Lab3

Exercises

Irimie Fabio

Contents

Exercise 1

A

Create the Lab3 project. Use the same structure used for Lab1 and Lab2: scripts, plots and data directories.

B

Write a function to calculate the sum of integer numbers from 1 to n

```
sum_integer <- function(n) {
  sum <- 0
  for (i in 1:n) {
    sum <- sum + i
  }
```

```r
  return(sum)
}

cat("The sum of the first 10 integers is: ", sum_integer(10), "\n")
```

```
## The sum of the first 10 integers is:  55
```

C

Write a function to calculate the product of integers from 1 to n, also known as n!

```r
prod_integer <- function(n) {
  val <- n
  for (i in (n - 1):1) {
    val <- val * i
  }
  return(val)
}

cat("The factorial of 5 is: ", prod_integer(5), "\n")
```

```
## The factorial of 5 is:  120
```

D

Try C. but do it recursively (hint: call the function itself inside the loop, remember to return 1 when n is equal to 0)

```r
factorial <- function(n) {
  if (n == 0) {
    return(1)
  } else {
    val <- n * factorial(n - 1)
  }

  return(val)
}

cat("The factorial of 5 is: ", factorial(5), "\n")
```

```
## The factorial of 5 is:  120
```

Exercise 2

A

Simulate the tossing of a fair dice and verify through the definition that the event $E = \{2, 3\}$ has probability $\frac{1}{3}$. $S = \{1, 2, 3, 4, 5, 6\}$; $E = \{2, 3\}$; $P(E) = \frac{1}{3}$

(hint: generate a sequence of integer random numbers between 1 and 6 using the sample() function)

```r
library(ggplot2)

n <- 100000 # Number of experiments
e <- c(2, 3) # Event of interest
```

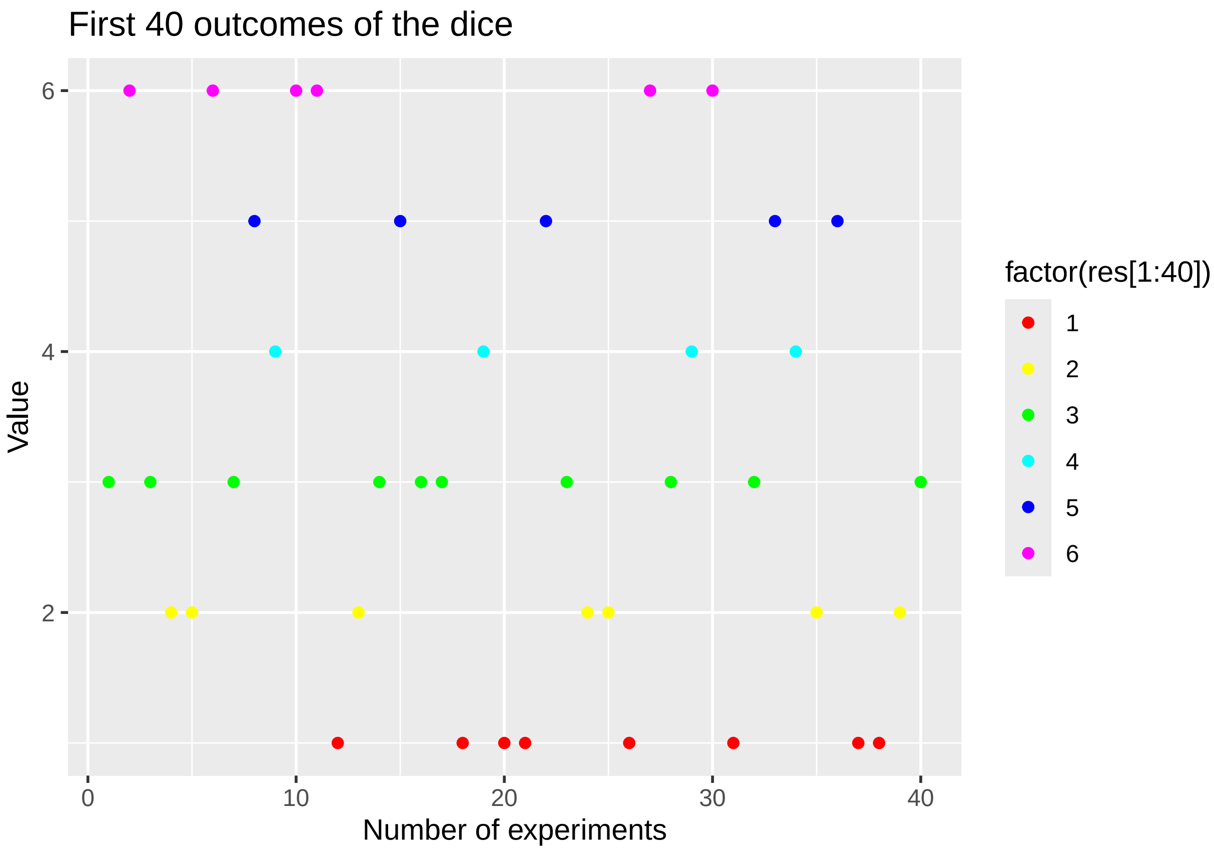
```r
# Outcomes of interest
set.seed(123)
res <- sample(x = c(1:6), size = n, replace = TRUE)

# Outcomes of E (1 when in E, 0 otherwise)
ne <- ifelse(res %in% e, 1, 0)
```

B

Plot the first 40 outcomes of the experiment.

```r
ggplot(
  data = data.frame(x = 1:40, y = res[1:40]),
  aes(
    x = x,
    y = y,
    color = factor(res[1:40])
  )
) +
  geom_point() +
  scale_color_manual(values = rainbow(6)) +
  labs(
    title = "First 40 outcomes of the dice",
    x = "Number of experiments", y = "Value"
  )
```



C

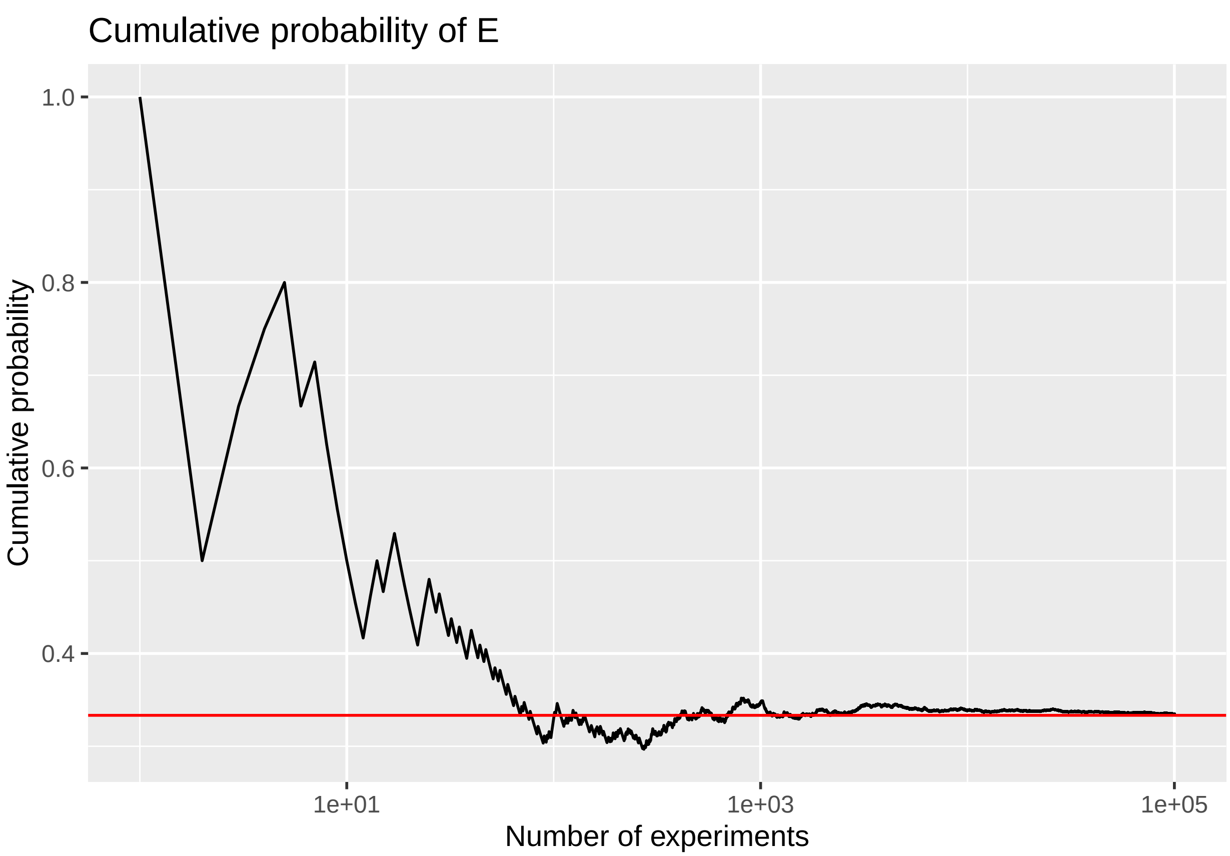
Plot the convergence of $P(E)$ at the value obtained from the classical definition $\frac{1}{3}$. (hint: the frequentist approach says that, as the number of trials approaches infinity, the relative frequency will converge exactly to the true probability)

```
# Probability of E
pe <- sum(ne) / n
cat("The probability of E is: ", pe, "\n")
```

```
## The probability of E is:  0.33474
```

```
cum_pe <- cumsum(ne) / 1:n

df <- data.frame(x = 1:n, y = cum_pe)
ggplot(data = df, aes(
  x = x, y = y,
)) +
  geom_line() +
  geom_hline(yintercept = 1 / 3, col = "red") +
  scale_x_continuous(trans = "log10") +
  labs(
    title = "Cumulative probability of E",
    x = "Number of experiments", y = "Cumulative probability"
  )
```



Exercise 3

A

Simulate the tossing of a fair dice and consider the following events: $A = \{1, 2\}$; $B = \{2, 3, 6\}$; $C = \{1, 4, 5\}$. (hint: compute $P(A), P(B), P(C)$).

```
n <- 100000 # Number of experiments

a <- c(1, 2) # Event A
b <- c(2, 3, 6) # Event B
c <- c(1, 4, 5) # Event C

res <- sample(x = c(1:6), size = n, replace = TRUE)

pa <- sum(res %in% a) / n
pa
## [1] 0.33191
pb <- sum(res %in% b) / n
pb
## [1] 0.50041
pc <- sum(res %in% c) / n
pc
## [1] 0.49959
```

B

Verify that A and B are independent and that B and C are dependent.

```
# A and B
pab <- sum(res %in% a & res %in% b) / n

# B and C
pbc <- sum(res %in% b & res %in% c) / n

cat("A and B are independent: ", pab == pa * pb, "\n")
```

```
## A and B are independent:  FALSE
```

```
cat("B and C are dependent: ", pbc != pb * pc, "\n")
```

```
## B and C are dependent:  TRUE
```

Exercise 4

A

Generate a sequence of $N = 10000$ random numbers that simulate the throwing of a dice.

```
n <- 10000
set.seed(123)
res <- sample(x = c(1:6), size = n, replace = TRUE)
```

B

Then simulate the throwing of a second dice.

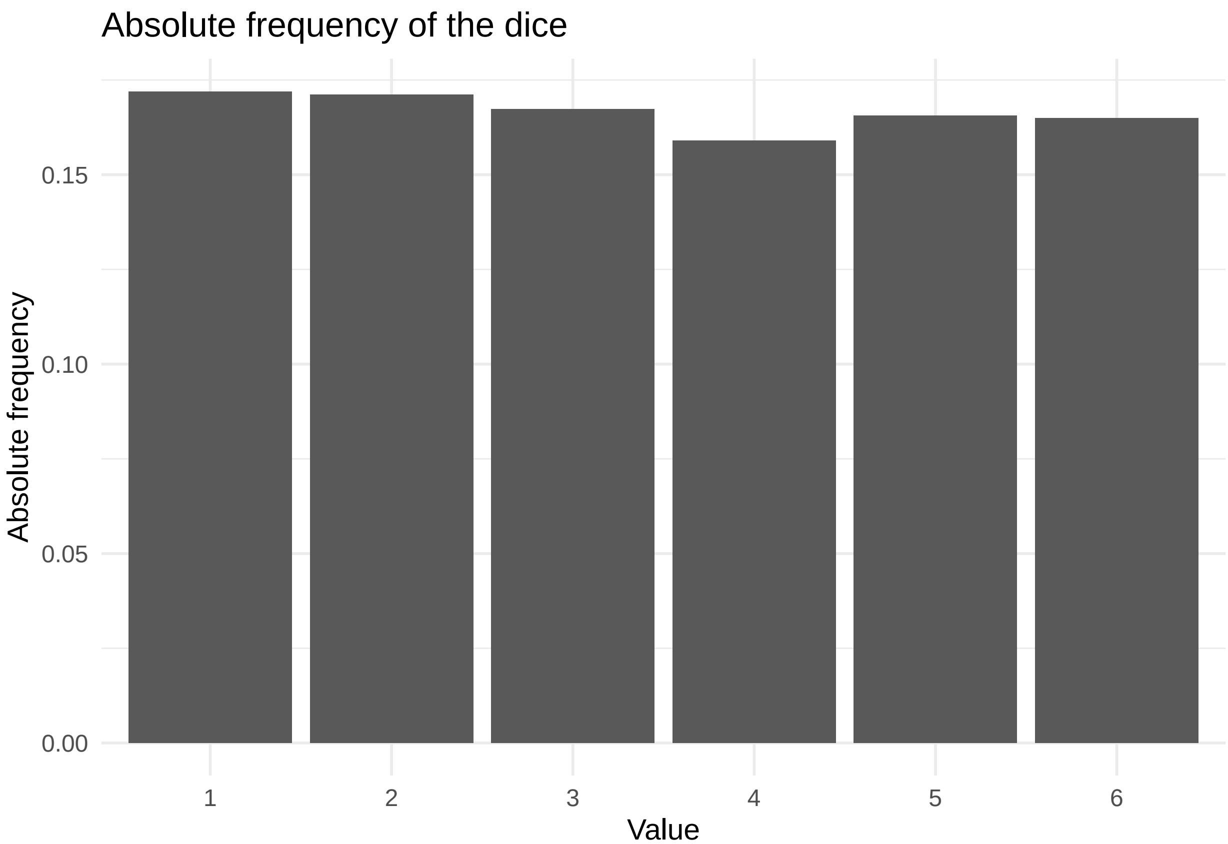
```r
res2 <- sample(x = c(1:6), size = n, replace = TRUE)
```

C

Plot the absolute and relative frequencies for A. and the relative frequency for the sum of the two dice for point B. using the geom_bar() or geom_col() functions.

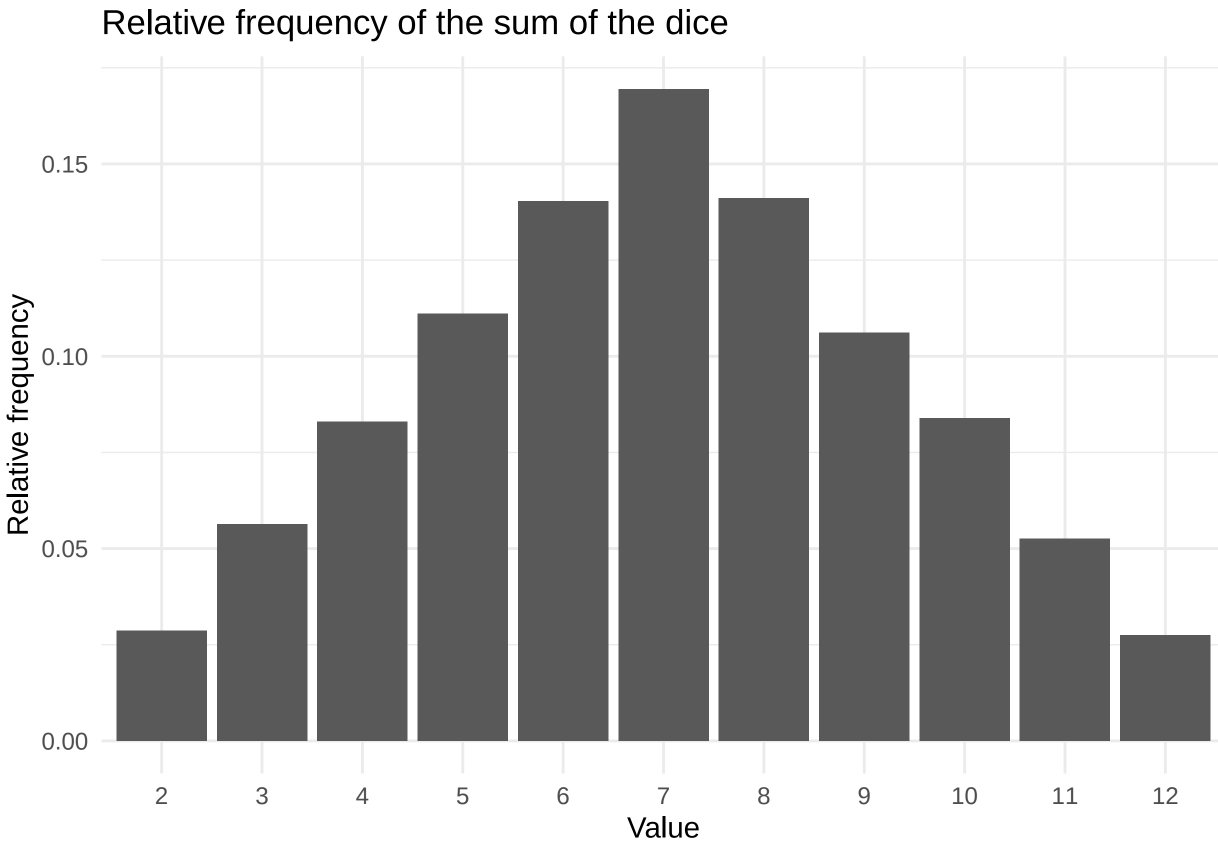
```r
df1 <- data.frame(x = 1:6, y = table(res) / n)
df2 <- data.frame(x = 2:12, y = table(res + res2) / n)

ggplot(data = df1, aes(
  x = factor(x),
  y = y.Freq
)) +
  geom_col() +
  labs(
    title = "Absolute frequency of the dice",
    x = "Value", y = "Absolute frequency"
  ) +
  theme_minimal()
```




```r
ggplot(data = df2, aes(
  x = factor(x),
  y = y.Freq
)) +
  geom_col() +
  labs(
```

```
    title = "Relative frequency of the sum of the dice",
    x = "Value", y = "Relative frequency"
  ) +
  theme_minimal()
```



Exercise 5

A

Four people are in a room. What is the probability that no two of them celebrate their birthday on the same day of the year?

```
people <- 4
days <- 365

n_bdays <- choose(days, people) * factorial(people)
p_bdays <- n_bdays / days^people
p_bdays
```

```
## [1] 0.9836441
```

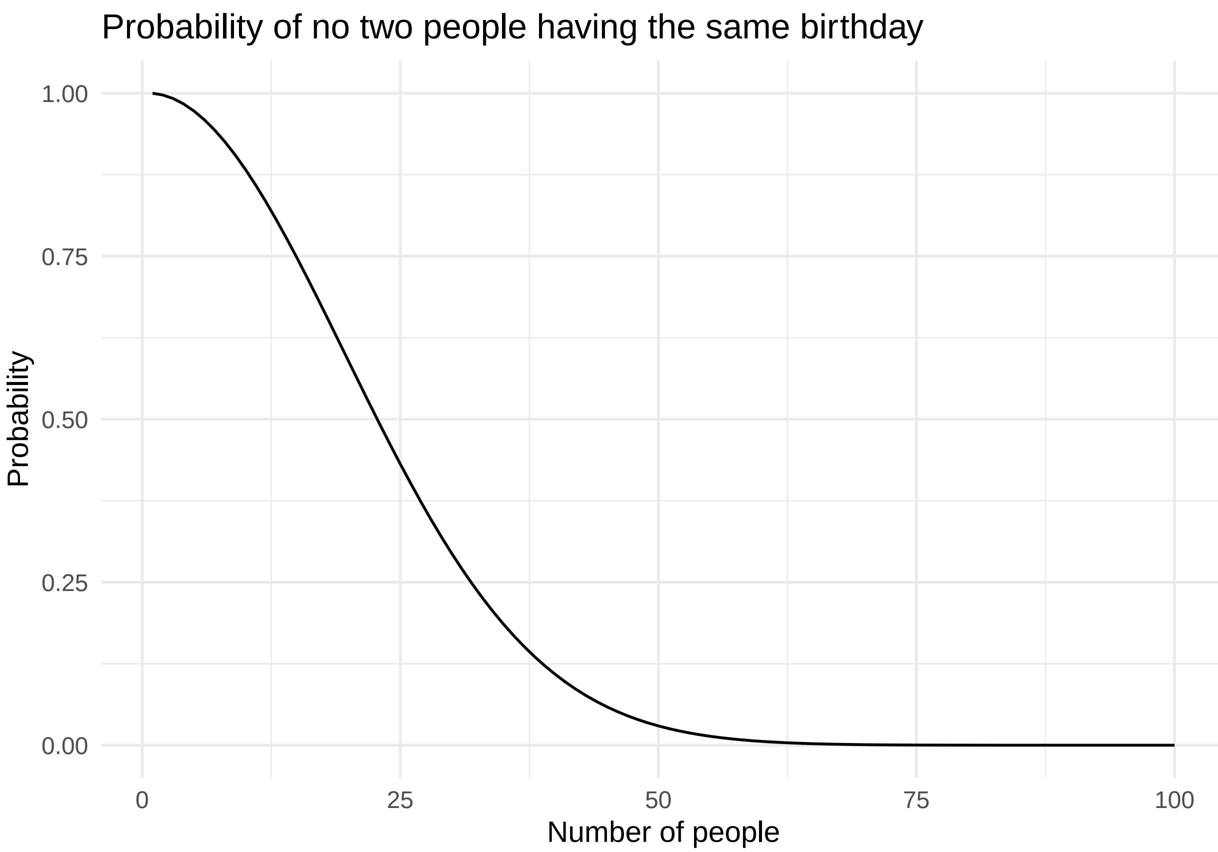
B

n people are in a room. What is the probability that no two of them celebrate their birthday on the same day of the year? Try this with n from 1 to 100 and plot the probability for each value of n.

```
n <- 100
p_bdays <- rep(NA, n)
```

```r
for (i in 1:n) {
  p_bdays[i] <- choose(days, i) * factorial(i) / days^i
}

df <- data.frame(x = 1:n, y = p_bdays)
ggplot(data = df, aes(
  x = x, y = y
)) +
  geom_line() +
  labs(
    title = "Probability of no two people having the same birthday",
    x = "Number of people", y = "Probability"
  ) +
  theme_minimal()
```



Exercise 6

A laboratory blood test is 99% effective in detecting a certain disease when it is, in fact, present. However, the test also yields a "false positive" (FP – Type I error) result for 1% of the healthy persons tested. If 0.5% of the population actually has the disease, what is the probability a person has the disease given that his test result is positive? (Hint: Use D as the event "the tested person has the disease" and E as "The result of the test is positive").

```r
p_d <- 0.005
p_e_d <- 0.99
p_e_nd <- 0.01
```

```r
p_d_e <- p_d * p_e_d / (p_d * p_e_d + (1 - p_d) * p_e_nd)
p_d_e
```

```
## [1] 0.3322148
```