EARS 202

Problem Set 2: Monte Carlo and Convergence

Due: Friday, January 24

Hugh Shields

1 Determining the mean of a normal distribution

Here, I seek to determine the mean and the 95 percentile from a known univariate normal distribution with a mean of 0 and a standard deviation of 1. I then need to quantify the uncertainty.

To do so, I need to address the key sources of uncertainty in this problem: the number of samples drawn and the seed used to generate the samples. To quantify the uncertainty of the former, I start by showing the results of varying the number of samples with a single seed. In this case, I set a seed and then draw n samples from a normal distribution with mean $\mu = 0$ and standard deviation $\sigma = 1$. For a seed of 0 and n ranging between 10^2 and 10^7, I show the results in Table 1.

Table 1: Simulation Results for Varying Sample Sizes

Sample Size (n)	**Mean (μ)**	**5th Percentile**	**95th Percentile**
100	0.02267	-1.29162	1.30084
1,000	-0.03363	-1.65860	1.63436
10,000	0.01115	-1.64345	1.65126
100,000	0.00037	-1.65181	1.63814
1,000,000	-0.00017	-1.64542	1.64444
10,000,000	-0.00055	-1.64580	1.64456

Note that as the number of samples increases, the mean and percentiles converge upon their true values (in this case $\mu = 0$, 5th ≈ -1.64485 and 95th ≈ 1.64485), which I know from the normal distribution.

Next, I address uncertainty due to seeding. If I only considered a single seed, I would be making the assumption that a seeded random number generator is producing truly random samples, but the samples are only pseudo-random. To address this inconsistency, I perform the same experiment as above, but I repeat it with 500 different seeds. This choice assumes that the sequences of pseudorandom numbers produced by the seeds are independent and identically distributed. I chose 500 as the convergence at 10^7 runs looks tight, and the run time is only a few minutes. I then report the mean of the 500 seeds,

$$\mu_n = \frac{\sum_{i=1}^{500} \mu_i^n}{500},$$

where μ_i^n is the mean of the ith simulation with n samples. I also report 5th to 95th percentile of the distribution of means μ_i^n from the 500 different seeds at each n. These percentiles quantify the total uncertainty of the estimate. The results are shown in Table 2 and Figure 1.

Table 2: Simulation Results for 500 Seeds and Varying Sample Sizes

Sample Size (n)	**Mean (μ_n)**	**5th Percentile**	**95th Percentile**
100	2.35549e-03	-1.61451e-01	1.70987e-01
1,000	7.17425e-04	-5.20880e-02	5.38192e-02
10,000	-6.32562e-06	-1.62360e-02	1.63076e-02
100,000	2.78997e-05	-5.21463e-03	4.84103e-03
1,000,000	-5.85099e-05	-1.66444e-03	1.55619e-03
10,000,000	-2.28184e-05	-5.66710e-04	4.93366e-04

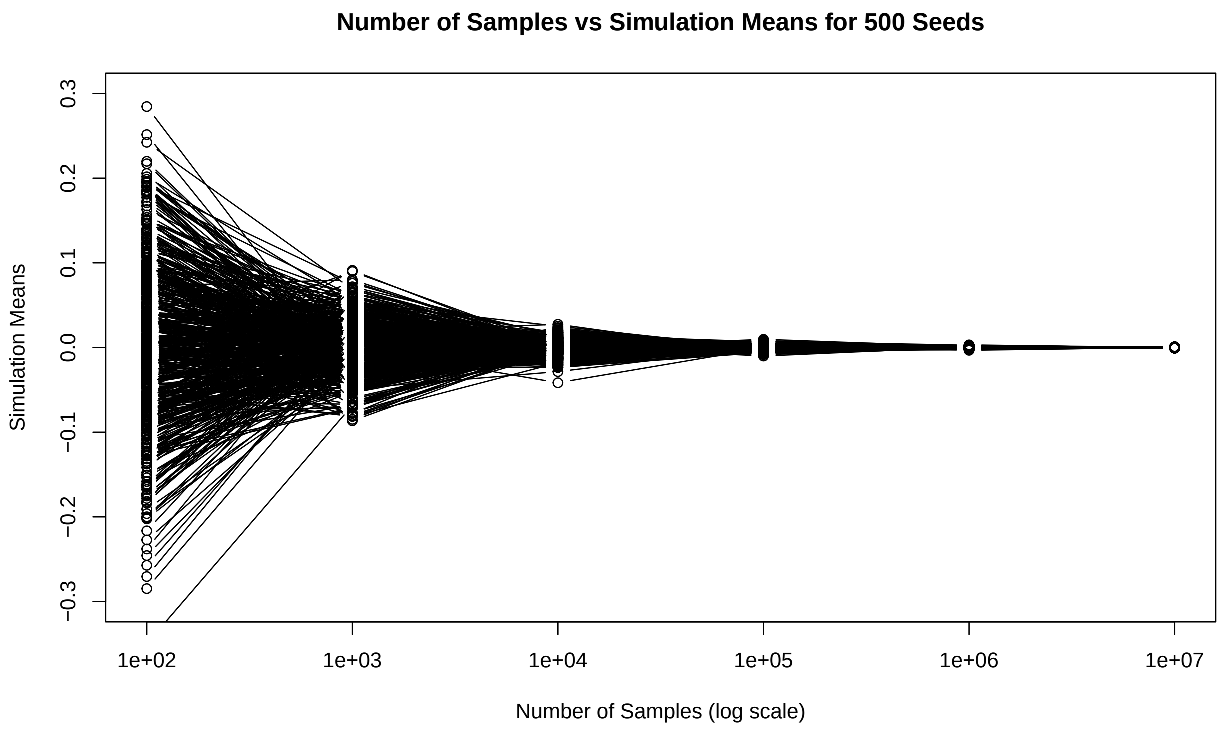

Figure 1: The means (μ) of simulations of varying sample size (n) for 500 seeds. Each line represents a unique seed.

I note that the percentiles shown in Table 2 and the spread shown in Figure 1 indicate convergence as the number of samples increases. In other words, as the number of samples increases, the effect of choosing any given seed to run a single simulation with decreases. Additionally, the uncertainty in the final estimate converges as sample size increases. In the case of 10^7, the 5th to 95th percentile range for 500 seeds is about $\pm 0.05\%$ of the true value.

Regarding modeling choices, the samples were chosen using R's `rnorm()` function. Convergence in the case of Figure 1 was determined visually. Alternatively, I could have chosen a convergence criteria; in this case, I would have selected the number of seeds over which to rerun the trials based on when the largest n values confidence interval fell below a certain threshold (e.g. within 1% of the expected mean). Additionally, I chose to sample with n values ranging between $n = 10^2$ and $n = 10^7$. The minimum number of points was based on the point where the mean was within 5% of the true value and the maximum was chosen due to run time constraints. The final uncertainty in Figure 1 and Table 2 was estimated using 5th to 95th percentiles, as this seemed like a reasonable quantification of the variation between seeds.

The analysis here is reproducible, as running the code in RStudio will produce Tables 1-2 and Figure 1.

2 Determining the value of π

Here, I seek to determine the value of π via Monte Carlo simulation and quantify the uncertainty.

The basic approach is as follows. The area of a circle is known to be given by

$$A = \pi r^2.$$

Thus, I can derive the value of π given the area and radius of any circle. To determine the area of a circle, I use Monte Carlo simulation. I first create a square with side length 1 centered at the point (0,0). To sample points from this square uniformly, I generate x_i and y_i values on a uniform distribution from -0.5 to 0.5. Then, I can determine which (x_i, y_i) fall within a circle inscribed within the square with the following inequality

$$\sqrt{x_i^2 + y_i^2} \leq r,$$

where in this case $r = 0.5$. Then, given that the area of the square is 1, the area of the inscribed circle is just the ratio of points falling within the circle to the values falling outside of it.

To quantify uncertainty, I use a similar method to Section 1. I vary the number of sampled points, n, and calculate the mean of 100 seeds μ_n, which represents my best estimate of π. The number of seeds was chosen for the same reasoning as in Section 1. The results are shown in Table 3 and Figure 2. Again, the reported 5th and 95th percentiles in Table 3 are of the percentiles of the distribution of individual calculated π value for the 100 different seeds at each sample size n.

In this problem, I make a number of assumptions. First, I assume that, as before, different seeds will produce independent sets of points. Additionally, I allow points that fall exactly on the circle to be within the circle. This is a function of numerical discretization, as in reality real numbers drawn from a uniform distribution would never fall exactly on the edge of the circle.

TABLE 3: Calculated Mean π Value (μ_n) for 100 Seeds and Varying Sample Sizes

Sample Size (n)	**Mean π Value (μ_n)**	**5th Percentile**	**95th Percentile**
100	3.13360	2.84000	3.44000
1,000	3.13832	3.03960	3.20860
10,000	3.14314	3.11960	3.16924
100,000	3.14069	3.13223	3.15101
1,000,000	3.14176	3.13885	3.14478
10,000,000	3.14160	3.14087	3.14232

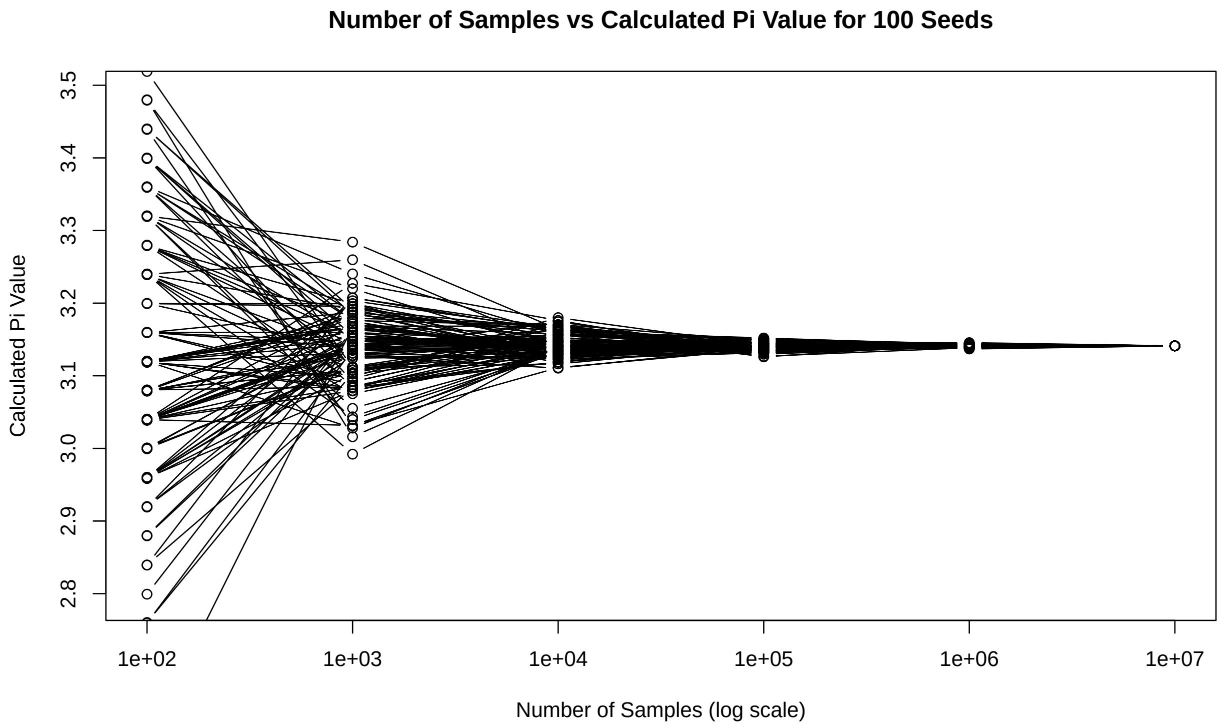

FIGURE 2: The calculated π values of simulations of varying sample size (n) for 100 seeds. Each line represents a unique seed.

As in Section 1, we see that the estimated value of π between seeds converges on the true value as the number of points drawn to make the estimate increases. In the case of 10^7 sample points, the 5th to 95th percentile range for 100 seeds is about $\pm 0.07\%$ of the true value of π.

Regarding modeling choices, the x and y values were chosen using R's `runif()` function. Again, convergence in the case of Figure 2 was determined visually. I could have alternatively chosen a convergence threshold. The final uncertainty in Figure 2 and Table 3 was estimated using 5th to 95th percentiles, as this again seemed like a reasonable quantification of the variation between seeds. This criteria could change depending on the use case.

The analysis here is reproducible, as running the code in RStudio will produce Table 3 and Figure 2.

A Normal Sampling Code

See `normal_sampling.R` below. This file can be found at https://github.com/hughshields/ENG_107.

```
############################################
##  file: normal_sampling.R
##   ENG 107 Problem Set 2 Part 1 Solutions
##   - determine the mean and the 95 percentile from a known univariate normal
##     distribution with a mean of zero and a standard deviation of one with
##     your estimated uncertainties
######################################################
##  author: Hugh Shields
##  copyright by the author
##  distributed under the GNU general public license
##  https://www.gnu.org/licenses/gpl.html
##  no warranty (see license details at the link above)
##################################################
# version 1: last changes: Jan. 24. 2025 by Hugh Shields
# contact: nikolaus.h.shields.gr@dartmouth.edu
##################################################
# sources:
# - R help files accessed through R-studio for syntax
# - discussions during ENG 107 class time
# - An introduction to R (2010) by Longhow Lam
# - coin-example.R from ENG 107 Canvas
####################################################
# how to run:
# - save the file in a directory
# - go to the directory with this file
# - open R
# - type 'source(normal_sampling.R)'
# - read the printed results in the console and
#   open the new pdf file to analyze the results
############################################

# Clear any existing variables and plots.
rm(list = ls())
graphics.off()

# Varying Number of Samples ############################

# define a seed for reproducibility
set.seed(0)
```

```r
# create sample size vector
number_samples = exp(seq(log(1e2), log(1e7), length.out = 6))

# do the simulation for each number of samples and print results
print('Running test for varying sample size')
for (i in seq_along(number_samples)) {
  n = number_samples[i]
  sim=rnorm(n, mean = 0, sd = 1)

  # calculate the mean and (5,95) percentiles of each simulation and print
  sim_mean = mean(sim)
  sim_percentiles = quantile(sim, probs = c(0.05, 0.95))
  cat(sprintf("n = %.f, mean = %.5f, (5th,95th) percentiles = (%.5f, %.5f)\n",
              n, sim_mean, sim_percentiles[1], sim_percentiles[2]))
}

# Varying Seeds and Number of Samples ######################

# define a seed for reproducibility
seeds = 1:500

# define the number of samples
sim_means = matrix(NA, nrow = length(seeds), ncol = length(number_samples))

# plot outputs
pdf(file="seed_means.pdf",10,6.17)

# run the simulation for lots of seeds
print('Running test for 500 seeds and varying sample size')
for (i in seq_along(seeds)) {
  set.seed(seeds[i])

  # do the simulation for each number of samples for seed i
  #######################
  for (j in seq_along(number_samples)) {
    n = number_samples[j]
    sim=rnorm(n, mean = 0, sd = 1)
    sim_means[i,j]=mean(sim)
  }
  if (i==1) {
    plot(number_samples, sim_means[i,], log = "x", type = "b", ylim = c(-0.3, 0.3),
         xlab = "Number of Samples (log scale)", ylab = "Simulation Means",
         main = "Number of Samples vs Simulation Means for 500 Seeds")
  } else {
    par(new = TRUE)
    plot(number_samples, sim_means[i,], log = "x", type = "b", ylim = c(-0.3, 0.3),
         axes = FALSE, xlab = "", ylab = "")
  }
}
dev.off()

# print the results
for (i in seq_along(number_samples)) {
  # calculate the mean and (5,95) percentiles for each n over 500 seeds
  n = number_samples[i]
  n_mean = mean(sim_means[,i])
  n_percentiles = quantile(sim_means[,i], probs = c(0.05, 0.95))
  cat(sprintf("n = %.f, mean = %.5e, (5th,95th) percentiles = (%.5e, %.5e)\n",
              n, n_mean, n_percentiles[1], n_percentiles[2]))
}
```

B Pi Sampling Code

See pi_sampling.R below. This file can be found at https://github.com/hughshields/ENG_107.

```r
#############################################
##  file: pi_sampling.R
##   ENG 107 Problem Set 2 Part 2 Solutions
##   - determine the value of pi with your estimated uncertainties
#######################################################
##  author: Hugh Shields
##  copyright by the author
##  distributed under the GNU general public license
##  https://www.gnu.org/licenses/gpl.html
##  no warranty (see license details at the link above)
###################################################
# version 1: last changes: Jan. 24. 2025 by Hugh Shields
# contact: nikolaus.h.shields.gr@dartmouth.edu
###################################################
# sources:
# - R help files accessed through R-studio for syntax
# - discussions during ENG 107 class time
# - An introduction to R (2010) by Longhow Lam
# - coin-example.R from ENG 107 Canvas
#####################################################
```

```
# how to run:
# - save the file in a directory
# - go to the directory with this file
# - open R
# - type 'source(pi_sampling.R)'
# - read the printed results in the console and
#   open the new pdf file to analyze the results
############################################

# Clear any existing variables and plots.
rm(list = ls())
graphics.off()

# Varying Seeds and Number of Samples #####################

# define a seed for reproducibility
seeds = 1:100

# create sample size vector
number_samples = exp(seq(log(1e2), log(1e7), length.out = 6))

# create a matrix to hold the calculated pi values
pi_calc = matrix(NA, nrow = length(seeds), ncol = length(number_samples))

# plot outputs
pdf(file="pi_means.pdf",10,6.17)

# run the simulation for lots of seeds
print('Running test for 100 seeds and varying sample size')
for (i in seq_along(seeds)) {
  set.seed(seeds[i])

  # calculate pi for seed i for each number of samples
  #######################
  for (j in seq_along(number_samples)) {
    n = number_samples[j]
    x = runif(n, min = -0.5, max = 0.5)
    y = runif(n, min = -0.5, max = 0.5)
    r = sqrt(x^2+y^2)
    area_rat = length(r[r<=0.5])/length(r)
    pi = area_rat/(0.5^2)
    pi_calc[i,j]=pi
  }
  if (i==1) {
    plot(number_samples, pi_calc[i,], log = "x", type = "b",
         ylim = c(pi-0.35, pi+0.35),
         xlab = "Number of Samples (log scale)", ylab = "Calculated Pi Value",
         main = "Number of Samples vs Calculated Pi Value for 100 Seeds")
  } else {
    par(new = TRUE)
    plot(number_samples, pi_calc[i,], log = "x", type = "b",
         ylim = c(pi-0.35, pi+0.35),
         axes = FALSE, xlab = "", ylab = "")
  }
}
dev.off()

# print the results
for (i in seq_along(number_samples)) {
  # calculate the mean and (5,95) percentiles for each n over 100 seeds
  n = number_samples[i]
  n_mean = mean(pi_calc[,i])
  n_percentiles = quantile(pi_calc[,i], probs = c(0.05, 0.95))
  cat(sprintf("n = %.f, mean pi value = %.5f,
              (5th,95th) percentiles = (%.5f, %.5f)\n",
              n, n_mean, n_percentiles[1], n_percentiles[2]))
}
```