ENGG 107 Bayesian Problem Set 2

Eason Cai

2025-01-24

1. Introduction

In this report, we perform two distinct Monte Carlo simulations:

1. **Normal Distribution Simulation**
   - We sample from a known univariate normal distribution ($\mu = 0$, $\sigma = 1$).
   - We estimate the mean and the 95th percentile ($p_{0.95}$) from these samples.
   - We also calculate uncertainties in these estimates.
2. **Estimation of π**
   - We use a classic geometric Monte Carlo approach.
   - We estimate π and quantify our uncertainty based on the sample size.

We discuss the choice of sample size, methods for convergence checks, and reproducibility. All code is provided in the Appendix and also in a separate ASCII file.

---

2. Method and Assumptions

2.1 Normal Distribution Simulation

- **Distribution**: $\mathcal{N}(0, 1)$
- **Estimates**:
  - Sample mean $\hat{\mu}$
  - Sample 95th percentile $\hat{q}_{0.95}$
- **Uncertainty**:
  - Measure the standard error of the mean given by $\sigma/\sqrt{N}$.
  - Uncertainty in the percentile estimate via repeated sampling.

2.1.1 Monte Carlo Simulation

- We start with a moderately large N which is 10^3 and increase by an increment of 10^4.
- We track how $\hat{\mu}$ and $\hat{q}_{0.95}$ change with increasing N for a total of 10 different seeds.
- We define "converged" when incremental increases in N do not significantly change the results or their uncertainties.

2.2 π Estimation

- **Geometric Interpretation**: Points uniformly sampled in the unit square $[0,1] \times [0,1]$.
- **Constraint**: A point (x, y) is inside the quarter circle of radius 1 if $x^2 + y^2 \leq 1$.
- **Formula**: $\pi \approx 4 \times \frac{\text{number of points in circle}}{\text{total points}}$.
- **Uncertainty**:
  - Approximate standard error given by $\sqrt{\frac{\hat{p}(1-\hat{p})}{N}}$ (where $\hat{p}$ is the estimated fraction of points within the circle), scaled appropriately by 4.

2.2.1 Monte Carlo Simulation

- Similar approach: for each seed, gradually increase the number of samples N.
- Observe the estimate's behavior and see at which N the estimate and its error stabilize.

---

3. Monte Carlo Simulations

3.1 Normal Distribution Simulation

3.1.1 Convergence check for the mean estimate

```r
library(ggplot2)  # For plot
library(dplyr)    # For generate a dataframe
library(viridis)  # For coloring the plot

N <- 1e5
sample_sizes <- seq(1000, N, by = 10000)
seeds <- c(123, 456, 789, 111, 222, 333, 555, 777, 888, 999)

# Generate data
df <- expand.grid(sample_size = sample_sizes, seed = seeds) %>%
  rowwise() %>%
  mutate(mean_estimate = {
    set.seed(seed)
    mean(rnorm(sample_size, 0, 1))}) %>%
  ungroup()

# Convert seed to factor for discrete color mapping
df$seed <- as.factor(df$seed)

ggplot(df, aes(x = sample_size, y = mean_estimate, color = seed, group = seed)) +
  geom_line() +
  geom_point() +
  scale_color_viridis_d(option = "cividis") +
  geom_hline(yintercept = 0, linetype = "dashed", color = "black") +
  labs(title = "Convergence of the Mean Estimate for Different Seeds",
       x = "Number of Runs",
       y = "Estimated Mean") +
  theme_minimal()
```

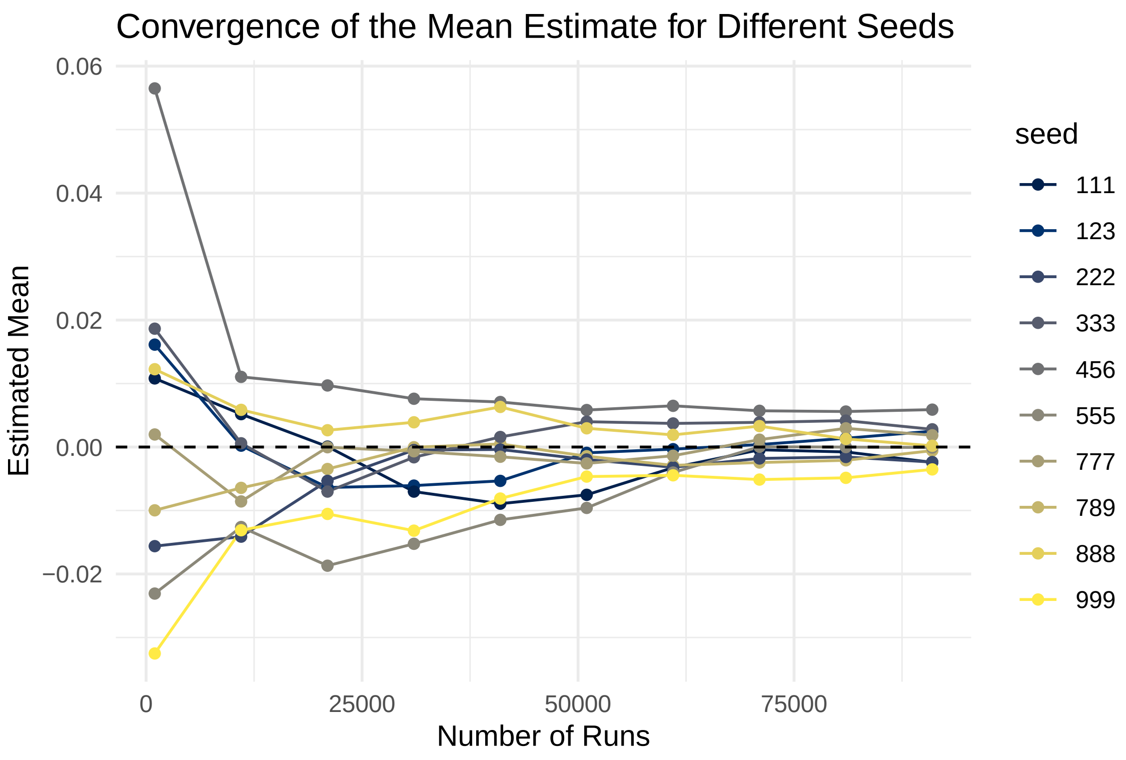

Figure 1: Convergence of mean estimate as sample size increases

```r
# Compute mean estimate and uncertainty
summary_mean <- df %>%
  group_by(sample_size) %>%
  summarise(
    mean = mean(mean_estimate),  # Average mean estimate for all seeds
    uncertainty = sd(mean_estimate) / sqrt(length(seeds))  # Standard Error
  )

kable(summary_mean, digits = 6)
```

sample_size	mean	uncertainty
1000	0.003512	0.008083
11000	-0.003193	0.002841
21000	-0.003902	0.002448
31000	-0.003280	0.002263
41000	-0.002028	0.001999
51000	-0.001590	0.001553
61000	-0.000738	0.001161
71000	0.000469	0.001019
81000	0.000606	0.000990
91000	0.000428	0.000910

We observe that the estimate quickly converges near the true mean of 0 as the uncertainty also decreases monotonically to 0.

3.1.2 Convergence check for the 95th percentile estimate

```r
df <- df %>%
  rowwise() %>%
  mutate(
    q95 = {
      set.seed(seed)
      quantile(rnorm(sample_size, 0, 1), 0.95)
    }
  ) %>%
  ungroup()

ggplot(df, aes(x = sample_size, y = q95, color = seed, group = seed)) +
  geom_line() +
  geom_point() +
  scale_color_viridis_d(option = "cividis") +
  geom_hline(yintercept = qnorm(.95, 0, 1), linetype = "dashed", color = "black") +
  labs(title = "Convergence of the 95th Percentile Estimate for Different Seeds",
       x = "Number of Runs",
       y = "Estimated 95th Percentile") +
  theme_minimal()
```

```r
# Compute 95 percentile estimate and uncertainty
summary_95 <- df %>%
  group_by(sample_size) %>%
  summarise(
    mean = mean(q95),
    uncertainty = sd(q95) / sqrt(length(seeds))
  )

kable(summary_95, digits = 6)
```

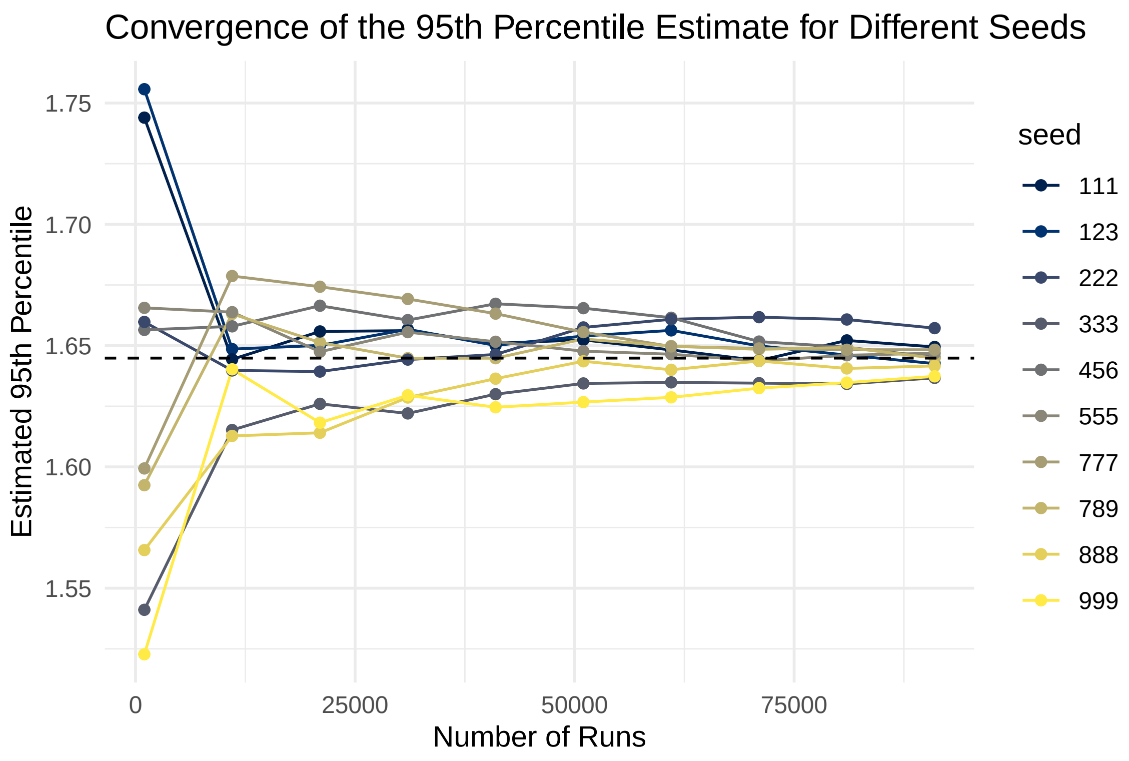

Figure 2: Convergence of 95th quantile estimate as sample size increases

sample_size	mean	uncertainty
1000	1.630283	0.025283
11000	1.646434	0.006625
21000	1.644294	0.006296
31000	1.646737	0.004952
41000	1.646511	0.004251
51000	1.648987	0.003615
61000	1.647629	0.003388
71000	1.645874	0.002664
81000	1.646134	0.002533
91000	1.645009	0.001909

We observe that the estimate quickly converges near the true mean of 1.645 as the uncertainty also decreases monotonically to 0.

3.2 Estimation of π

3.2.1 Convergence check for the pi estimate

```
df <- df %>%
  rowwise() %>%
  mutate(
    pi_estimate  = {
      set.seed(seed)
      x <- runif(sample_size, 0, 1)
      y <- runif(sample_size, 0, 1)
      inside_pts <- (x^2 + y^2) <= 1
      4 * sum(inside_pts) / sample_size
    }
  ) %>%
  ungroup()

ggplot(df, aes(x = sample_size, y = pi_estimate, color = seed, group = seed)) +
  geom_line() +
  geom_point() +
  scale_color_viridis_d(option = "cividis") +
  geom_hline(yintercept = pi, linetype = "dashed", color = "black") +
  labs(title = "Convergence of Pi Estimate for Different Seeds",
       x = "Number of Runs",
       y = "Estimated Pi Value") +
  theme_minimal()
```

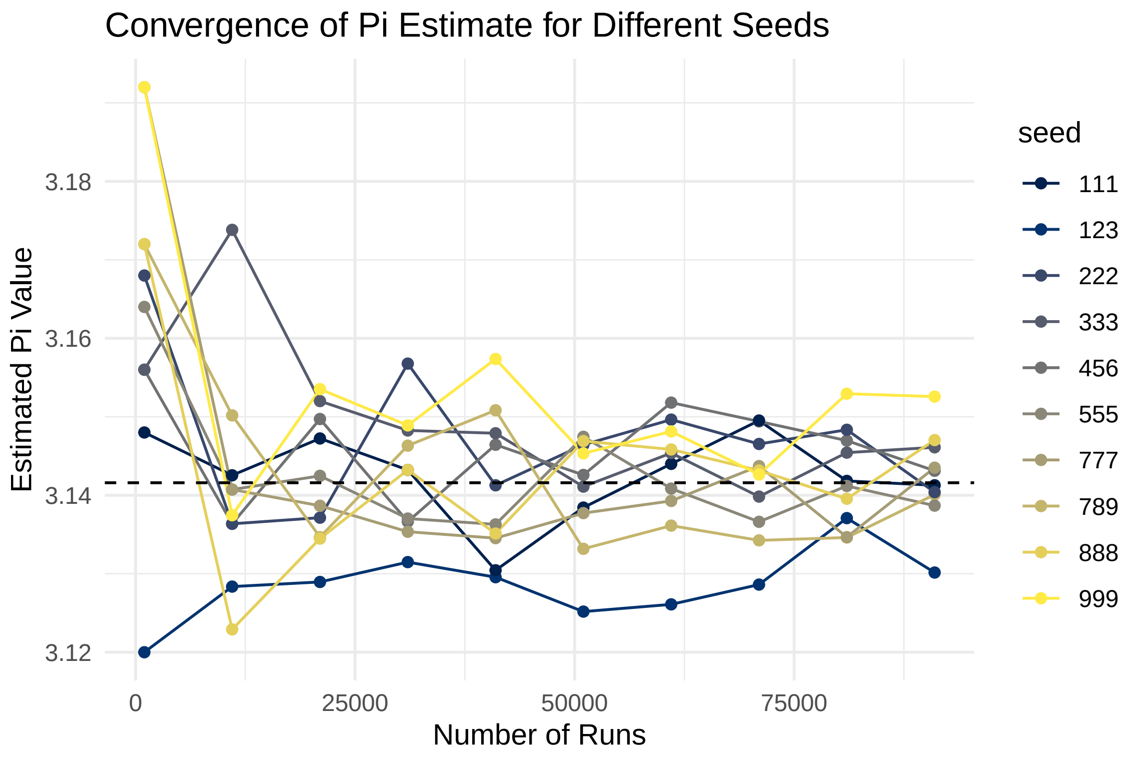

Figure 3: Convergence of pi estimate as sample size increases

```
# Compute pi estimate and uncertainty
summary_pi <- df %>%
  group_by(sample_size) %>%
  summarise(
    mean = mean(pi_estimate),
    uncertainty = sd(pi_estimate) / sqrt(length(seeds))
  )
```

```r
kable(summary_pi, digits = 6)
```

sample_size	mean	uncertainty
1000	3.164000	0.006693
11000	3.140945	0.004354
21000	3.141886	0.002657
31000	3.142723	0.002426
41000	3.140976	0.002949
51000	3.140431	0.002243
61000	3.142715	0.002388
71000	3.141431	0.002124
81000	3.142262	0.001929
91000	3.142295	0.001870

We see that the estimate oscillates around the true value $\pi \approx 3.14159$ as the uncertainty decreases monotonically to 0.

---

4. Summary

1. **Choice of Sample Sizes**
   - For all simulations, we started at 1000 and went to 91000.
   - This provided sufficiently small standard errors for illustrative purposes.
   - Larger sample sizes continue to refine the estimates but also increase computation time.
2. **Convergence**
   - We used incremental sampling to visualize whether the mean, 95th percentile (for the normal distribution) and the π estimate stabilize around a specific value.
   - To simulate real world scenario, we used 10 different seed to observe the convergence.
3. **Reproducibility**
   - All code and random seeds are included in this document.
   - By setting `seed`, anyone can reproduce the same numeric results.
   - This R Markdown file can be knitted to produce the same figures and estimates.
4. **Limitations/Assumptions**
   - We assumed an ideal scenario with no measurement errors other than sampling uncertainty.
   - More sophisticated techniques (e.g., variance reduction) could improve convergence rates.
5. **Citations**
   - GeeksforGeeks. (2024, January 23). *Standard error.* Retrieved from https://www.geeksforgeeks.org/standard-error/

- The Modern Scientist. (2021, August 19). *Estimating π using Monte Carlo methods.* Medium. Retrieved from https://medium.com/the-modern-scientist/estimating-pi-using-monte-carlo-methods-dbdf26c888d6

6. **License**

7. **Copyright**

Appendix: Code

You can also access the repository using the following URL: https://github.com/easoncai999/bayesian-ps2

```r
library(ggplot2)  # For plot
library(dplyr)    # For generate a dataframe
library(viridis)  # For coloring the plot

N <- 1e5
sample_sizes <- seq(1000, N, by = 10000)
seeds <- c(123, 456, 789, 111, 222, 333, 555, 777, 888, 999)

### PART 1 ###

# Mean Estimate
df <- expand.grid(sample_size = sample_sizes, seed = seeds) %>%
  rowwise() %>%
  mutate(mean_estimate = {
    set.seed(seed)
    mean(rnorm(sample_size, 0, 1))}) %>%
  ungroup()

df$seed <- as.factor(df$seed)

ggplot(df, aes(x = sample_size, y = mean_estimate, color = seed, group = seed)) +
  geom_line() +
  geom_point() +
  scale_color_viridis_d(option = "cividis") +
  geom_hline(yintercept = 0, linetype = "dashed", color = "black") +
  labs(title = "Convergence of the Mean Estimate for Different Seeds",
       x = "Number of Runs",
       y = "Estimated Mean") +
  theme_minimal()

summary_mean <- df %>%
  group_by(sample_size) %>%
  summarise(
    mean = mean(mean_estimate),  # Average mean estimate across seeds
    uncertainty = sd(mean_estimate) / sqrt(length(seeds))  # Standard Error (SE) across seeds
  )

# 95th Percentile Estimate
df <- df %>%
  rowwise() %>%
  mutate(
    q95 = {
      set.seed(seed)
      quantile(rnorm(sample_size, 0, 1), 0.95)
    }
  ) %>%
  ungroup()

ggplot(df, aes(x = sample_size, y = q95, color = seed, group = seed)) +
  geom_line() +
  geom_point() +
  scale_color_viridis_d(option = "cividis") +
  geom_hline(yintercept = qnorm(.95, 0, 1), linetype = "dashed", color = "black") +
  labs(title = "Convergence of the 95th Percentile Estimate for Different Seeds",
       x = "Number of Runs",
       y = "Estimated 95th Percentile") +
  theme_minimal()

summary_95 <- df %>%
  group_by(sample_size) %>%
  summarise(
    mean = mean(q95),
    uncertainty = sd(q95) / sqrt(length(seeds))
  )

### PART 2 ###

# Pi Estimate
df <- df %>%
  rowwise() %>%
  mutate(
    pi_estimate  = {
      set.seed(seed)
      x <- runif(sample_size, 0, 1)
      y <- runif(sample_size, 0, 1)
      inside_pts <- (x^2 + y^2) <= 1
      4 * sum(inside_pts) / sample_size
    }
  ) %>%
  ungroup()

ggplot(df, aes(x = sample_size, y = pi_estimate, color = seed, group = seed)) +
  geom_line() +
  geom_point() +
  scale_color_viridis_d(option = "cividis") +
  geom_hline(yintercept = pi, linetype = "dashed", color = "black") +
  labs(title = "Convergence of Pi Estimate for Different Seeds",
       x = "Number of Runs",
       y = "Estimated Pi Value") +
  theme_minimal()

summary_pi <- df %>%
  group_by(sample_size) %>%
  summarise(
    mean = mean(pi_estimate),
    uncertainty = sd(pi_estimate) / sqrt(length(seeds))
  )
```