Problem set 6

2024-10-25

Please answer each of the exercises below. For those asking for a mathematical calculation please use LaTeX to show your work.

Important: Make sure that your document renders in less than 5 minutes.

1. Write a function called `same_birthday` that takes a number `n` as an argument, randomly generates `n` birthdays and returns `TRUE` if two or more birthdays are the same. You can assume nobody is born on February 29.

Hint: use the functions `sample`, `duplicated`, and `any`.

```
same_birthday <- function(n){
 x <- sample(1:365, n, replace = TRUE)

 return(any(duplicated(x)))
}
```

2. Suppose you are in a classroom with 50 people. If we assume this is a randomly selected group of 50 people, what is the chance that at least two people have the same birthday? Use a Monte Carlo simulation with $B=$1,000 trials based on the function `same_birthday` from the previous exercises.

```
B <- 10^3
n <- 50

probability <- mean(replicate(B, same_birthday(n)))
probability
```

```
[1] 0.97
```

3. Redo the previous exercises for several values on `n` to determine at what group size do the chances become greater than 50%. Set the seed at 1997.

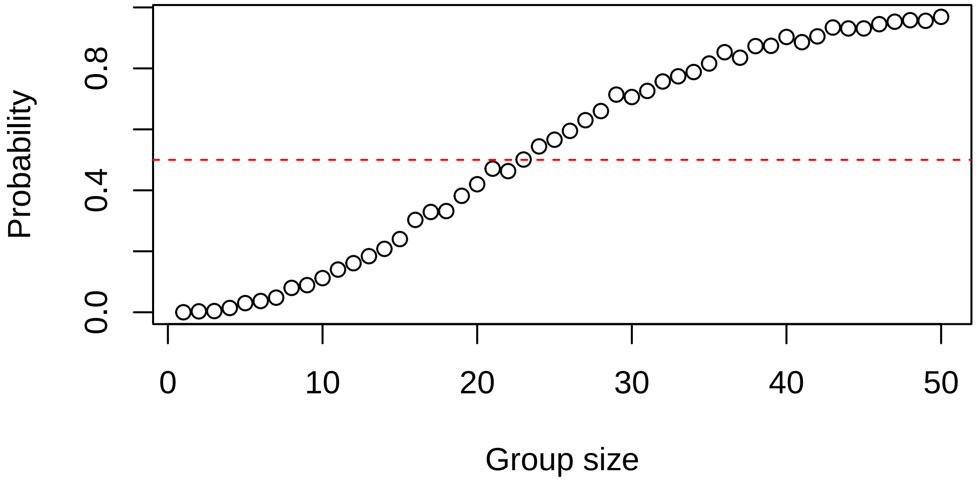
```
set.seed(1997)
compute_prob <- function(n, B = 10^3) {

  results <- replicate(B, same_birthday(n))
  mean(results)
}

prob <- which(sapply(1:n, compute_prob)>0.5)[1]
prob
```

```
[1] 23
```

```
plot(1:n, sapply(1:n, compute_prob), xlab = "Group size", ylab = "Probability")
abline(h=0.5, col = "red", lty = 2)
```



4. These probabilities can be computed exactly instead of relying on Monte Carlo approximations. We use the multiplication rule:

$$\text{Pr}(n \text{ different birthdays}) = 1 \times \frac{364}{365} \times \frac{363}{365} ... \frac{365-n+1}{365}$$

Plot the probabilities you obtained using Monte Carlos as a points and the exact probabilities with a red line.

Hint: use the function **prod** to compute the exact probabilities.

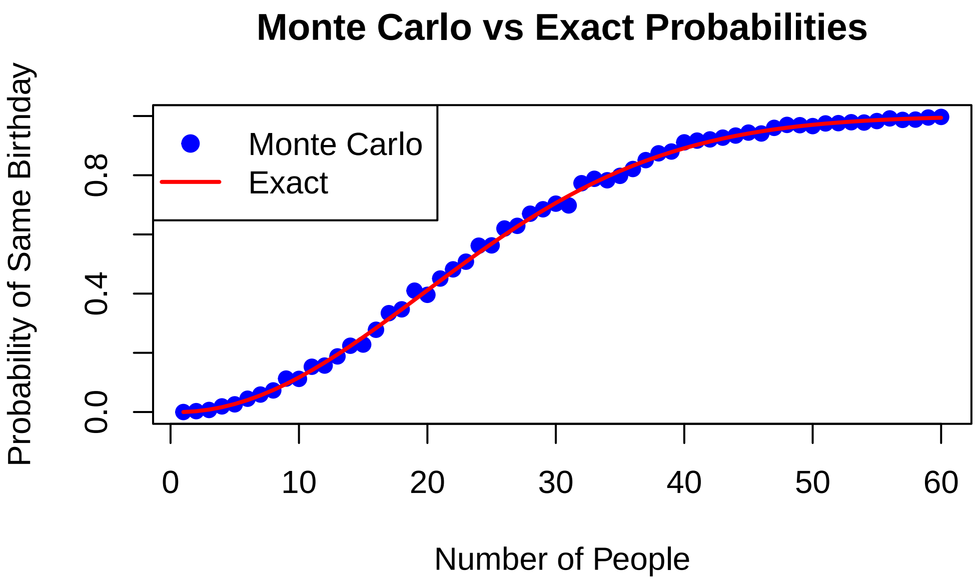
```
n <- seq(1,60)

exact_prob <- function(n){
  if(n>365) return (1)

  prob <- 1- prod((365: (365-n+1))/365)
  return(prob)
}

exact <- sapply(n, function(n) exact_prob(n))
mc<- sapply(n, function(n) compute_prob(n, B=10^3))

plot(n, mc, type = "p", pch = 19, col = "blue",
     xlab = "Number of People", ylab = "Probability of Same Birthday",
     main = "Monte Carlo vs Exact Probabilities")
lines(n, exact, col = "red", lwd = 2)  # Add exact probabilities as a red line
legend("topleft", legend = c("Monte Carlo", "Exact"), col = c("blue", "red"), pch = c(19, NA
```



5. Note that the points don't quite match the red line. This is because our Monte Carlos simulation was based on only 1,000 iterations. Repeat exercise 2 but for `n = 23` and try `B <- seq(10, 250, 5)^2` number iterations. Plot the estimated probability against `sqrt(b)`. Describe when it starts to stabilize in that the estimates are within 0.005 for the exact probability. Add horizontal lines around the exact probability ± 0.005. Note this could take several seconds to run. Set the seed to 1998.

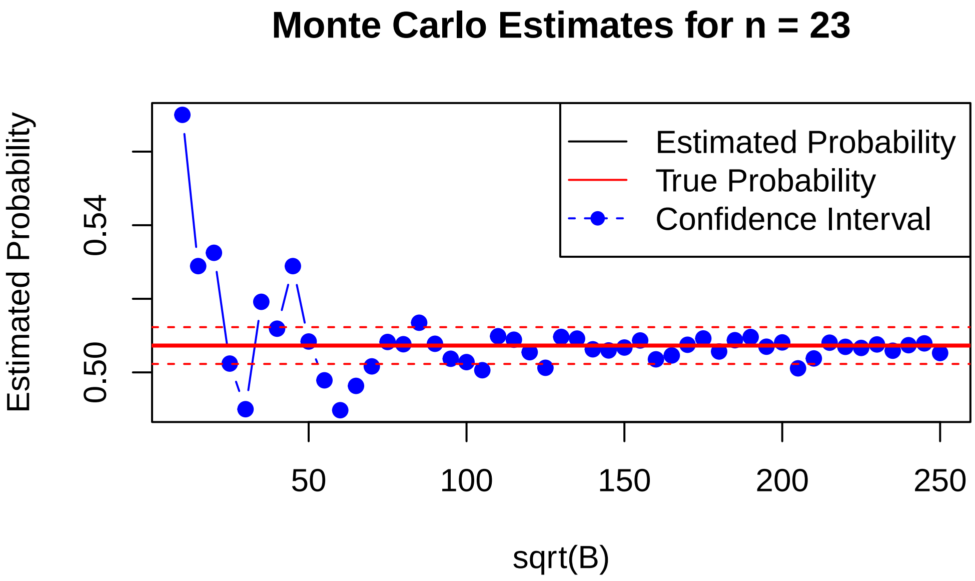
```r
set.seed(1998)
B <- seq(10, 250, 5)^2

exact_prob <- function(n){
  if(n>365) return (1)

  prob <- 1- prod((365: (365-n+1))/365)
  return(prob)
}

n <- 23
exact_p <- exact_prob(n=23)
mc<- sapply(B, function(i) compute_prob(n, i))

plot(sqrt(B), mc, type = "b", col = "blue", pch = 19,
     xlab = "sqrt(B)", ylab = "Estimated Probability",
     main = "Monte Carlo Estimates for n = 23")
abline(h = exact_p, col = "red", lwd = 2)
abline(h = exact_p-0.005,col = "red", lty = 2 )
abline(h = exact_p+0.005,col = "red", lty = 2 )
legend("topright", legend=c("Estimated Probability", "True Probability", "Confidence Interval
       col=c("black", "red", "blue"), lty=c(1, 1, 2), lwd=c(1,1,1), pch=c(NA, NA, 16))
```



```
for (i in B) {
  mc_p <- compute_prob(n, i)
  if (abs(mc_p - exact_p) <= 0.001) {
    print(i)
  }
}
```

```
[1] 2500
[1] 4900
[1] 18225
[1] 36100
[1] 44100
[1] 48400
[1] 52900
[1] 60025
[1] 62500
```

6. Repeat exercise 4 but use the the results of exercise 5 to select the number of iterations so that the points practically fall on the red curve.

Hint: If the number of iterations you chose is too large, you will achieve the correct plot but your document might not render in less than five minutes.

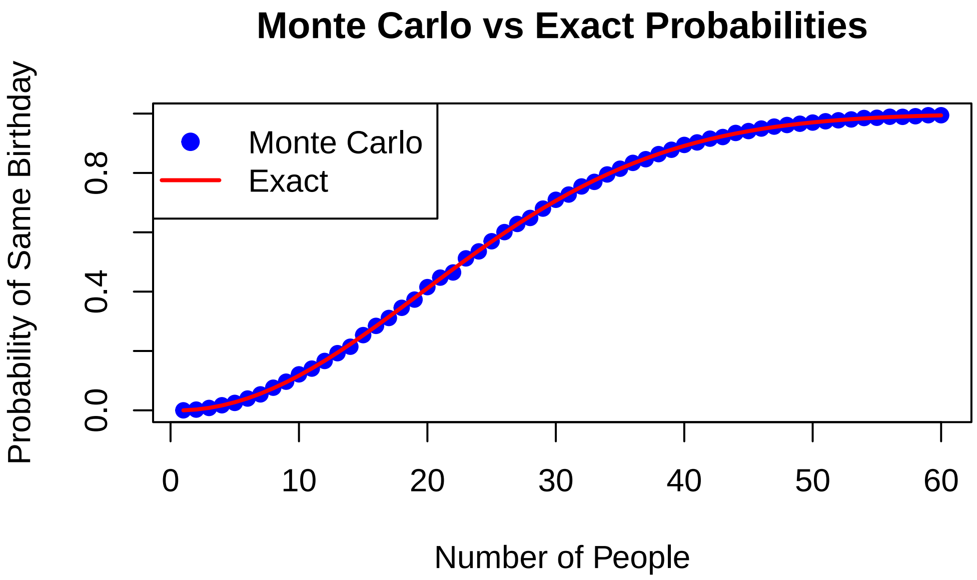
```
n <- seq(1,60)

exact_prob <- function(n){
  if(n>365) return (1)

  prob <- 1- prod((365: (365-n+1))/365)
  return(prob)
}

exact <- sapply(n, function(n) exact_prob(n))
mc<- sapply(n, function(n) compute_prob(n, B=18225))

plot(n, mc, type = "p", pch = 19, col = "blue",
     xlab = "Number of People", ylab = "Probability of Same Birthday",
     main = "Monte Carlo vs Exact Probabilities")
lines(n, exact, col = "red", lwd = 2)  # Add exact probabilities as a red line
legend("topleft", legend = c("Monte Carlo", "Exact"), col = c("blue", "red"), pch = c(19, NA
```



7. In American Roulette, with 18 red slots, 18 black slots, and 2 green slots (0 and 00), what is the probability of landing on a green slot?

$$\text{P(green slot)} = \frac{\text{Number of green slots}}{\text{Total number of slots}} = \frac{2}{38} = \frac{2}{18+18+2} = \frac{1}{19} \approx 0.0526$$

8. The payout for winning on green is $17 dollars. This means that if you bet a dollar and it lands on green, you get $17. If it lands on red or black you lose your dollar. Create a sampling model using `sample` to simulate the random variable X for the Casino's winnings.

```
n <- 1

casino_winnings <- function(n) {
  outcomes <- c(1, -17)
  probs <- c(36/38, 2/38)

  total_winnings <- sum(sample(outcomes, size = n, replace = TRUE, prob = probs))
  return(total_winnings)
}

X = casino_winnings(n)
X
```

```
[1] 1
```

9. Now create a random variable S of the Casino's total winnings if $n = $1,000 people bet on green. Use Monte Carlo simulation to estimate the probability that the Casino loses money.

```
n <- 1000
S <- replicate(10000, casino_winnings(n))
cat("The first 10 replicate of S :", head(S,10),'\n')
```

```
The first 10 replicate of S : 154 82 -98 82 64 100 118 136 316 280
```

```
probability <- mean(replicate(10000, casino_winnings(n)<0))
cat("The probability that the Casino loses money :", probability)
```

```
The probability that the Casino loses money : 0.3246
```

10. What is the expected value of X?

$$\mathbb{E}[X] = \mu = \sum(x \cdot P(x)) = 1 \times \frac{36}{38} - 17 \times \frac{2}{38} = \frac{-34}{38} + \frac{36}{38} = \frac{1}{19} \approx 0.0526$$

11. What is the standard error of X?

$$\mathbb{E}[X^2] = (1)^2 \times \frac{36}{38} + (-17)^2 \times \frac{2}{38} \approx 16.1579$$

$$\mathrm{Var}(X) = \mathbb{E}[X^2] - (\mathbb{E}[X])^2 = 16.1579 - 0.0526^2 = 16.15513$$
$$\mathrm{SE}(X) = \sqrt{\mathrm{Var}(X)} = \sqrt{16.15513} \approx 4.02$$

12. What is the expected value of S? Does the Monte Carlo simulation confirm this?

Since S is the sum of 1,000 independent bets, the expected value of S is:

$$\mathbb{E}[S] = n\mathbb{E}[X] = 1000 * \frac{1}{19} \approx 52.63$$

Monte Carlos simulation

```
monte_carlo_mean <- mean(replicate(10000, casino_winnings(n)))
monte_carlo_mean
```

```
[1] 53.7598
```

The expected value of S is as above shown, and the Monte Carlos simulation confirm this with its estimated expected value very close to what I calculated here.

13. What is the standard error of S? Does the Monte Carlos simulation confirm this?

Since S is the sum of 1,000 independent bets, the expected value of S is:

$$\mathrm{Var}(S) = n\mathrm{Var}(X) = 1000 * 16.15513 = 16155.13\mathrm{SE}(S) = \sqrt{\mathrm{Var}(S)} = \sqrt{16155.13} \approx 127.10$$

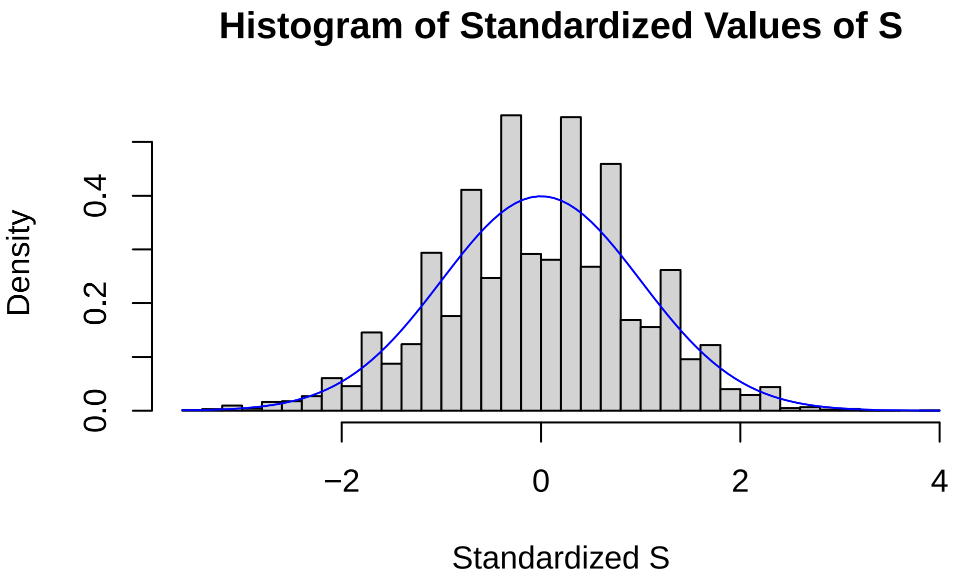
```
monte_carlo_sd <- sd(replicate(10000, casino_winnings(n)))
monte_carlo_sd
```

```
[1] 127.19
```

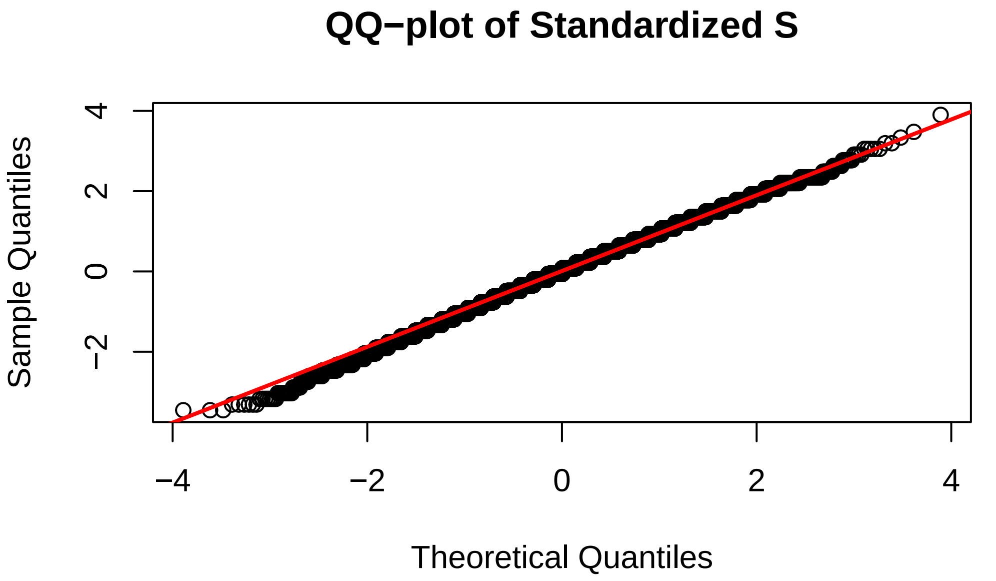
The standard error of S is as above shown, and the Monte Carlos simulation confirm this with estimated standard error very close to what I calculated here.

14. Use data visualization to convince yourself that the distribution of S is approximately normal. Make a histogram and a QQ-plot of standardized values of S. The QQ-plot should be on the identity line.

```r
standardized_S <- (replicate(10000, casino_winnings(n)) - monte_carlo_mean) / monte_carlo_sd
hist(standardized_S, probability = TRUE, main = "Histogram of Standardized Values of S",
     xlab = "Standardized S", breaks = 30)
curve(dnorm, col = "blue", add = TRUE)
```



```r
qqnorm(standardized_S, main = "QQ-plot of Standardized S")
qqline(standardized_S, col = "red", lwd = 2)
```



In the histogram, the shape of the histogram closely follows the bell curve of a normal distribution, suggesting that the distribution of S is approximately normal. In the QQ-plot, the points lie close to the red identity line, indicating that the distribution of S is well approximated by a normal distribution.

15. Notice that the normal approximation is slightly off for the tails of the distribution. What would make this better? Increasing the number of people playing n or the number of Monte Carlo iterations B?

Increasing the number of people betting (n) will make the distribution of S more normal. This is due to the Central Limit Theorem (CLT), which states that when the sample size is large, the probability distribution of the sum of the independent draws is approximately normal. Therefore, making n larger will help the distribution more approach normal distribution.

Increasing the number of Monte Carlo iterations (B) improves the accuracy of the simulation but doesn't change the shape of the distribution of S. It just gives a more precise estimate of the distribution.

16. Now approximate the probability estimated using CLT. Does it agree with the Monte Carlo simulation?

$$P(S < 0) = P\left(Z < \frac{0 - 52.63}{127.10}\right) = P\left(Z < -0.414\right)$$

```r
prob_z = pnorm(-0.414)
prob_z
```

```
[1] 0.3394371
```

It agrees with the Monte Carlo simulation. In question 9, we used Monte Carlo simulation to estimate the probability that the Casino loses money and the value is very close to what we calculated here (p = 0.3394).

17. How many people n must bet on green for the Casino to reduce the probability of losing money to 1%. Check your answer with a Monte Carlo simulation.

$$P(S < 0) = 0.01P\left(Z < \frac{0 - n\mathbb{E}[X]}{\sqrt{n} * \mathrm{SE}(X)}\right) = 0.01$$

From qnorm(0.01) we know the z-score for a 1% left-tail probability is approximately -2.33.

$$\frac{0 - n\mathbb{E}[X]}{\sqrt{n} * \mathrm{SE}(X)} = -2.33$$

$$n = [2.33 * \mathrm{SE}(X)/\mathbb{E}[X]]^2 = [2.33 * 4.02/0.526]^2 = 31611$$

```r
z_value <- qnorm(0.01)
expected_X <- 0.0526
se_X <- 4.02

n_required <- (z_value * se_X / expected_X)^2
cat("Number of people required to reduce the probability of losing to 1%:",
    ceiling(n_required))
```

```
Number of people required to reduce the probability of losing to 1%: 31611
```

Check my answer with a Monte Carlo simulation:

```r
simulations <- replicate(10000, casino_winnings(31611))
prob_casino_loses <- mean(simulations < 0)
cat("Estimated probability that the Casino loses money with n =", n, ":",
    prob_casino_loses)
```

```
Estimated probability that the Casino loses money with n = 1000 : 0.0107
```

Monte Carlo confirm n = 31611 is the correct choice with probability it estimated $\approx$ 0.01.