Assignment 5 solutions

EMATM0061: Statistical Computing and Empirical Methods, TB1, 2024

Dr. Rihuan Ke

Introduction

Load packages

Some of the questions in this assignment require the following packages.

1. The tidyverse package:

```
library(tidyverse)
```

2. The Stat2Data package and then the dataset Hawks:

```
library(Stat2Data)
data("Hawks")
```

1. Exploratory data analysis

This section covers some of the concepts from Lecture 7 on Exploratory Data Analysis.

We will use the Hawks dataset that you have loaded.

```
head(Hawks)
```

```
##   Month Day Year CaptureTime ReleaseTime  BandNumber Species Age Sex
Wing
## 1     9  19 1992       13:30               877-76317      RT   I
385
## 2     9  22 1992       10:30               877-76318      RT   I
376
## 3     9  23 1992       12:45               877-76319      RT   I
381
## 4     9  23 1992       10:50               745-49508      CH   I   F
265
## 5     9  27 1992       11:15              1253-98801      SS   I   F
205
## 6     9  28 1992       11:25             1207-55910       RT   I
412
##   Weight Culmen Hallux Tail StandardTail Tarsus WingPitFat KeelFat
Crop
## 1    920   25.7   30.1  219           NA     NA         NA      NA
NA
## 2    930     NA     NA  221           NA     NA         NA      NA
```

```
NA
## 3    990   26.7   31.3  235           NA      NA          NA       NA
NA
## 4    470   18.7   23.5  220           NA      NA          NA       NA
NA
## 5    170   12.5   14.3  157           NA      NA          NA       NA
NA
## 6   1090   28.5   32.2  230           NA      NA          NA       NA
NA
```

1.1 Location estimators

(Q1) Let's start by computing some location estimators for Hawks' "Tail".

First, create a vector called "HawksTail", the elements of which are from the "Tail" column of Hawks data frame.

Answer:

```
HawksTail = Hawks$Tail
head(HawksTail)

## [1] 219 221 235 220 157 230
```

Second, use the "mean" and "median" functions to compute the sample mean and sample median from the vector "HawksTail". (note that inputs of the "mean" function are vectors. Type ?mean for further details).

Answer:

```
print(mean(HawksTail))

## [1] 198.8315

print(median(HawksTail))

## [1] 214
```

1.2 Combining location estimators with the summarise function

(Q1) Use a combination of the "summarise()", "mean()" and "median()" to compute the sample mean, sample median and trimmed sample mean (with $q = 0.5$) of the Hawk's wing length and Hawk's weight (i.e., the "Wing" and "Weight" columns). You may need to remove the NA values. What can you say by comparing the results of the median and the trimmed mean that you obtain?

Answer:

```
summarise(Hawks, Wing_mean=mean(Wing, na.rm=TRUE),
Wing_t_mean=mean(Wing, trim=0.5, na.rm=TRUE),
          Wing_med=median(Wing, na.rm=TRUE), Weight_mean=mean(Weight,
na.rm=TRUE),
```

```
          Weight_t_mean=mean(Weight, trim=0.5, na.rm=TRUE),
Weight_med=median(Weight, na.rm=TRUE))

##   Wing_mean Wing_t_mean Wing_med Weight_mean Weight_t_mean
Weight_med
## 1  315.6375         370      370    772.0802           970
970
```

(Q2) Combine them with the "group_by()" function to obtain a breakdown by species.

Answer:

```
group_by(Hawks, Species) %>%
  summarise(Wing_mean=mean(Wing, na.rm=TRUE), Wing_t_mean=mean(Wing,
trim=0.5, na.rm=TRUE),
          Wing_med=median(Wing, na.rm=TRUE), Weight_mean=mean(Weight,
na.rm=TRUE),
          Weight_t_mean=mean(Weight, trim=0.5, na.rm=TRUE),
Weight_med=median(Weight, na.rm=TRUE))

## # A tibble: 3 × 7
##   Species Wing_mean Wing_t_mean Wing_med Weight_mean Weight_t_mean
Weight_med
##   <fct>       <dbl>       <dbl>    <dbl>       <dbl>         <dbl>
<dbl>
## 1 CH           244.         240      240        420.          378.
378.
## 2 RT           383.         384      384       1094.         1070
1070
## 3 SS           185.         191      191        148.          155
155
```

1.3 Location and dispersion estimators under linear transformations

(Q1) Suppose that a variable of interest X has values $X_1, \cdots, X_n$. Suppose that $X_1, \cdots, X_n$ has a sample mean A. Let $a, b \in \mathbb{R}$ be real numbers and define a new variable $\tilde{X}$ with $\tilde{X}_1, \cdots, \tilde{X}_n$ defined by $\tilde{X}_i = aX_i + b$ for $i = 1,2, \cdots, n$. What is the sample mean of $\tilde{X}_1, \cdots, \tilde{X}_n$ as a function of a, b and A? (please write down your answer as an expression of a, A, and b. You don't need to use R).

Answer $a * A + b$

Now using the vector "HawksTail" that you created in Section 1.1 as data and letting $a = 2$ and $b = 3$, verify your conclusion using R codes: Compute the mean of "HawksTail*a+b" and then compare it with the one obtained from the mean of "HawksTail" and your conclusion.

Answer

```
mean(HawksTail)*2+3
```

```
## [1] 400.663
```

```
mean(HawksTail*2+3)
```

```
## [1] 400.663
```

(Q2) Suppose further that $X_1, \cdots, X_n$ has sample variance p and standard deviation q. What is the sample variance of $\tilde{X}_1, \cdots, \tilde{X}_n$? What is the sample standard deviation of $\tilde{X}_1, \cdots, \tilde{X}_n$? (Please write down your results.)

Now using the vector "HawksTail" that you created in Section 1.1 as data and letting $a = 2$ and $b = 3$, verify your result using R codes again.

Answer: Variance: $a * a * p$ and standard deviation $a * q$.

Answer

```
var(HawksTail)*4
```

```
## [1] 5424.147
```

```
var(HawksTail*2+3)
```

```
## [1] 5424.147
```

Answer

```
sd(HawksTail)*2
```

```
## [1] 73.64881
```

```
sd(HawksTail*2+3)
```

```
## [1] 73.64881
```

1.4 Robustness of location estimators

In this exercise we shall investigate the robustness of several location estimators: The sample mean, sample median and trimmed mean.

We begin by extracting a vector called "hal" consisting of the talon lengths of all the hawks with any missing values removed.

```
hal<-Hawks$Hallux # Extract the vector of hallux lengths
hal<-hal[!is.na(hal)] # Remove any nans
```

To investigate the effect of outliers on estimates of location we generate a new vector called ""corrupted_hall"" with 10 outliers each of value 100 created as follows:

```
outlier_val<-100
num_outliers<-10
corrupted_hal<-c(hal,rep(outlier_val,times=num_outliers))
```

We can then compute the mean of the original sample and the corrupted sample as follows.

```
mean(hal)

## [1] 26.41086

mean(corrupted_hal)

## [1] 27.21776
```

Now let's investigate what happens as the number of outliers changes from 0 to 1000. The code below generates a vector called ""means_vect"" which gives the sample means of corrupted samples with different numbers of outliers. More precisely, "means_vect" is a vector of length 1001 with the i-th entry equal to the mean of a sample with $i-1$ outliers.

```
num_outliers_vect <- seq(0,1000)
means_vect <- c()
for(num_outliers in num_outliers_vect){
  corrupted_hal <- c(hal,rep(outlier_val,times=num_outliers))
  means_vect <- c(means_vect, mean(corrupted_hal))
}
```

(Q1) *Sample median*:

Copy and modify the above code to create an additional vector called ""medians_vect"" of length 1001 with the i-th entry equal to the median of a sample ""corrupted_hal"" with $i-1$ outliers.

Answer:

```
num_outliers_vect <- seq(0,1000)
medians_vect <- c()
for(num_outliers in num_outliers_vect){
  corrupted_hal <- c(hal,rep(outlier_val,times=num_outliers))
  medians_vect <- c(medians_vect, median(corrupted_hal))
}
```

(Q2) *Sample trimmed mean*:

Amend the code further to add an additional vector called "t_means_vect" of length 1001 with the i-th entry equal to the trimmed mean of a sample with $i-1$ outliers, where the trimmed mean has a trim fraction $q = 0.1$.

Answer:

```
num_outliers_vect <- seq(0,1000)
t_means_vect <- c()
for(num_outliers in num_outliers_vect){
  corrupted_hal <- c(hal,rep(outlier_val,times=num_outliers))
```

```
  t_means_vect <- c(t_means_vect, mean(corrupted_hal, trim=0.1))
}
```

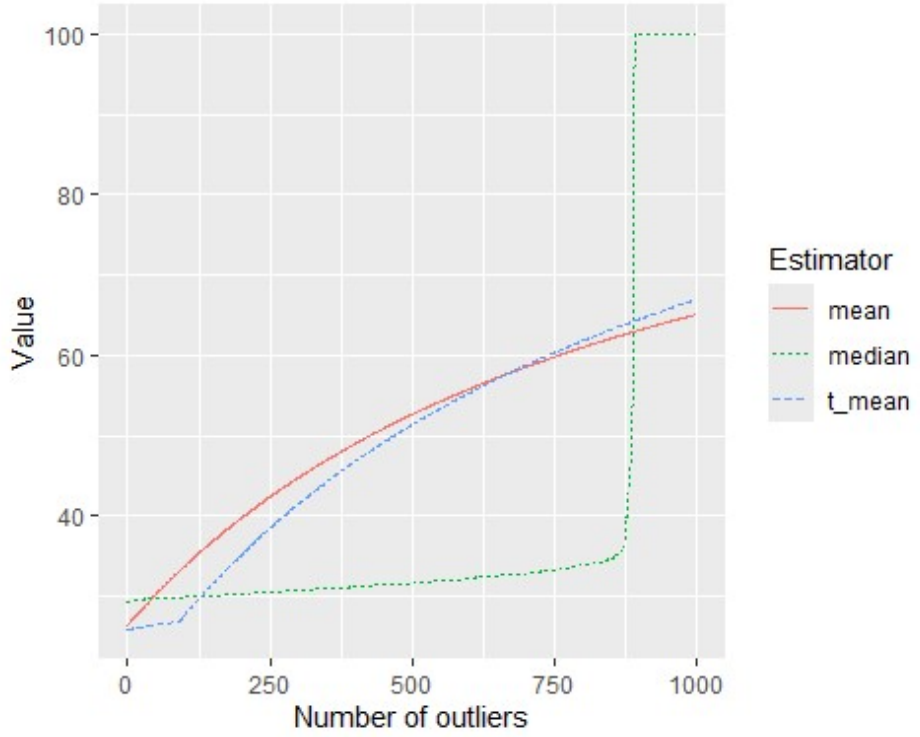
(Q3) *Visualisation*

Now you should have the vectors ""num_outliers_vect"","means_vect"","medians_vect"" and ""t_means_vect"". Combine these vectors into a data frame with the following code.

```
df_means_medians <- data.frame(num_outliers=num_outliers_vect,
mean=means_vect,
                               t_mean=t_means_vect,
median=medians_vect)
```

Now use the code below to reshape and plot the data. Recall that the function "pivot_longer()" below is used to reshape the data. Your result should look like:

```
df_means_medians %>%
  pivot_longer(!num_outliers, names_to = "Estimator", values_to =
"Value") %>%
  ggplot(aes(x=num_outliers,color=Estimator,
linetype=Estimator,y=Value)) +
  geom_line()+xlab("Number of outliers")
```



Which quantity is the most robust when the number of outliers is small? (Note that, in this experiment, the term outliers simply means the artificial data used to corrupt the vector. It is not related to the outliers computed in the next question).

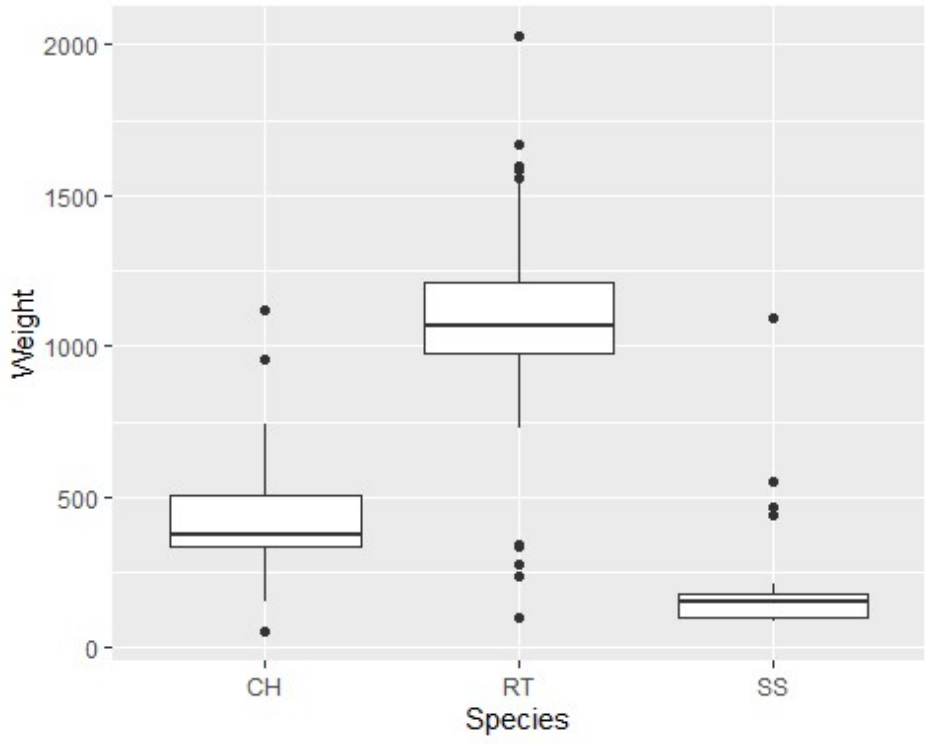
Answer: When the number of outliers is small, the sample median is the most robust one.

1.5 Box plots and outliers

(Q1) Use the functions "ggplot()" and "geom_boxplot()" to create a box plot which summarises the distribution of hawk weights broken down by species. Your plot should look as follows:

Answer

```
ggplot(Hawks, aes(x=Species, y=Weight)) + geom_boxplot()
```



Note the outliers are displayed as individual dots.

(Q2) *quantile and boxplots*

Compute the 0.25-quantile, 0.5-quantile, 0.75-quantile of the "Weight" grouped by Species. Your results should look like this **Answer**

```
group_by(Hawks, Species) %>%
  summarise(quantile025=quantile(Weight, probs=0.25, na.rm=TRUE),
            quantile050=quantile(Weight, probs=0.5, na.rm=TRUE),
            quantile075=quantile(Weight, probs=0.75, na.rm=TRUE))

## # A tibble: 3 × 4
##   Species quantile025 quantile050 quantile075
##   <fct>         <dbl>       <dbl>       <dbl>
```

```
## 1 CH              335        378.        505
## 2 RT              980       1070        1210
## 3 SS              100        155         178.
```

Now compare these values with the boxplot above. Can you explain which parts of the boxplot these numbers correspond to?

Answer: The $0.25-quantile$ and $0.75-quantile$ correspond to the bottom and the top of each box in the plot, respectively. The $0.5-quantile$ corresponds to the bar at the middle of the box.

(Q3) *Outliers*

Suppose we have a sample $X_1, \cdots, X_n$. Let "q25" denote the 0.25-quantile of the sample and let "q75" denote the 0.75-quantile of the sample. We can then define the interquartile range, denoted IQR by IQR := q75-q25. In the context of boxplots, an outlier X_i is any numerical value such that the following holds if either of the following holds:

$$X_i < \text{q25} - 1.5 \times \text{IQR}, \quad \text{or}$$
$$X_i > \text{q75} + 1.5 \times \text{IQR}.$$

Create a function called ""num_outliers"" which computes the number of outliers within a sample (with missing values excluded). The function should take a vector as input as output a number.

Answer

```
num_outliers <- function(x){
  q25 <- quantile(x, 0.25, na.rm=TRUE)
  q75 <- quantile(x, 0.75, na.rm=TRUE)
  iq_range <- q75 - q25
  num <- sum( (x>q75+1.5*iq_range)|(x<q25-1.5*iq_range), na.rm=TRUE )
  return (num)
}
```

Test your ""num_outliers"" function using the code below:

```
num_outliers( c(0, 40,60,185))
```

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

(Q4) *Outliers by group*

Now combine your function "num_outliers()" with the functions "group_by()" and "summarise()" to compute the number of outliers for the three samples of hawk weights broken down by species. Your result should look as follows:

Answer

```
group_by(Hawks, Species) %>%
  summarise(num_outliers_weight = num_outliers(Weight))
```

```
## # A tibble: 3 × 2
##   Species num_outliers_weight
##   <fct>                 <int>
## 1 CH                        3
## 2 RT                       13
## 3 SS                        4
```

You may want to go back to the above box plot to check the number of dots displayed for each group.

1.6 Covariance and correlation under linear transformations

(Q1) Compute the covariance and correlation between the "Weight" and "Wing" of the "Hawks" data. You can use the cov and cor functions.

Answer:

```
cov(Hawks$Weight, Hawks$Wing, use='complete.obs')
```

```
## [1] 41174.39
```

```
cor(Hawks$Weight, Hawks$Wing, use='complete.obs')
```

```
## [1] 0.9348575
```

(Q2) Suppose that we have a pair of variables: X with values $X_1, \cdots, X_n$ and Y with values $Y_1, \cdots, Y_n$. Suppose that $X_1, \cdots, X_n$ and $Y_1, \cdots, Y_n$ have the sample covariance S and correlation R. Let $a, b \in \mathbb{R}$ be real numbers and define a new variable $\tilde{X}$ with $\tilde{X}_1, \cdots, \tilde{X}_n$ defined by $\tilde{X}_i = aX_i + b$ for $i = 1,2, \cdots, n$. In addition, let $c, d \in \mathbb{R}$ be real numbers and define a new variable $\tilde{Y}$ with $\tilde{Y}_1, \cdots, \tilde{Y}_n$ defined by $\tilde{Y}_i = cY_i + d$.

What is the covariance between $\tilde{X}_1, \cdots, \tilde{X}_n$ and $\tilde{Y}$ with $\tilde{Y}_1, \cdots, \tilde{Y}_n$ (as a function of S, a, b, c, d)? Assuming that $a \neq 0$ and $c \neq 0$, what is the correlation between $\tilde{X}_1, \cdots, \tilde{X}_n$ and $\tilde{Y}$ with $\tilde{Y}_1, \cdots, \tilde{Y}_n$? Please write down the mathematical expressions.

Answer

covariance: $a \cdot b \cdot S$

correlation:

$$\text{sign}(a \cdots b) \cdot R$$

Let $a = 2.4, b = 7.1, c = -1, d = 3$, and let X be the hawk's weight and Y be the hawk's Wing. Verify your conclusion with R codes in a similar way to Section 1.3 (Q1).

Answer

```
cov(Hawks$Weight, Hawks$Wing, use='complete.obs')*2.4*(-1) -
  cov(Hawks$Weight*2.4+7.1, Hawks$Wing*(-1)+3, use='complete.obs')
```

```
## [1] 0
```

Answer

```
cor(Hawks$Weight, Hawks$Wing, use='complete.obs')*sign(2.4*(-1)) -
  cor(Hawks$Weight*2.4+7.1, Hawks$Wing*(-1)+3, use='complete.obs')
```

```
## [1] 1.110223e-16
```

2. Random variables and discrete random variables

This section covers some of the concepts from Lectures 12 and 13.

Random variables and discrete random variables Suppose we have a probability space $(\Omega, \mathcal{E}, \mathbb{P})$. A random variable is a mapping $X: \Omega \to \mathbb{R}$, such that

$$\text{for every } a, b \in \mathbb{R}, \{\omega \in \Omega: X(\omega) \in [a, b]\} \text{ is an event in } \mathcal{E}$$

A discrete random variable is a random variable $X: \Omega \to \mathbb{R}$ whose distribution is supported on a discrete (and hence finite or countably infinite) set $A \subseteq \mathbb{R}$

Expectation

The expectation $\mathbb{E}(X)$ of the random variable X is defined by $\mathbb{E}(X) := \sum_{x \in \mathbb{R}} x \cdot p_X(x)$.

Linearity of Expectation: Given random variables $X_1, X_2, \cdots X_n$ and numbers $\alpha_1, \alpha_2, \cdots, \alpha_n$, we have

$$\mathbb{E}\left(\sum_{i=1}^{n} \alpha_i X_i\right) = \sum_{i=1}^{K} \alpha_i \, \mathbb{E}(X_i)$$

Equivalent condition for independent random variables

Let $X_1, \cdots, X_k: \Omega \to \mathbb{R}$ be a sequence of random variables. Then $X_1, \cdots, X_k$ are independent if and only if the following relationship holds for every sequence of well-behaved function $f_1, f_2, \cdots, f_k$,

$$\mathbb{E}\big(f_1(X_1) \cdots f_k(X_k)\big) = \mathbb{E}\big(f_1(X_1)\big) \cdots \mathbb{E}\big(f_k(X_k)\big).$$

2.1 Expectation and variance

(Q1) Suppose that we have random variables X and Y. Recall that the covariance between X and Y is defined by

$$\text{Cov}(X, Y) := \mathbb{E}[(X - \bar{X}) \cdot (Y - \bar{Y})]$$

where $\bar{X}$ and $\bar{Y}$ are the expectations of X and Y, respectively.

Now, suppose X and Y are ***independent***. Apply the linearity of expectation and the equivalent condition for independent random variables described above, to prove that $\mathrm{Cov}(X, Y) = 0$. To apply the above condition for independent random variables, you may choose proper functions $f_1, f_2, \cdots$ that you need.

Answer

Firstly, we can write the covariance into the following form

$$\begin{aligned}\mathrm{Cov}(X, Y) &:= \mathbb{E}[(X - \bar{X}) \cdot (Y - \bar{Y})] \\ &= \mathbb{E}[XY] - \mathbb{E}[X]\mathbb{E}[Y].\end{aligned}$$

Secondly, since X and Y are independent, letting $f_1(x) := x$, $f_2(x) := x$, we have

$$\mathbb{E}[XY] = \mathbb{E}[f_1(X)f_2(Y)] = \mathbb{E}[f_1(X)] \cdot \mathbb{E}[f_2(Y)] = \mathbb{E}[X]\mathbb{E}[Y].$$

Therefore,

$$\mathrm{Cov}(X, Y) = \mathbb{E}[XY] - \mathbb{E}[X]\mathbb{E}[Y] = 0.$$

2.2 Distributions

Suppose that $\alpha, \beta \in [0,1]$ with $\alpha + \beta \leq 1$ and let X be a discrete random variable with distribution supported on $\{0,3,10\}$. Suppose that $\mathbb{P}(X = 3) = \alpha$ and $\mathbb{P}(X = 10) = \beta$ and $\mathbb{P}(X \notin \{0,3,10\}) = 0$.

Given the random variable X, answer the following questions (Q1), (Q2), …, Q(4).

(Q1) Expectation and variance of a discrete random variable

1. What is the probability mass function p_X for X?
2. What is the expectation of X?
3. What is the variance of X?
4. What is the standard deviation of X?

Answer

The probability mass function is

$$p_X(x) = \begin{cases} 1 - \alpha - \beta & \text{if } x = 0, \\ \alpha & \text{if } x = 3, \\ \beta & \text{if } x = 10, \\ 0 & \text{otherwise .} \end{cases}$$

The expectation is

$$\mathbb{E}(X) = 3 \times \alpha + 10 \times \beta = 3\alpha + 10\beta$$

The variance is

$$\begin{aligned}\mathrm{Var}(X) = \mathbb{E}(X^2) - \mathbb{E}(X)^2 &= (3^2 \times \alpha + 10^2 \times \beta) - (3\alpha + 10\beta)^2 \\ &= 9\alpha + 100\beta - 9\alpha^2 - 100\beta^2 - 60\alpha\beta\end{aligned}$$

The standard deviation is

$$\sigma(X) = \sqrt{\mathrm{Var}(X)} = \sqrt{9\alpha + 100\beta - 9\alpha^2 - 100\beta^2 - 60\alpha\beta}.$$

(Q2) Distribution and distribution function.

Recall that the distribution of a random variable maps a subset S of $\mathbb{R}$ to a real number.

1. Write down the distribution P_X of X. You can use the indicator function $\mathbf{1}_S$ to "indicate" if a number is in the set S. (To give an example, the distribution of a Bernoulli random variable can be written as $P(S) = (1-q)\mathbf{1}_S(0) + q\mathbf{1}_S(1)$.)
2. Write down the distribution function F_X of X

Answer

The distribution is given by

$$P_X(S) = (1 - \alpha - \beta)\mathbf{1}_S(0) + \alpha\mathbf{1}_S(3) + \beta\mathbf{1}_S(10).$$

The distribution function is given by

$$F_X(x) = \begin{cases} 0, & \text{if } x < 0 \\ 1 - \alpha - \beta, & \text{if } 0 \le x < 3 \\ 1 - \beta, & \text{if } 3 \le x < 10 \\ 1, & \text{if } 10 \le x \end{cases}$$

(Q3) Variance and Covariance.

Define a new random variable $Y = X_1 + X_2 + \cdots + X_n$ where $X_1, X_2, \cdots, X_n$ are independent random variables, each of which has the same distribution as the random variable X.

Derive the variance of Y as a mathematical expression of α, β and n. You can use the conclusion from Question 2.1 (Q1).

Answer

Since $X_1, \cdots, X_n$ are independent, $\mathrm{Cov}(X_i, X_j) = 0$ for any $i \neq j$.

$$\mathrm{Var}\left(\sum_{i=1}^{n} X_i\right) = \sum_{i=1}^{n} \mathrm{Var}\,(X_i) + 2 \sum_{1 \le i < j \le n} \mathrm{Cov}\,(X_i, X_j) = \sum_{i=1}^{n} \mathrm{Var}\,(X_i)$$
$$= n \cdot (9\alpha + 100\beta - 9\alpha^2 - 100\beta^2 - 60\alpha\beta).$$

(Q4) Carry out an experiment with **"R" programming** to explore the distribution of the sum of n independent random variables when n is large.

Let Y be defined above, and additionally, let $\alpha = 0.2$ and $\beta = 0.3$. Recall that $X_1, X_2, \cdots, X_n$ are discrete random variables. Explain why the random variable Y is discrete.

Now let's explore the probability mass function of Y, and see its behaviour when n is large.

(Step 1).

First, write a function called "Gen_X_numbers" to generate a sample for $X_1, X_2, \cdots, X_n$. The function "Gen_X_numbers" should take a number "n" as input and return a vector containing n random numbers generated from the distribution of X.

To generate a single random number of X, first generate a random number from the uniform distribution of [0,1]. If the generated number is in [0,0.5), assign 0 to the value of X. If the generated number is in [0,5,0.7), assign 3 to the value of X. If the generated number is in [0,7,1], assign 10 to the value of X.

To generate random numbers from the uniform distribution of [0,1], you could use the "runif" function (type ?runif to see the information about runif).

Answers

```
Gen_X_numbers <- function(n){
  Uniform <- runif(n)
  X = 0*(Uniform<0.5) + 3 * ( (Uniform>=0.5)*(Uniform<0.7) ) + 10 *
(Uniform>0.7)
  return (X)
}

set.seed(1002)
```

The result may look like this:

```
Gen_X_numbers(4)

## [1]  0 10  3  0

# the results you get might be different because of randomness
```

(step 2)

Write a function called "Gen_Y_samples" to generate a sample of Y. The function "Gen_Y_samples" should take m and n as input and return a data frame that has a column called Y and m rows. The Y column contain sample of Y of size m (i.e., m random numbers for the random variable $Y := X_1 + X_2 + \cdots + X_n$). You may want to use the "Gen_X_numbers" function you defined and the "map_dbl" function for iterations.

Answers

```
Gen_Y_samples <- function(m,n){
  Y_sample <- data.frame(index=seq(m)) %>%
    mutate(Y=map_dbl(index, ~ sum(Gen_X_numbers(n)) ))
  return (Y_sample)
}
```

The result may look like this:

```
Gen_Y_samples(5, 2)
```

```
##   index  Y
## 1     1 10
## 2     2 13
## 3     3 10
## 4     4 13
## 5     5  3
```

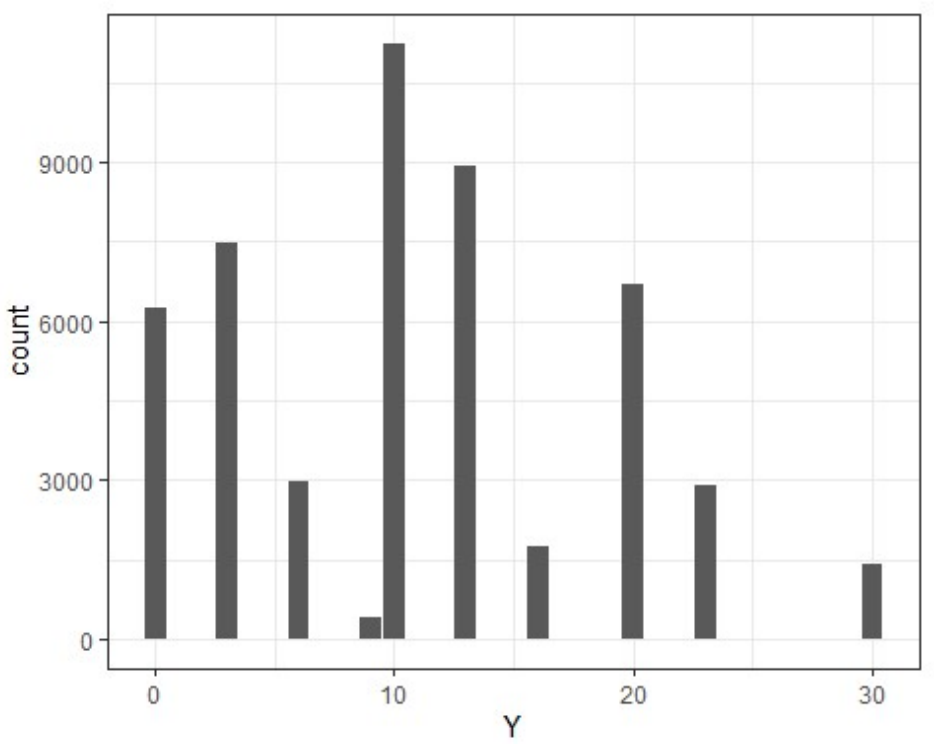
```
# the results you get might be different because of randomness
```

(Step 3)

Now let $n = 3$ and therefore $Y := X_1 + X_2 + X_3$. Generate $m := 50000$ samples of Y. Use the "ggplot" and "geom_bar()" function to create a bar plot for the samples of Y.

Answer

```
samples_Y <- Gen_Y_samples(50000, 3)
ggplot(samples_Y, aes(Y)) + geom_bar() + theme_bw()
```



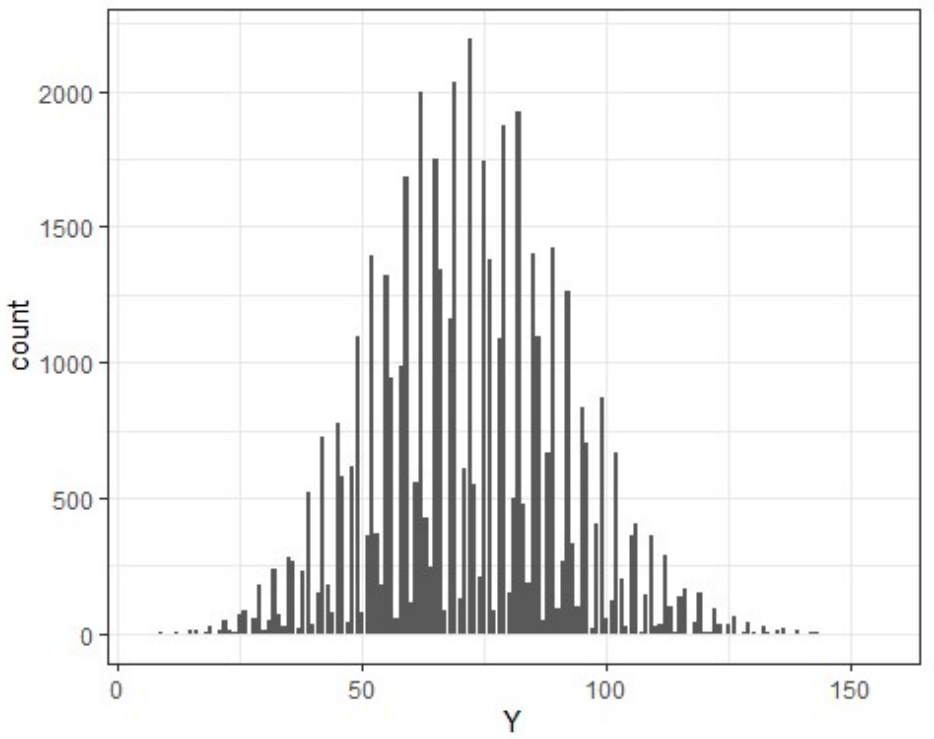
Of course your results might look different because of the randomness when using the "runif()" function. Notice that Y takes values from a subset of [0,30].

(Step 4)

Now, increase the values of n to 20. Generate a new sample for Y and create a new bar plot for the sample of Y. What are the maximum and minimum values of the samples? What is the sample range?

Answer

```
samples_Y <- Gen_Y_samples(50000, 20)
ggplot(samples_Y, aes(Y)) + geom_bar() + theme_bw()
```



The minimum value, maximum value, and range of the sample is

```
print(range(samples_Y))
```

```
## [1]     1 50000
```

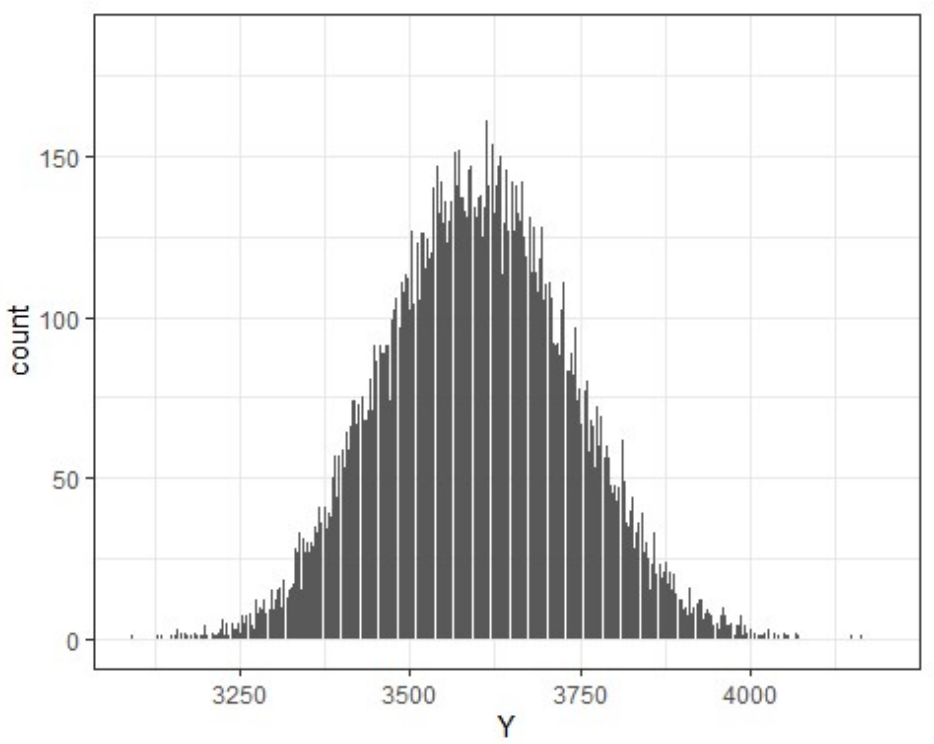
```
print(diff(range(samples_Y)))
```

```
## [1] 49999
```

(Step 5) Next, increase n to 1000 and do a similar plot for the sample of Y.

Answer

```
samples_Y <- Gen_Y_samples(50000, 1000)
ggplot(samples_Y, aes(Y)) + geom_bar() + theme_bw()
```



The bar plots contain information about the distribution of $Y := X_1 + X_2 + \cdots + X_n$ for different n.

Notice that as we increase n, the distribution of Y tends to follow a bell-like shape. While Y is a discrete random variable (no matter how large n is), the distribution looks closer to the distribution of a continuous random variable, which is known as a Gaussian random variable. We will explore this behaviour in our lecture 14

3. Continuous random variables and limit laws

This section is mainly based on Lecture 14. We will explore several continuous random variables and the limiting behaviors of sequences of i.i.d. random variables.

3.1 Simulating data with the uniform distribution

Suppose that $\alpha, \beta \in [0,1]$ with $\alpha + \beta \leq 1$ and let X be a discrete random variable with distribution supported on $\{0,3,10\}$. Suppose that $\mathbb{P}(X = 3) = \alpha$ and $\mathbb{P}(X = 10) = \beta$, $\mathbb{P}(X = 0) = 1 - \alpha - \beta$, and $\mathbb{P}(X \notin \{0,3,10\}) = 0$.

We will use the uniform distribution to simulate data for the discrete random variable X. A standard uniformly distributed random variable U is a continuous random variable with probability density function

$$p_U(x) = \begin{cases} 1 \text{ if } x \in [0,1], \\ 0 \text{ otherwise.} \end{cases}$$

(Q1). Show that for any pair of numbers $a, b \in \mathbb{R}$ with $0 \leq a \leq b \leq 1$, we have $\mathbb{P}(U \in [a,b]) = b - a$.

Answers

We have

$$\mathbb{P}(U \in [a,b]) = \int_a^b p_U(x)dx = \int_a^b 1\, dx = b - a.$$

as required.

(Q2).

Now, let's consider the discrete random variable X (as defined above) and the case with $\alpha = \beta = 0.25$. You can generate a sequence of i.i.d. copies $X_1, X_2, \cdots, X_n$ of X as follows:

```
set.seed(0)
n <- 1000
sample_X <- data.frame(U=runif(n)) %>%
  mutate(X=case_when(
    (0<=U)&(U<0.25)~3,
    (0.25<=U)&(U<0.5)~10,
    (0.5<=U)&(U<=1)~0)) %>%
  pull(X)
```

Why does this "sample_X" correspond to a sequence of i.i.d. copies $X_1, X_2, \cdots, X_n$ of X where $\mathbb{P}(X = 3) = \alpha$ and $\mathbb{P}(X = 10) = \beta$, and $\mathbb{P}(X = 0) = 1 - \alpha - \beta$ with $\alpha = \beta = 0.25$?

Hint: You can answer this question by finding the values of $\mathbb{P}(X_i = 3)$, $\mathbb{P}(X_i = 10)$, $\cdots$ for the generated X_i. Your answer should make use of the distribution of U. Confirm that the distributions of X_i and X are the same. Then $X_1, X_2, \cdots, X_n$ are i.i.d. copies of X because they are independent.

Answer

The numbers in "sample_X" are generated from the distribution of X, with $\mathbb{P}(X = 3) = \mathbb{P}(U \in [0,0.25)) = 1/4$, $\mathbb{P}(X = 10) = \mathbb{P}(U \in [0.25,0.5)) = 1/4$, and $\mathbb{P}(X = 0) = \mathbb{P}(U \in [0.5,1]) = 1/2$. So the sample "sample_X" can be viewed as a sequence of i.i.d. copies $X_1, X_2, \cdots, X_n$.

(Q3)

Now create a function called "sample_X_0310()" which takes as inputs "alpha", "beta" and "n" and outputs a sample $X_1, X_2, \cdots, X_n$ of independent copies of X where $\mathbb{P}(X = 3) = \alpha$ and $\mathbb{P}(X = 10) = \beta$, and $\mathbb{P}(X = 0) = 1 - \alpha - \beta$ with $\alpha = \beta = 0.25$.

Answer

```
sample_X_0310 <- function(alpha, beta, n){
  sample_X <- data.frame(U=runif(n)) %>%
  mutate(X=case_when(
    (0<=U)&(U<alpha)~3,
    (alpha<=U)&(U<alpha+beta)~10,
    (alpha+beta<=U)&(U<=1)~0)) %>%
  pull(X)
  return (sample_X)
}
```

(Q4)

Next take $\alpha = 1/2$ and $\beta = 1/10$, and use your function "sample_X_0310()" to create a sample of size $n = 10000$ of the form $X_1, X_2, \cdots, X_n$ consisting of independent copies of X. What is the sample average of $X_1, X_2, \cdots, X_n$? How does this compare with the theoretical value of $\mathbb{E}(X)$ (recall that We have worked on the expression for this expectation in assignment 5 Section 2.2)? Then use your understanding of the law of large numbers to explain this behavior.

Answers

```
n <- 10000
alpha <- 1/2
beta <- 1/10
sample_X <- sample_X_0310(alpha,beta,n)
mean(sample_X)

## [1] 2.5135
```

The expectation is

$$\mathbb{E}(X) = 3 \times \alpha + 10 \times \beta = 3\alpha + 10\beta = 3/2 + 1 = 2.5$$

The sample mean is close to the population mean $\mathbb{E}(X)$, this is a direct consequence of the law of large numbers, which states that the sample mean gets close to the expectation for large samples of independent and identically distributed random variables.

(Q5)

Based on the sample $X_1, X_2, \cdots, X_n$ generated from the last question, compute the sample variance of $X_1, X_2, \cdots, X_n$ and compare it with the population variance $\mathrm{Var}(X)$ (again, the expression of $\mathrm{Var}(X)$ has been derived in Assignment 5).

Answer

```
var(sample_X)

## [1] 8.328651
```

The population variance is

$$\begin{aligned}\mathrm{Var}(X) = \mathbb{E}(X^2) - \mathbb{E}(X)^2 &= (3^2 \times \alpha + 10^2 \times \beta) - (3\alpha + 10\beta)^2 \\ &= 9\alpha + 100\beta - 9\alpha^2 - 100\beta^2 - 60\alpha\beta = 8.25\end{aligned}$$

So the sample variance is close to the population variance.

(Q6)

Now take $n = 100$, $\alpha = 1/10$ and vary β in increments of 0.01 from 0 to $9/10$ (including $9/10$). Create a data frame that has the following four columns. You can use the functions "mutate", "map", and "map_dbl" to help you.

1. The first column is called "beta" (contains different β values ranging from 0 to $9/10$).
2. The second column is called "sample_X" including samples generated with the corresponding β. More specifically, for each value of β (in one of the rows), create a sample of $X_1, X_2, \cdots, X_n$ consisting of independent copies of X, by using your function "sample_X_0310()".
3. The third column is called "samplemean", which contains sample means of the samples in the second column.
4. The last column is called "Expectation", which contains numerical values of the theoretical population mean $\mathbb{E}(X)$ (for the corresponding value of β in the same row).

Answer

```
n = 100
alpha = 1/10
samples <- data.frame(beta = seq(0, 9/10, 0.01)) %>%
  mutate( sample_X =  map(beta, ~sample_X_0310(alpha,.x,n) )) %>%
  mutate( samplemean = map_dbl(sample_X, mean) ) %>%
  mutate( Expectation = 3*alpha + 10*beta)
```

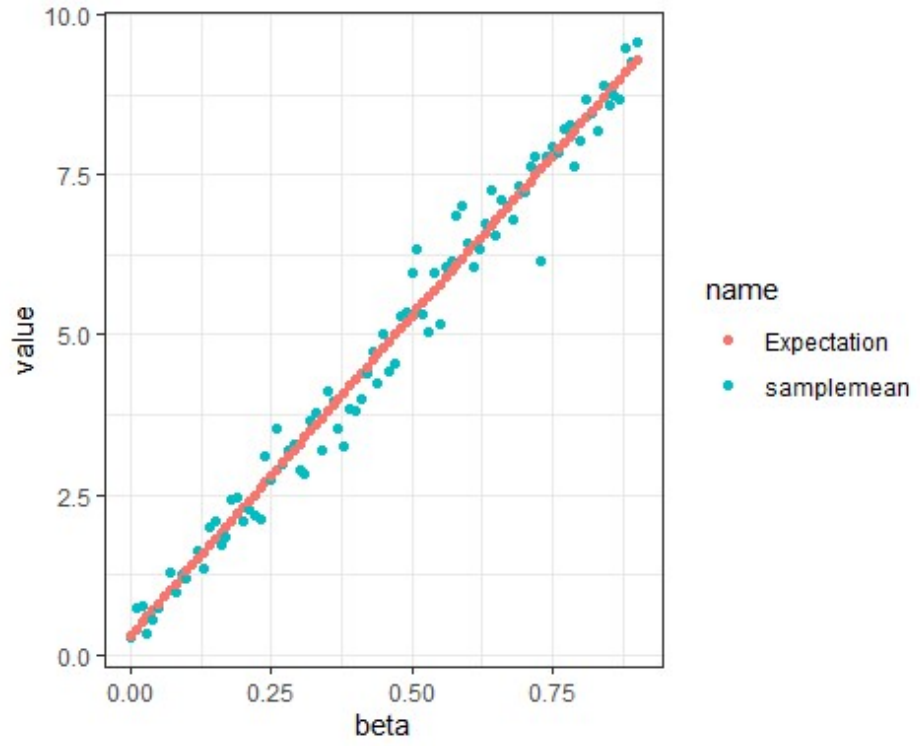
(Q7)

Create a plot of the sample averages and $\mathbb{E}(X)$ as a function of β

Answer

```
df <- samples %>% pivot_longer(cols=c(samplemean, Expectation),
                               names_to = 'name', values_to = 'value')
```

```
ggplot(df, aes(x=beta, y=value, color=name)) +
  geom_point()  + theme_bw()
```



3.2 Exponential distribution

Let $\lambda > 0$ be a positive real number. An exponential random variable X with rate parameter λ is a continuous random variable with density $p_\lambda: \mathbb{R} \to (0, \infty)$ defined by

$$p_\lambda(x) = \begin{cases} 0 \text{ if } x < 0, \\ \lambda e^{-\lambda x} \text{ if } x \geq 0. \end{cases}$$

(Q1)

Prove that p_λ is a well-defined probability density function. Then derive mathematical expressions for the cumulative distribution function F_X and the quantile function for exponential random variables with parameter λ.

Answer

Clearly, $p_\lambda(x) \geq 0$ for $x \in \mathbb{R}$, and

$$\int_{-\infty}^{\infty} p_\lambda(x)dx = \int_0^{\infty} \lambda\, e^{-\lambda x} dx = \left[-e^{-\lambda x}\right]_0^{\infty} = 1.$$

Therefore, $p_\lambda(x)$ is a well-defined probability function.

The cumulative distribution function is

$$F_X(x) = \int_{-\infty}^{x} p_\lambda(t)dt = \left[-e^{-\lambda}\right]_0^x = 1 - e^{-\lambda x}$$

for $x \geq 0$, and $F_X(x) = 0$ for $x < 0$.

The quantile function is $F_X^{-1}(p) = \inf\{x \in \mathbb{R}: F_X(x) \leq p\}$, so

$$F_X^{-1}(p) = \ln(1-p)/(-\lambda)$$

for $0 < p \leq 1$ and $F_X^{-1}(p) = -\infty$ for $p = 0$.

(Q2)

Now implement a function called "my_cdf_exp()". The function "my_cdf_exp()" should take as input two numbers $x \in \mathbb{R}$ and $\lambda > 0$ and output the value of the cumulative distribution function $F_X(x)$ where X is an exponential random variable with rate parameter λ.

Answer

```
my_cdf_exp <- function(x, lambda){
  if (x<0) {return (0)}
  return (1-exp(-lambda*x))
}
```

Check your function my_cdf_exp() gives rise to the following output:

```
lambda <- 1/2
map_dbl(.x=seq(-1,4), .f=~my_cdf_exp(x=.x,lambda=lambda) )

## [1] 0.0000000 0.0000000 0.3934693 0.6321206 0.7768698 0.8646647
```

Then type ?pexp into your R console to learn more about R's inbuilt cumulative distribution function for the exponential distribution. We can now confirm that our "my_cdf_exp" is correct when $\lambda = 1/2$ as follows:

```
test_inputs <- seq(-1,10,0.1)
my_cdf_output <- map_dbl(.x=test_inputs,
.f=~my_cdf_exp(x=.x,lambda=lambda))
inbuilt_cdf_output <-
map_dbl(.x=test_inputs,.f=~pexp(q=.x,rate=lambda))
all.equal(my_cdf_output,inbuilt_cdf_output)

## [1] TRUE
```

(Q3)

Next implement a function called "my_quantile_exp()". The function "my_quantile_exp()" should take as input two arguments $p \in [0,1]$ and $\lambda > 0$ and output the value of the quantile function $F_X^{-1}(p)$ (recall the mathematical expression of the quantile function in (Q1)) where X is an exponential random variable with rate parameter λ.

Once you have implemented your function compare it with R's inbuilt qexp function using the same procedure as we used above (in (Q2)) for the cumulative distribution function for inputs $\lambda = 1/2$ and $p \in \{0.01, 0.02, \cdots, 0.99\}$. Note that you don't need to consider inputs $p \leq 0$ or $p \geq 1$ here.

Answer

```
my_quantile_exp <- function(p, lambda){
  if (p<=0) return (-Inf)
  return (log(1-p)/(-lambda))
}

test_inputs <- seq(0.01, 0.99, 0.01)
my_quantile_output <- map_dbl(.x=test_inputs,
.f=~my_quantile_exp(p=.x,lambda=lambda))
inbuilt_quantile_output <-
map_dbl(.x=test_inputs,.f=~qexp(p=.x,rate=lambda))
all.equal(my_quantile_output,inbuilt_quantile_output)

## [1] TRUE
```

(Q4)

From the probability density function, derive an expression for the population mean and variance of an exponential random variable X with parameter λ. You may want to use integration by parts when computing the integrals.

Answer

The expectation is

$$\begin{aligned}
\mathbb{E}(X) &= \int_{-\infty}^{\infty} x\, p_\lambda(x) dx \\
&= \int_0^{\infty} x \cdot \lambda e^{-\lambda x} dx \\
&= \left[-xe^{-\lambda x}\right]_0^{\infty} + \int_0^{\infty} e^{-\lambda x}\, dx \\
&= 0 + \left[-\frac{1}{\lambda} e^{-\lambda x}\right]_0^{\infty} = \frac{1}{\lambda},
\end{aligned}$$

where in the 3rd equality we have used integration by parts.

To compute the variance, with integration by parts again we have

$$\begin{aligned}\mathbb{E}(X) &= \int_{-\infty}^{\infty} x^2\, p_\lambda(x)dx \\ &= \int_0^{\infty} x^2 \cdot \lambda e^{-\lambda x} dx \\ &= \left[-x^2 e^{-\lambda}\ \right]_0^{\infty} + 2\int_0^{\infty} x\, e^{-\lambda x} dx \\ &= 0 + \frac{2}{\lambda}\int_0^{\infty} \lambda\, x e^{-\lambda x} = \frac{2}{\lambda}\cdot \mathbb{E}(X) = \frac{2}{\lambda^2},\end{aligned}$$

Hence, the variance $\mathrm{Var}(X) = \mathbb{E}(X^2) - \big(\mathbb{E}(x)\big)^2 = \frac{2}{\lambda^2} - \frac{1}{\lambda^2} = \frac{1}{\lambda^2}$.

3.3 The Binomial distribution and the central limit theorem

Two important discrete distributions are the Bernoulli distribution and the Binomial distribution.

We say that a random variable X has Bernoulli distribution with parameter $p \in [0,1]$ if $\mathbb{P}(X = 1) = p$ and $P(X = 0) = 1 - p$. This is often abbreviated as $X \sim \mathcal{B}(p)$.

Given $n \in \{1,2,3,\cdots\}$ and $p \in [0,1]$, we say that a random variable Z has a Binomial distribution with parameters n and p if $Z = X_1 + \cdots + X_n$ for some random variables $\{X_i\}$ where $X_i \sim \mathcal{B}(p)$ and $X_1, \cdots, X_n$ are independent and identically distributed. This is often abbreviated as $Z \sim \mathrm{Binom}(n, p)$.

(Q1)

Give an expression for the expectation and variance of $Z \sim \mathrm{Binom}(n, p)$. You may want to make use of the following two useful facts:

1. Given any sequence of random variables $W_1, \cdots, W_k$ we have $\mathbb{E}\big(\sum_{i=1}^k W_i\big) = \sum_{i=1}^k \mathbb{E}\,(W_i)$.
2. Given **independent** random variables $W_1, \cdots, W_k$ we have $\mathrm{Var}\big(\sum_{i=1}^k W_i\big) = \sum_{i=1}^k \mathrm{Var}\,(W_i)$.

Note that the second fact may not hold if $W_1, \cdots, W_k$ are not independent.

Answer

Note that since $X_i \in \mathcal{B}(p)$, we have $\mathbb{E}(X_i) = p$ and $\mathrm{Var}(X_i) = p(1-p)$. From the fact 1., we have

$$\mathbb{E}(Z) = \mathbb{E}\left(\sum_{i=1}^n X_i\right) = \sum_{i=1}^n \mathbb{E}\,(X_i) = \sum_{i=1}^p p = np.$$

From the fact 2., we have

$$\mathrm{Var}(Z) = \mathrm{Var}\left(\sum_{i=1}^{n} X_i\right) = \sum_{i=1}^{n} \mathrm{Var}\,(X_i) = np(1-p).$$

(Q2)

The function "dbinom()" in R allows us to compute the probability mass function of a Binomial random variable $Z \sim \mathrm{Binom}(n,p)$. For $x \in \{0,1,\cdots,n\}$, the function "dbinom(x,size=n,prob=p)" will return the value of the probability mass function evaluated at x, i.e., it returns $p_z(x) = \mathbb{P}(Z = x)$. You can run ?dbinom in the R console to find out more.

Consider the case where $n = 50$ and $p = 7/10$. Use the "dbinom()" to generate a data-frame called "binom_df" with two columns called "x" and "pmf".

1. The first column contains the numbers $\{0,1,\cdots,50\}$ inclusive. These are different values of x.

2. The second column gives the corresponding value of the probability mass function $p_z(x) = \mathbb{P}(Z = x)$ with $Z \sim \mathrm{Binom}(50,0.7)$. Use the "head()" function to observe the first 3 rows of your data frame.

Answers

```
binom_df <- data.frame(x = seq(0,50)) %>%
  mutate(pmf = map_dbl(x, ~dbinom(.x, size=50, prob=0.7)) )

head(binom_df, 3)
```

```
##   x          pmf
## 1 0 7.178980e-27
## 2 1 8.375477e-25
## 3 2 4.787981e-23
```

(Q3)

The function "dnorm()" in R allows us to compute the probability density function of a Gaussian random variable $W \sim \mathcal{N}(\mu,\sigma^2)$ with expectation μ and variance σ^2. The function "dnorm(x,mean=mu,sd=sigma)" will return the probability density function evaluated at x, i.e., $f_W(x)$ for $W \sim \mathcal{N}(\mu,\sigma^2)$. You can run ?dnorm in the R console to find out more.

We shall consider a case where $\mu = 50 \cdot 0.7$ and $\sigma = \sqrt{50 \cdot 0.7 \cdot (1-0.7)}$. Use the "dnorm()" to generate a data-frame called "gaussian_df" with two columns called "x" and "pdf".

1. The first column contains the numbers 0,0.01,0.02, ⋯ ,50. These numbers represent different values of x.
2. The second column gives the corresponding value of the probability density function $f_W(x)$ with $W \sim \mathcal{N}(\mu, \sigma^2)$. Use the "head()" function to observe the first 3 rows as your data frame.

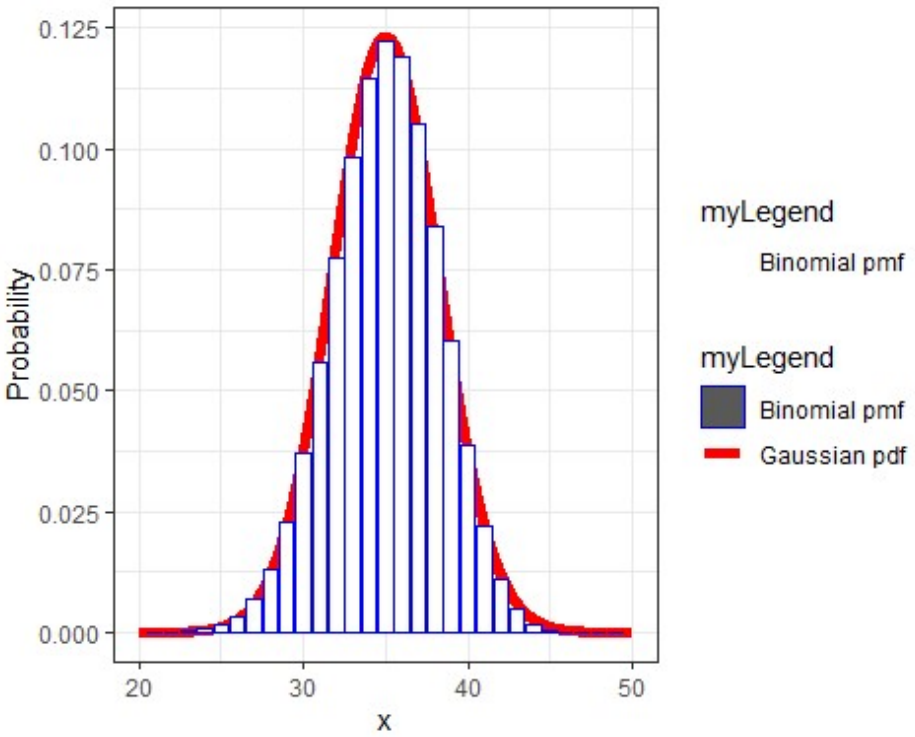
Answers

```
gaussian_df <- data.frame(x = seq(0, 50, 0.01)) %>%
  mutate(pdf = map_dbl(x, ~dnorm(.x, mean=50*0.7, sd=sqrt(50*0.7*(1-
0.7)) ) ) )
head(gaussian_df, 3)

##      x           pdf
## 1 0.00 5.707825e-27
## 2 0.01 5.901264e-27
## 3 0.02 6.101201e-27
```

(Q4)

Next, based on the "binom_df" and "gaussian_df" you generated above, use the following code to create a plot which compares the probability density for your Gaussian distribution $W \sim \mathcal{N}(\mu, \sigma^2)$ where $\mu = n \cdot p$ and $\sigma = \sqrt{n \cdot p \cdot (1-p)}$ and the probability mass function for your Binomial distribution $Z \sim \text{Binom}(n, p)$. Is this a consequence of the central limit theorem? Justify your answer.

```
colors<-c("Gaussian pdf"="red", "Binomial pmf"="blue")
fill<-c("Gaussian pdf"="white", "Binomial pmf"="white")


ggplot() + labs(x="x",y="Probability") + theme_bw() +
  # create plot of Gaussian density
  geom_line(data=gaussian_df, aes(x,y=pdf,color="Gaussian pdf"),size=2)
+
  # create a bar chart from PMF of Binomial distribution
  geom_col(data=binom_df, aes(x=x,y=pmf, color="Binomial
pmf",fill="Binomial pmf")) +
  # set color
  scale_color_manual(name = "myLegend", values=colors) +
  scale_fill_manual(name = "myLegend", values=fill) +
  xlim(c(20,50))
```



Answers

Since $Z \sim \text{Binom}(n, p)$, we can write out $Z = X_1 + \cdots X_n$ where $X_1, X_2, \cdots, X_n$ are independent copies of $X \sim \mathcal{B}(p)$. So $\mathbb{E}(X) = p$ and $\text{Var}(X) = p(1-p)$. Let $W \sim \mathcal{N}\big(np, np(1-p)\big)$ be a Gaussian random variable. Hence

$$\widetilde{W} = \frac{W - np}{\sqrt{np(1-p)}}$$

is a standard Gaussian random variable i.e. $\widetilde{W} \sim \mathcal{N}(0,1)$.

Hence, by the central limit theorem, for each $t \in \mathbb{R}$, we have

$$
\begin{aligned}
& \lim_{n\to\infty}\mathbb{P}\left(Z \le np + t\sqrt{np(1-p)}\right) \\
= \quad & \lim_{n\to\infty}\mathbb{P}\left(\sum_{i=1}^{n} X_i \le np + t\sqrt{np(1-p)}\right) \\
= \quad & \lim_{n\to\infty}\mathbb{P}\left\{\sqrt{\frac{n}{p(1-p)}}\left(\frac{1}{n}\sum_{i=1}^{n} X_i - p\right) \le t\right\} \\
= \quad & \mathbb{P}\left(\widetilde{W} \le t\right) \\
= \quad & \mathbb{P}\left(\frac{W - np}{\sqrt{np(1-p)}} \le t\right) \\
= \quad & \mathbb{P}\left(W \le np + t \cdot \sqrt{np(1-p)}.\right)
\end{aligned}
$$

Hence, for sufficiently large n, we expect $Z \sim \text{Binom}(n, p)$ to be well-approximated by $W \sim \mathcal{N}\left(np, np(1-p)\right)$.