STT 465 HW 3

Derien Weatherspoon

2023-02-13

```
library(ggplot2)
```

Question 1:

Define a probability density function as p(x) = cx3 on the interval $0 < X < 3$. a) Determine what c is.

a: Setting the integral up to be c * integrand from 0 to 3 * x^3 dx, solving for c warrants 4/81

b) Use the inverse cdf method to simulate 10,000 outcomes of the distribution. Estimate the mean and standard deviation of the distribution.

```
#Build an integral here with the function cx^3 with bounds from 0 to 3

integral_1 <- function(x) {
  c <- 4/81
  result <- c*x^3
  return(result)
}
n <- 10000
u <- runif(n,0,3)
integrate(integral_1, lower = 0, upper = 3)
```

```
## 1 with absolute error < 1.1e-14
```

```
df_q1 <- integral_1(u)
mean(u)
```

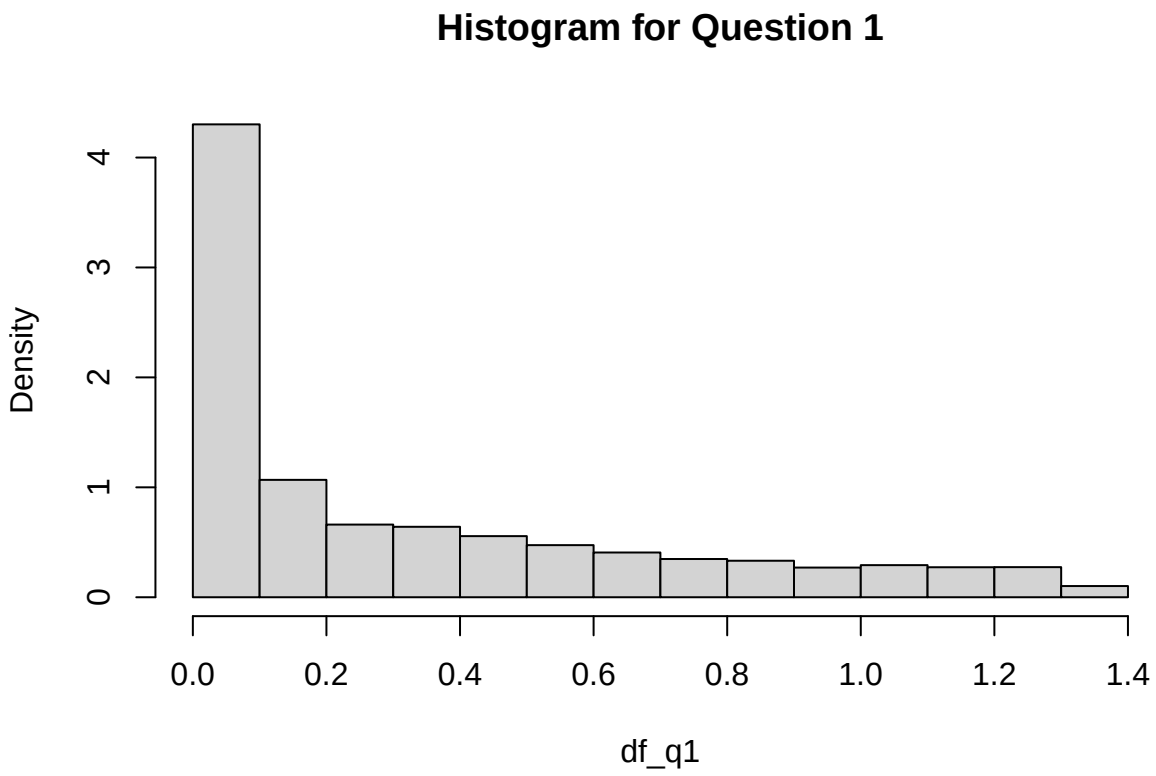
```
## [1] 1.488977
```

```
sd(u)
```

```
## [1] 0.8724134
```

c) Plot a histogram of your simulation results. Does it match what you would expect?

```
#Plotting a histogram
hist(df_q1, probability = T, main = "Histogram for Question 1")
```

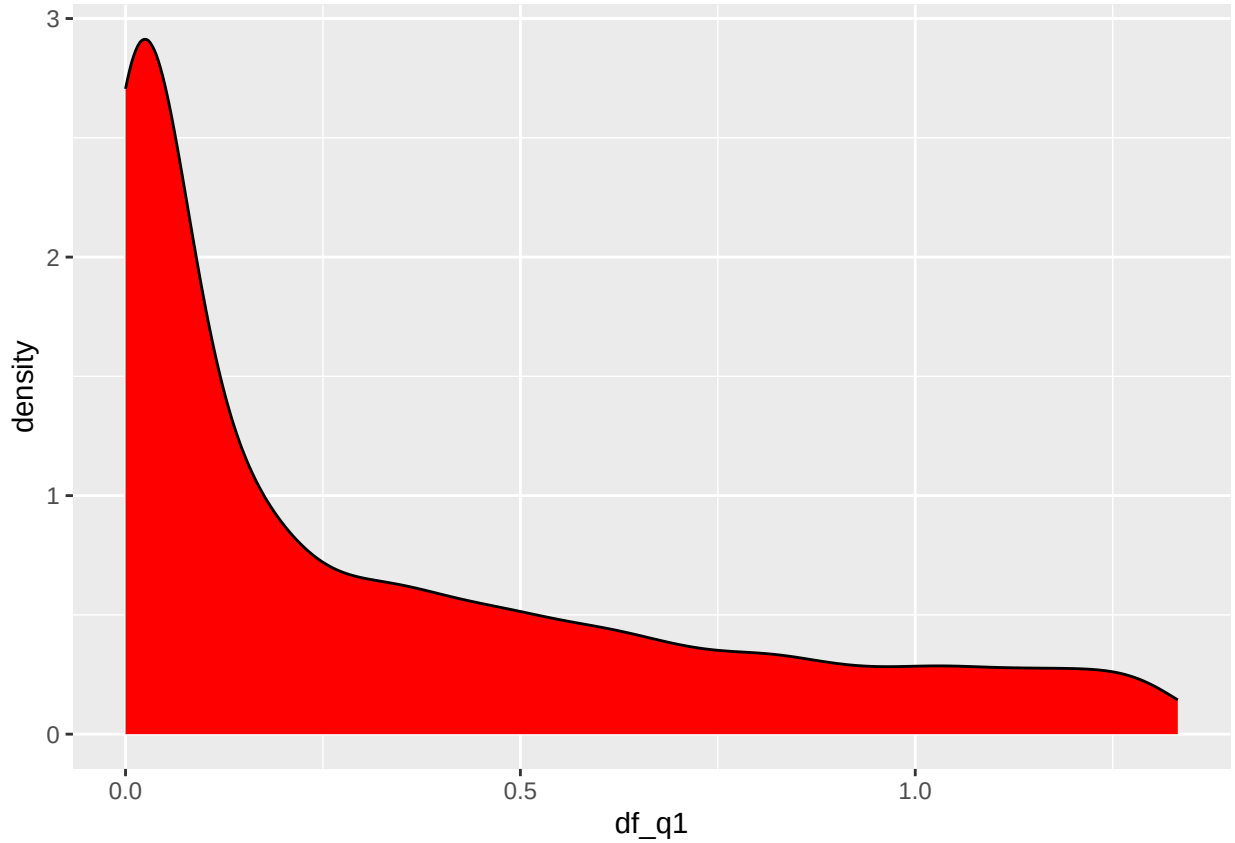



```
#plot(density(df_q1),col="blue",lwd=2,main="PDF for n=10000") <- another way to do it
```

Both plots follow the same density pattern, so it adds up. It is plotting the distribution.

d) Re-do (c) with geom_density instead of geom_histogram. What is it doing?

```
ggplot() + geom_density(aes(df_q1), fill = "red", color = "black")
```



Fits a smooth curve of the density of the distribution

e) Numerically approximate the expected value, variance, and standard deviation of the distribution. Calculate the expected value from the definition with an integral and compare how close the 2 are.

```
#expected value
calc_expected_value <- function(distribution) {
  # Calculate expected value
  mean(distribution)
}
calc_expected_value(df_q1)
```

```
## [1] 0.3314724
```

```
#variance
var(df_q1)
```

```
## [1] 0.1439885
```

```
#standard deviation
sd(df_q1)
```

```
## [1] 0.3794582
```

f) Use a simulation with acceptance/rejection to approximate the expectation of X, given that x is greater than 2.

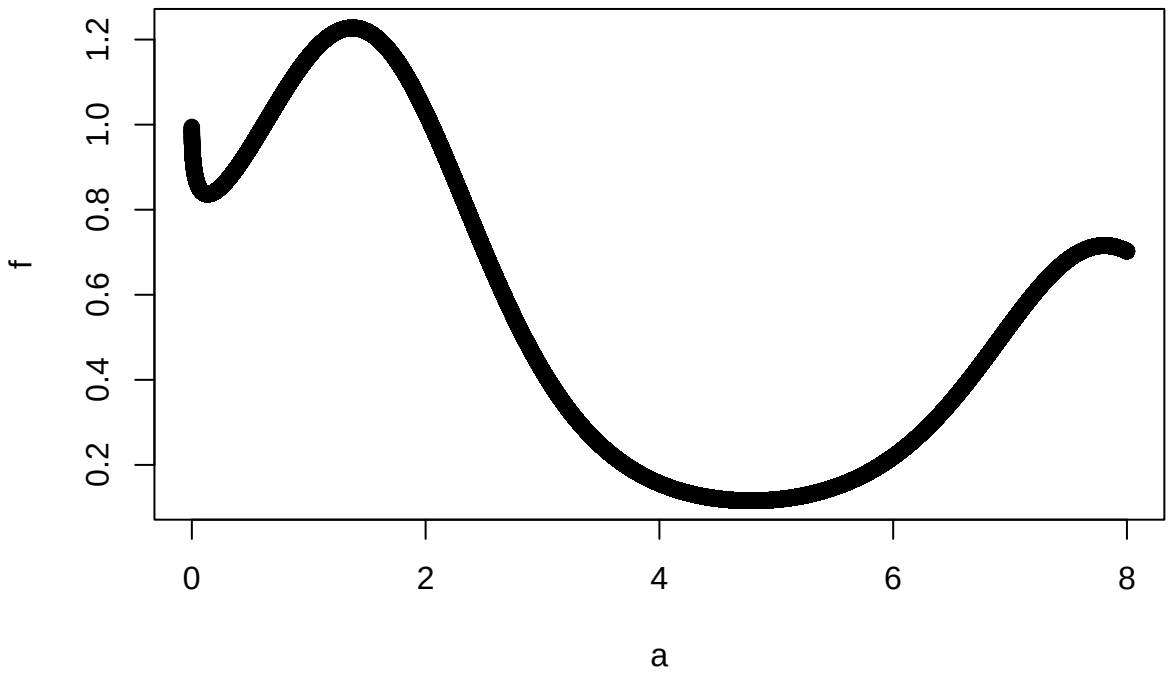
```
p <- function(x) 0.25 * x^3
g <- function(x) 1
M <- p(max(seq(2, 3, by=0.01))) / g(max(seq(2, 3, by=0.01)))
M
```

```
## [1] 6.75
```

Question 2:

For the following integral: (on PDF) a) Generate a plot of the region.

```
a <- b <- runif(100000,0,8)
f <- exp(sin(b))/(1+sqrt(b))
plot(a,f)
```



b) Estimate using standard Monte Carlo integration (n = 100,000)

```
mean(f)*8
```

```
## [1] 4.393923
```

c) Estimate using acceptance/rejection Monte Carlo (again, n = 100,000)

```
fx <- runif(100000,0,4)
p2 <- sum(fx < f)/100000
integral_2 <- p2*8*4
integral_2
```

```
## [1] 4.4224
```

Question 3:

For the following integral (on PDF):

a) Compute the following integral both with the standard Monte Carlo approach (with appropriately truncated bounds) and with Importance Sampling, using an appropriate normal distribution as an importance function. Use n = 10,000 both times.

```
# Define the function to be integrated
num <- 10000
integral_3 <- function(x) {
  result_2 <- exp(-(x-6)^4)
}
#integrate
u2 <- runif(num,4,8)
df_q3 <- mean(integral_3(u2))*(8-4)
df_q3
```

```
## [1] 1.78103
```

b) Re-do both methods 100 times. Compute the variance of the answers from each method and compare.

```
# Monte Carlo method. Use a for loop as mentioned in class.
simulated_integral <- c() #store vector
for (i in 1:100) {#100 times
  t <- runif(10000,4,8) #same num
  simulated_integral[i] <- mean(integral_3(t))*(8-4)
}
mc_var <- var(simulated_integral)

#method for importance sampling
simulated_integral_2 <- c()
for (i in 1:100){
  r <- abs(rnorm(10000, mean = 6, sd = 1))
  simulated_integral_2[i] <- mean(integral_3(r)/rnorm(r,6,1))
}
is_var <- var(simulated_integral_2)
mc_var
```

```
## [1] 0.0002038991
```

```r
is_var #seems like the answer should not be that small
```

```
## [1] 4.975974e-07
```

Both are very small in value, though I think the importance sampling variance is low due to error in code.

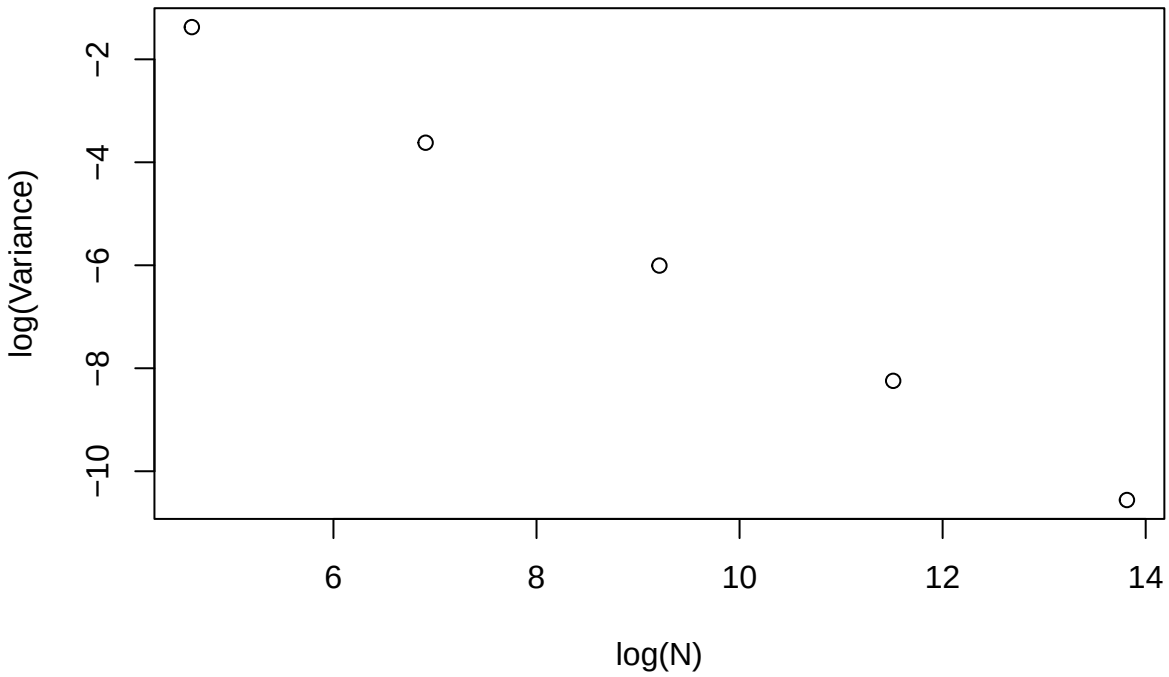
Question 4:

Suppose we have an exponential probability distribution with parameter lambda = 1/5, so that its expected value is 5. We will perform some simulation experiments to determine the behavior of the sample mean. a) Generate 2,000 simulations which each constitute 100 outcomes of this distribution. Calculate the mean values for the 2,000 experiments. b) Next, calculate the mean value and variance for the 2,000 sample means. c) Repeat (a) and (b) for N = 1,000, 10,000, 100,000, and 1,000,000 outcomes (2,000 simulations of each). Note: 1,000,000 could take several minutes. d) Plot the log(Variance) vs. log(N). Do you get something which is close to a straight line? What is the slope? Does the relationship between N and the variance close to what you would expect with the Central Limit Theorem?

```r
set.seed(304)
lambda <- 1/5 # parameter


# function for generating sample means
sample_mean <- function(n) {
  x <- rexp(n, lambda)
  mean(x)
}

# N values to simulate
N_values <- c(100, 1000, 10000, 100000, 1000000)

# store the results
results <- matrix(NA, nrow = length(N_values), ncol = 3)
colnames(results) <- c("N", "Mean of Sample Means", "Variance of Sample Means")

for (i in seq_along(N_values)) {
  N <- N_values[i]
  samp_means <- replicate(2000, sample_mean(N))
  mean_samp_means <- mean(samp_means)
  variance_samp_means <- var(samp_means)
  results[i, ] <- c(N, mean_samp_means, variance_samp_means)
}
print(results)
```

```
##          N Mean of Sample Means Variance of Sample Means
## [1,] 1e+02             5.023053             2.528876e-01
## [2,] 1e+03             5.001772             2.679669e-02
## [3,] 1e+04             5.000788             2.464378e-03
## [4,] 1e+05             5.000479             2.627683e-04
## [5,] 1e+06             5.000084             2.596559e-05
```

```r
plot(log(results[, 1]), log(results[, 3]), xlab = "log(N)", ylab = "log(Variance)")
```



It appears to a negative slope in a straight line. This is expected.

Sources

https://bookdown.org/rdpeng/advstatcomp/rejection-sampling.html https://harrison4192.github.io/Simulacra/simulations.html#monte-carlo-integration