should hold:

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

## [1] FALSE

Proof by contradiction.