

PSC 290 MC Simulations Homework 1

Mijke Rhemtulla

2024-09-26

Intro

Answer all the questions below. This homework is due on Oct 16 before class. If you don't like the idea of a homework due before 9am, then it is due on Oct 15 at midnight. Upload both this Rmd file (completed) and the PDF output. If you are having troubles knitting to pdf, try knitting to html instead. Be sure your troubles aren't due to Now plot a histogram of the values missing objects in the Rmd file (e.g., all code should run without producing errors).

You may work on this assignment with others, and you may ask for help (e.g., on Slack) while working. The work that you submit, though, must be your own.

Part 1: distribution functions in R

This part has you use the R functions `dnorm`, `qnorm`, `pnorm`, etc. to find likelihoods, quantiles, and probabilities.

Question 1

Let x be a variable that follows a normal distribution with mean = 3 and standard deviation = 17, i.e., $x \sim N(3, 17)$. What proportion of the distribution of x falls below 20?

```
pnorm(20, mean = 3, sd = 17)
```

```
## [1] 0.8413447
```

Question 2

Let y be a variable that follows a normal distribution with mean = 24 and standard deviation = 3, i.e., $x \sim N(24, 3)$. What value of y cuts off the upper 5% of the distribution?

```
qnorm(p = .95, mean = 24, sd = 3)
```

```
## [1] 28.93456
```

Question 3

Let $p \sim N(10, 6)$. How much more likely is it that you would observe a value of 12 compared to a value of 14?

```
p <- dnorm(c(12, 14), mean = 10, sd = 6)
p[1]/p[2]
```

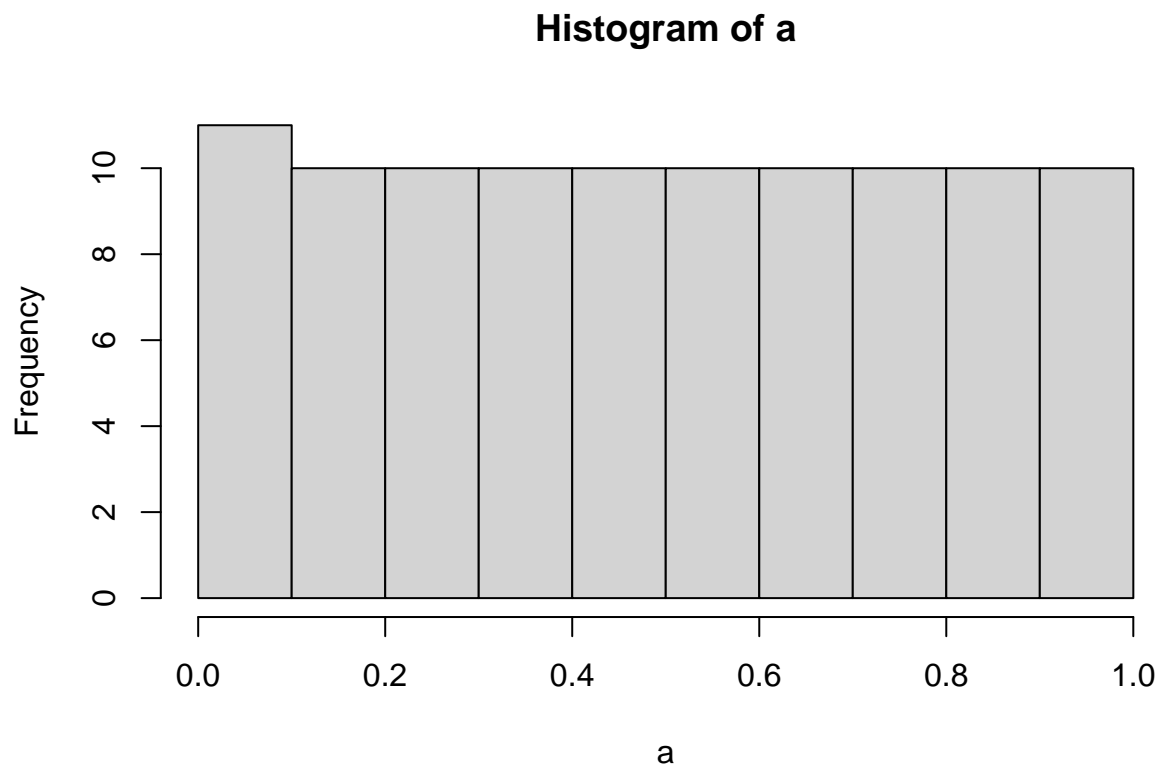
```
## [1] 1.18136
```

Part 2: Understanding PDFs and CDFs

Question 4

Using the `seq()` function, generate a sequence of numbers from 0 to 1, in increments of .01. Name this object `a`. Plot a histogram of the values in the vector you just created.

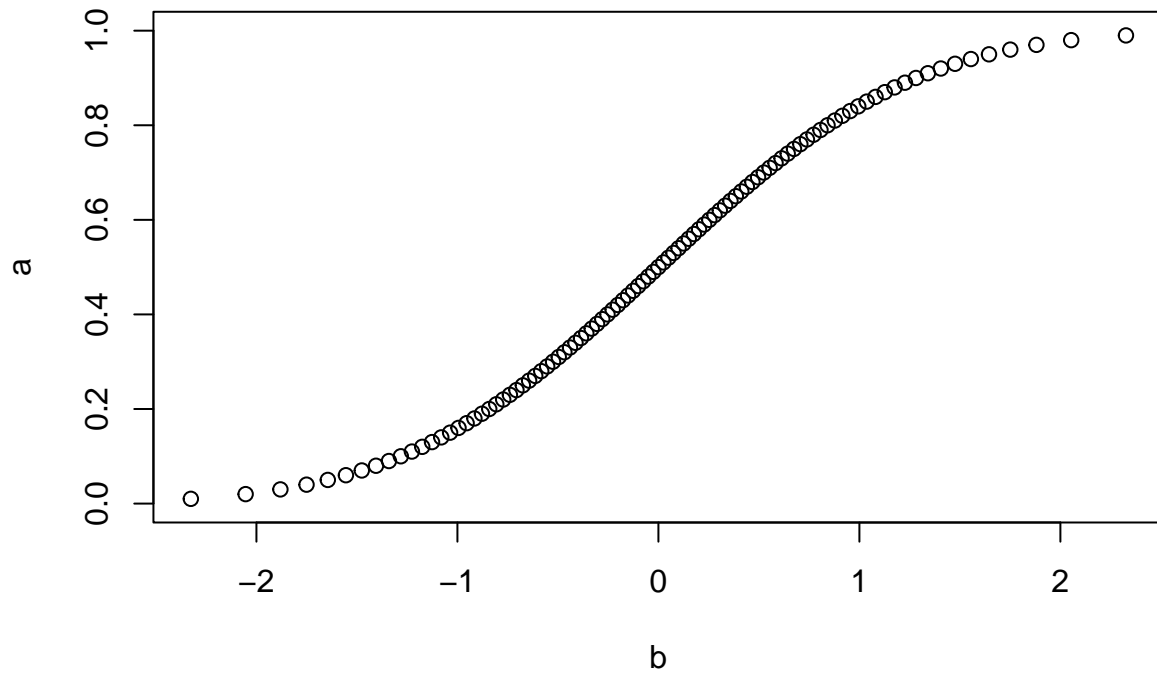
```
a <- seq(0, 1, by = .01)
hist(a)
```



Question 5

Apply the `qnorm` function to the vector of numbers you just created, and call this new object `b`. Make a scatterplot (or line graphs with points) with `b` on the x-axis and `a` on the y-axis. What is this plot called, and what does it show?

```
b <- qnorm(a)
plot(b, a)
```

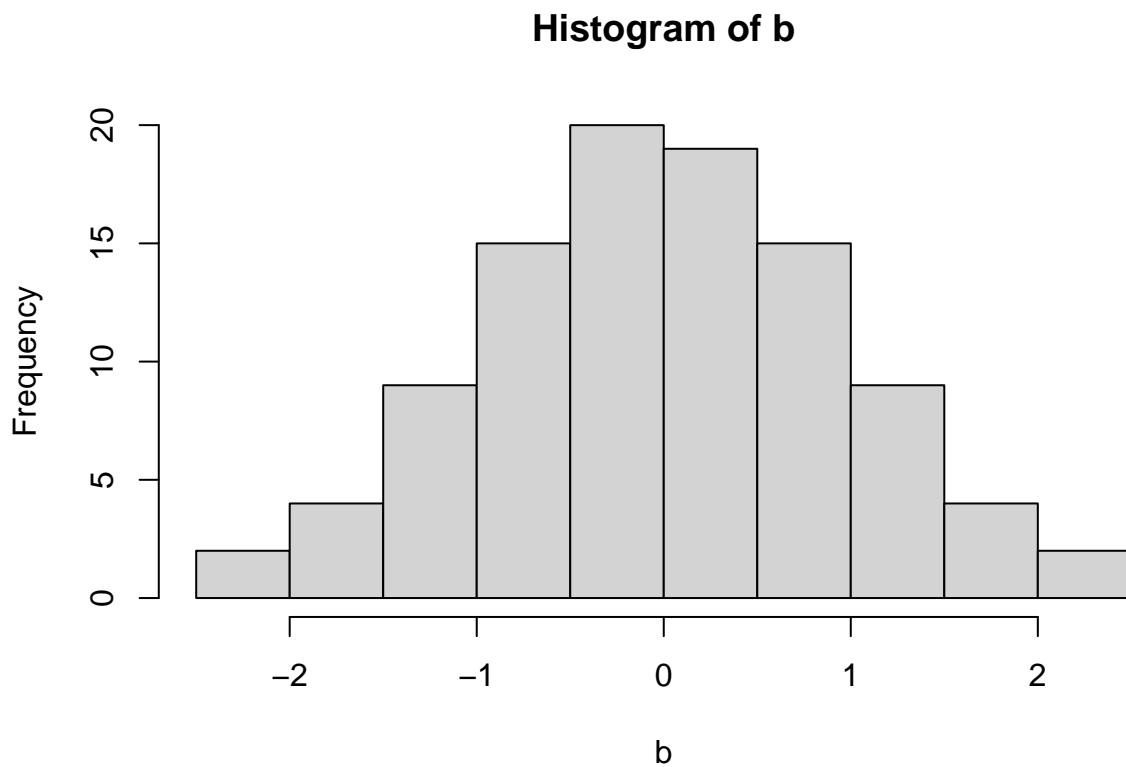


Sigmoidal curve.

Question 6

Now plot a histogram of the values in **b**. What does this distribution look like? Explain what happened.

```
hist(b)
```



Looks like a normal distribution. **a** is a vector of quantiles ranging from 0 to 1. To create **b** we map the quantiles defined in **a** in a standard normal distribution. The histogram of **b** is a this standard normal distribution. [rough draft]

Part 3: Generating Random Data with different distributional forms

Question 7

The previous question showed us how to transform a sequence of probabilities, uniformly distributed from 0-1, into a vector of quantiles that follow a normal distribution. We can use that same method to generate a *random* variable that follows a normal distribution, by first generating random uniform data.

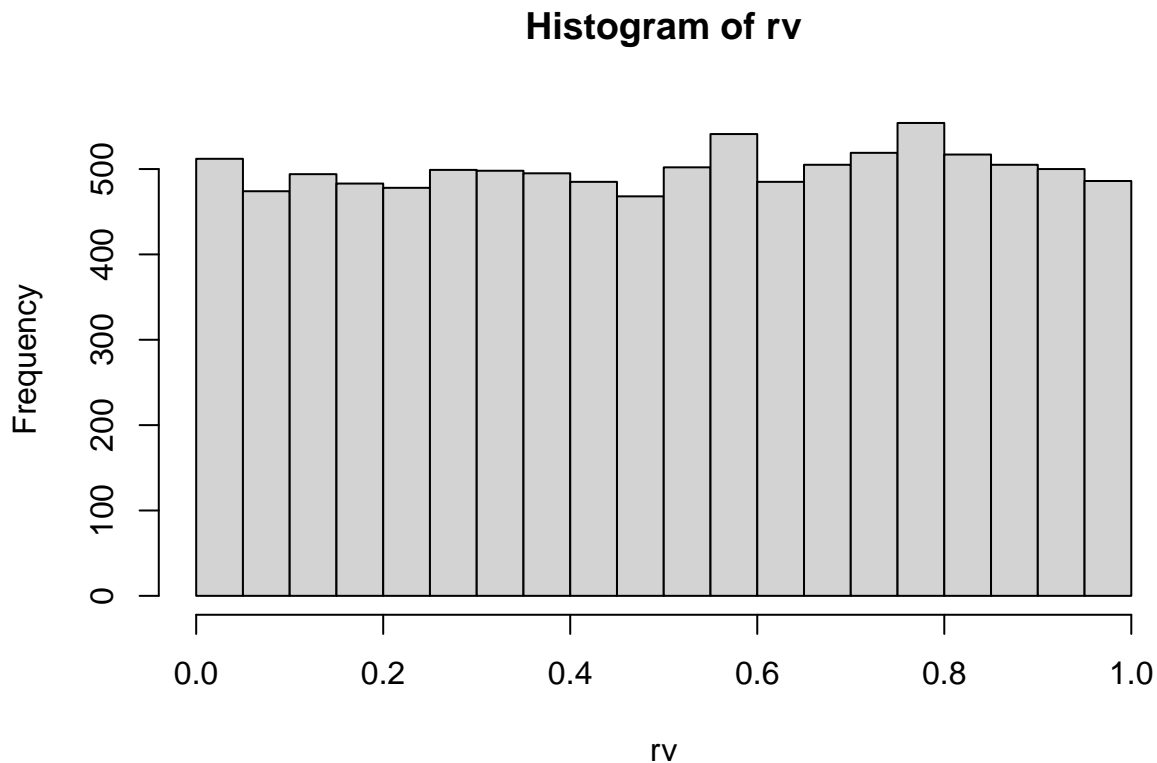
Generate a random variable of length 10000 (i.e., 10000 random values) from a uniform distribution with range (0, 1). Use `set.seed` to produce replicable results. Give your variable a name.

```
set.seed(890)
rv <- runif(10000, 0, 1)
```

Question 8

Use a histogram to plot the values of the variable you just generated. You can use the `hist()` function in base R, or use `ggplot` and make it pretty. Set the width of each histogram bun to be .05 units wide.

```
hist(rv)
```



Question 9

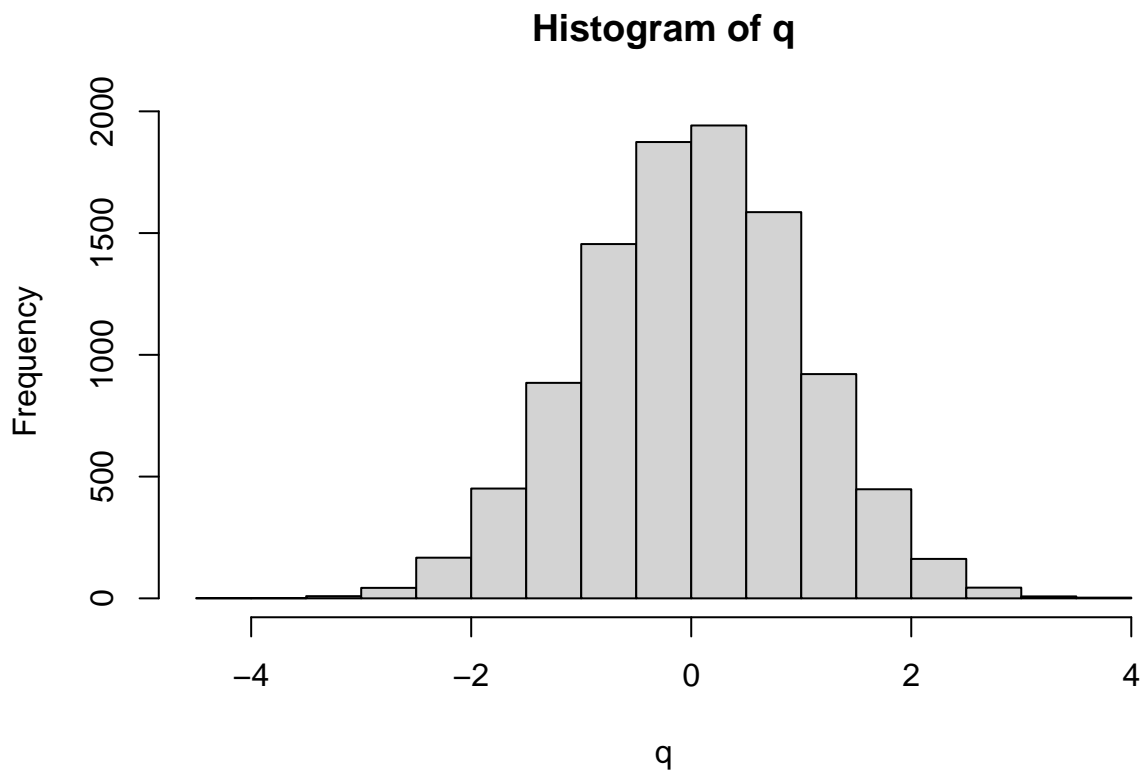
Use the `qnorm` function to find the quantiles of the normal distribution corresponding to each value that you generated. Save these quantiles in a new named object.

```
q <- qnorm(rv)
```

Question 10

Use a histogram to plot the quantiles that you found in the previous question. Again, you can use the `hist()` function in base R, or use `ggplot` and make it pretty. Whatever you use, give your axes interpretable labels.

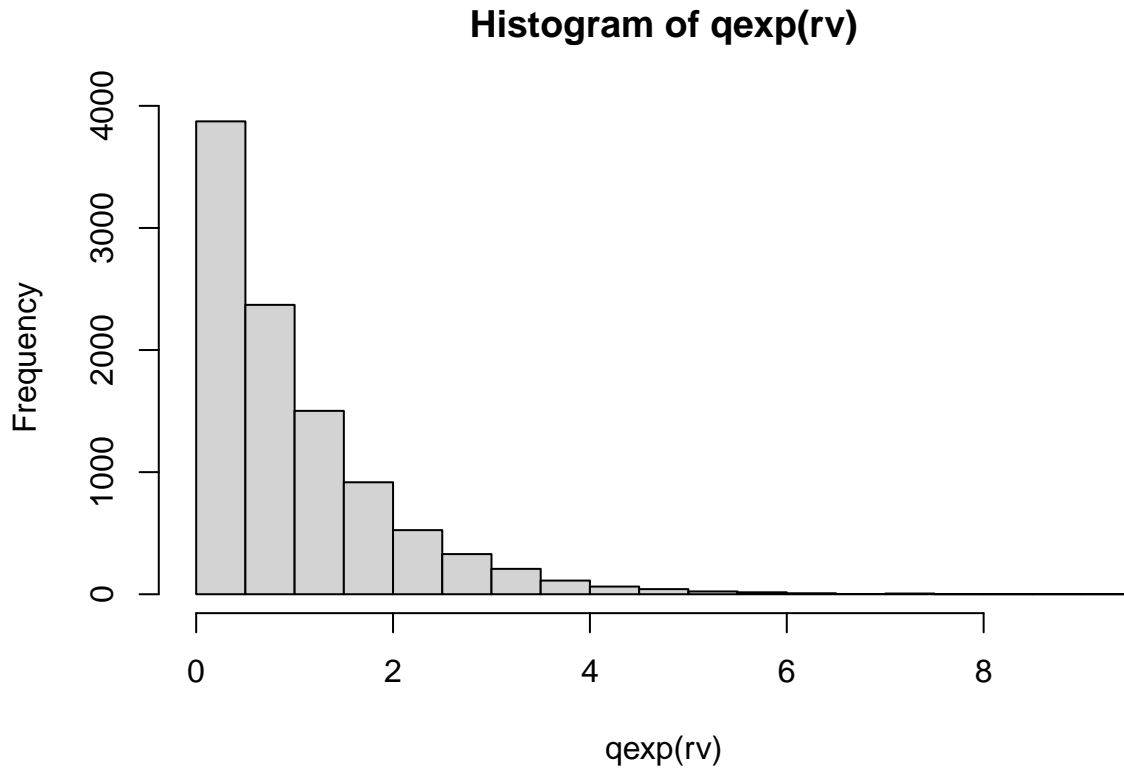
```
hist(q)
```



Question 11

That was cool. Take your same original random uniform variable, and give it a different distribution. Plot a histogram of your transformed values to check that your method worked.

```
hist(qexp(rv))
```



Part 4: some mini-simulations to answer questions

Question 12

In class you simulated data to sum up a number of die rolls. Now you're going to do the same thing but write a function to do it. Your function should have 2 arguments: N is the size of the sample (i.e., the number of dice that are rolled), and w is the number of sides on the die (i.e., the number of possible outcomes the die can take). Your function should return the sum or average of the N rolls. Run your function to demonstrate that it works.

```
die_roll <- function(N = 1, w = 6) {  
  sides <- seq(1, w, by = 1)  
  res <- sample(sides, N, replace = TRUE)  
  return(mean(res))  
}
```

Question 13

How many rolls of a 6-sided die do you need to add together (or average) before you get a distribution that behaves more or less like a normal distribution? You could use many different criteria to define “behaves more or less like a normal distribution”. Pick one that can be quantified in some way (“I eyeballed the histogram and it looks normal to me” is not a quantifiable criterion) and justify your choice. Then run the simulation and report your result. Use the function that you write in the previous question in your simulation.

```

reps <- 10000

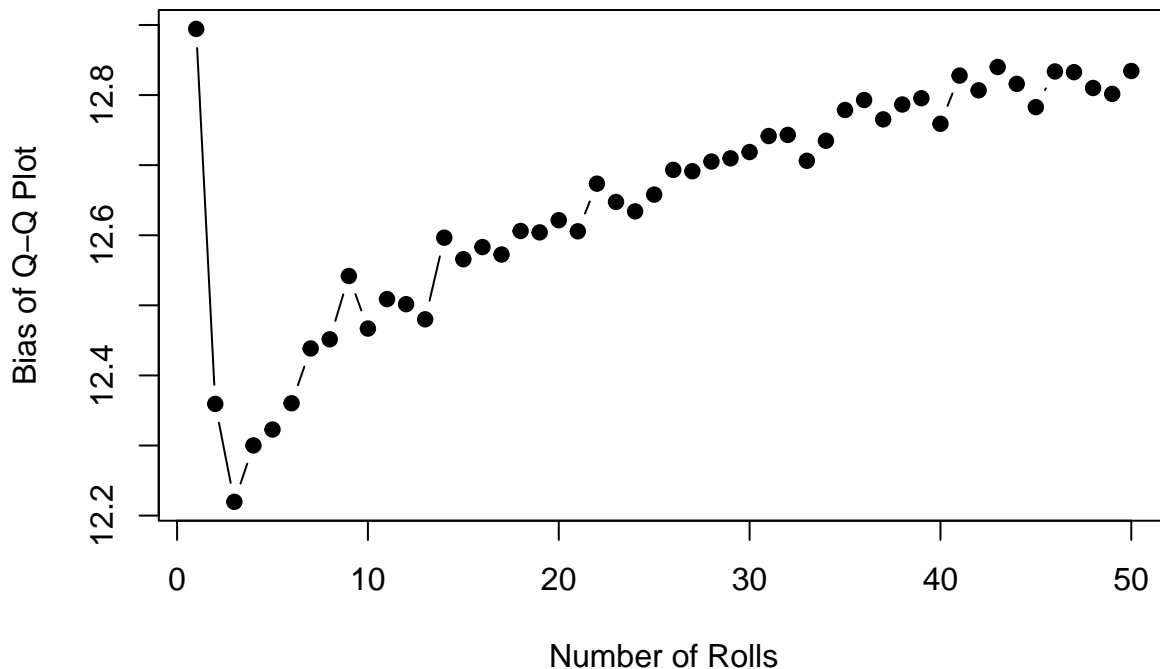
rolls_list <- 1:50

mse_qq <- sapply(rolls_list, function(n) {
  sim_data <- replicate(reps, die_roll(N = n))
  theoretical_quantiles <- qnorm(seq(1/(reps+1), 1 - 1/(reps+1), length.out = reps))
  sample_quantiles <- quantile(sim_data, probs = seq(1/(reps+1), 1 - 1/(reps+1), length.out = reps))
  return(mean((sample_quantiles - theoretical_quantiles)^2))
})

plot(rolls_list, mse_qq, type = "b", pch = 19,
     xlab = "Number of Rolls", ylab = "Bias of Q-Q Plot",
     main = "Bias vs Number of Rolls for Approximating Normality")
abline(h = min(mse_qq) * 1.1, col = "red", lty = 2)

```

Bias vs Number of Rolls for Approximating Normality



Question 14

At what sample size is it ok to use the standard normal (z) distribution rather than the t -distribution for hypothesis testing? Suppose that by “ok” we mean that the type-I error rate (rate of false positives) is within half a percent of the nominal .05 rate (assuming an alpha level of .05). Answer this question in two ways: First, do it analytically, using the `qnorm` and `pt` functions to find the proportion of various t -distributions beyond the critical values of the z -distribution. Assume that the degrees of freedom of the t -distribution are $N - 1$. Second, once you have found an answer, verify it by simulating a large number of samples from the t -distribution at the sample size you identified as being “ok”: what proportion of your simulated values fall in the tails past the critical z -values? Remember to set a random number seed to produce a replicable result. How much monte carlo error do you have (i.e., how accurate is your simulation result relative to your analytical result)?

```
## Analytical solution

## critical z values
lower_critical <- qnorm(0.05/2)
upper_critical <- qnorm(1 - 0.05/2)

# starting sample size
n <- 2

while (TRUE) {

  df <- n - 1
  # corresponding t values
  t_lower <- pt(lower_critical, df)
  t_upper <- 1 - pt(upper_critical, df)

  # error rate
  t_error <- t_lower + t_upper

  if (abs(t_error - 0.05) <= 0.005) {
    return(n)
  }
  n <- n + 1
}

#
print(n)
```

```
## [1] 57
```

```
## Simulation

reps <- 10000
tvals <- rt(reps, n - 1)

prop_tails <- mean(abs(tvals) > upper_critical)
prop_tails
```

```
## [1] 0.054
```

```
mc_error <- abs(prop_tails - 0.05)
mc_error
```

```
## [1] 0.004
```

The required sample size is 57.