Probability, Computation and Simulation Homework 4

Brian Cervantes Alvarez
November 1, 2024

Problem

I've read about the birthday problem, and how you only need 23 randomly chosen people for there to be a 50% chance that two people share a birthday. But how many people would you need for there to be a 50% chance that every possible birthday is represented by at least one person?

To solve this, we will estimate the probability that all birthdays (excluding February 29th and assuming each day is equally likely) are covered with `numPeople` people for various values of `numPeople`, identifying the value where the probability is 0.5.

Steps:

1. **Write a function to estimate** $p_M = P(\text{All birthdays are represented with } M \text{ people})$ using simulations.
2. **Try different values for `numPeople`** to find where p_M is approximately 0.2 and 0.8.
3. **Construct a data frame** with a range of `numPeople` values between those found in step 2.
4. **Add a new column `probHat`** with the estimated probs.
5. **Create a plot of `numPeople` versus the estimated probs**, including confidence intervals based on the Central Limit Theorem (CLT).
6. **Find the value of `numPeople`** that satisfies $p_M = 0.5$.
7. **Generalize steps 2–6** with different simulation numbers `nSim = 10000, 50000, 100000`.
8. **Compare the results** from different values of `nSim` and explain the findings.

1

We define a function `estBirthdayProb` that estimates the probability that all 365 birthdays are represented among `numPeople` individuals:

```r
# Load necessary libraries
library(purrr)
library(dplyr)
library(ggplot2)
```

```r
estBirthdayProb <- function(totBds = 365,
                            numPeople, nSim,
                            probs = rep(1, totBds) / totBds) {
  successes <- replicate(nSim, {
    sampBirthdays <- sample(1:totBds, numPeople,
                               replace = TRUE, prob = probs)
    length(unique(sampBirthdays)) == totBds
  })
  mean(successes)
}
```

2

We iteratively test different `numPeople` values to find where the estimated probability is approximately 0.2 and 0.8.

- Starting with `numPeople = 1200`:
  - $p_M(M = 1200) \approx 0.2$
- Testing `numPeople = 1800`:
  - $p_M(M = 1800) \approx 0.8$

3. Constructing the Data Frame

We create a sequence of `numPeople` values between 1200 and 1800:

```
numPeopleRange <- seq(1500, 2800, by = 50)
```

For each `numPeople` in `numPeopleRange`, we estimate `probHat` using `purrr::map_dbl`:

```
set.seed(202425)
nSim <- 10000
# Estimate probs
probEst <- map_dbl(numPeopleRange, ~ estBirthdayProb(numPeople = .x,
                                                     nSim = nSim))
simulationResults <- tibble(
  numPeople = numPeopleRange,
  probHat = probEst
)
```

Using the Central Limit Theorem, the `standardError` for each `probHat` is:

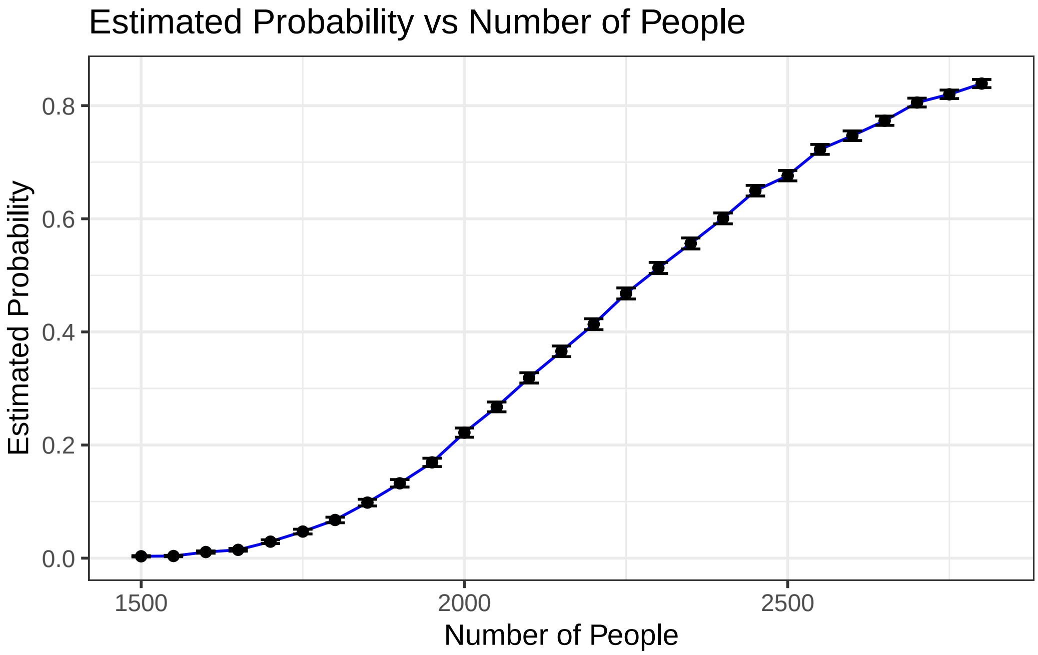
$$SE = \sqrt{\frac{\hat{p}(1-\hat{p})}{B}}$$

Thus, We can add the confidence intervals:

```
simulationResults <- simulationResults %>%
  mutate(
    standardError = sqrt(probHat * (1 - probHat) / nSim),
    lowerCI = probHat - 1.96 * standardError,
    upperCI = probHat + 1.96 * standardError
  )
```

After adding the CI, we can then plot the CDF:

```
ggplot(simulationResults, aes(x = numPeople, y = probHat)) +
  geom_line(color = "blue") +
  geom_point() +
  geom_errorbar(aes(ymin = lowerCI, ymax = upperCI), width = 30) +
  labs(
    title = "Estimated Probability vs Number of People",
    x = "Number of People",
    y = "Estimated Probability"
  ) +
  theme_bw()
```


Estimated Probability vs Number of People
Estimated Probability
0.0
0.2
0.4
0.6
0.8
1500
2000
2500
Number of People

6

We can interpolate to find `numPeople` where `probHat   0.5`:

```
interpolatedM <- approx(x = simulationResults$probHat,
                        y = simulationResults$numPeople, xout = 0.5)$y
interpolatedM
```

```
[1] 2285.556
```

From the interpolation, we find:

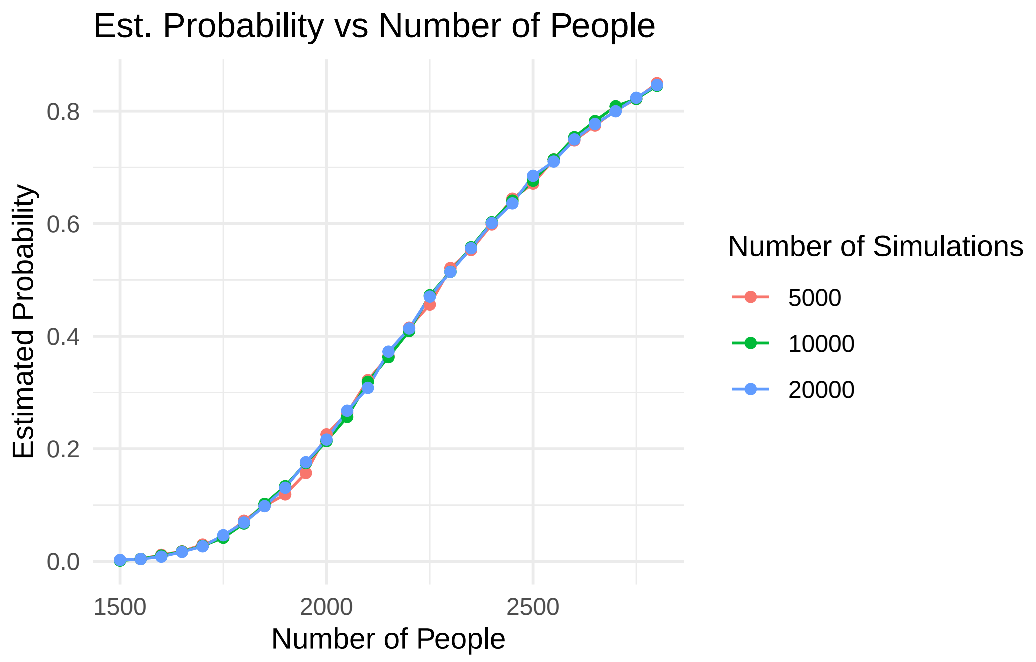
- `numPeople   2286` gives `probHat   0.5`.

We repeat the simulation for `nSim = 10000, 50000, 100000`:

```
# Define different simulation counts
simTiers <- c(5000, 10000, 20000)
# Function to perform simulations for a given number of simulations
simulateCDF <- function(currentSims) {
  probEst <- map_dbl(numPeopleRange, ~ {
    estBirthdayProb(numPeople = .x, nSim = currentSims)
  })
  tibble(
    numPeople = numPeopleRange,
    probHat = probEst,
    standardError = sqrt(probEst * (1 - probEst) / currentSims),
    nSim = currentSims
  )
}
# Perform simulations across different simulation counts
simulateCDFResults <- map_dfr(simTiers, simulateCDF)
```

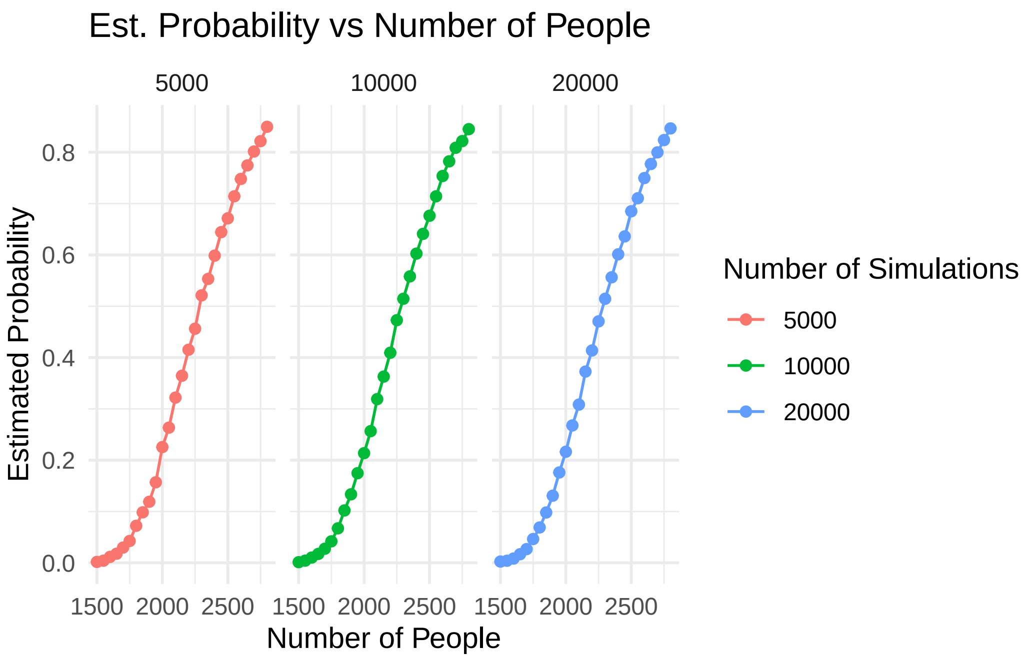
We plot the results for $B = \{5000, 10000, 20000\}$

```
# Combined Plot
ggplot(simulateCDFResults, aes(x = numPeople,
                               y = probHat, color = factor(nSim))) +
  geom_line() +
  geom_point() +
  labs(
    title = "Est. Probability vs Number of People",
    x = "Number of People",
    y = "Estimated Probability",
    color = "Number of Simulations"
  ) +
  scale_color_discrete(labels = c("5000", "10000", "20000")) +
  theme_minimal()
```




```
# Facet Plot
ggplot(simulateCDFResults, aes(x = numPeople,
                               y = probHat, color = factor(nSim))) +
  geom_line() +
  geom_point() +
  labs(
    title = "Est. Probability vs Number of People",
    x = "Number of People",
```

```
    y = "Estimated Probability",
    color = "Number of Simulations"
  ) +
  scale_color_discrete(labels = c("5000", "10000", "20000")) +
  theme_minimal() +
  facet_wrap(~ nSim)
```



Conclusion

Through simulation, we estimate that approximately **2286 people** are needed for there to be a **50% chance** that every possible birthday is represented. Increasing the number of simulations improves the accuracy of our estimates but does not significantly change the estimated number of people. Intriguingly, if we wanted there to be a **80% chance** that every possible birthday is represented, we estimate that it would require approximately **2692 people**! That is just an increase of **406 people** (17.76%) to get an extra **30%** chance that every possible birthday is represented.