Homework 04 - STAT440

Joseph Sepich (jps6444)

09/19/2020

```
set.seed(42)
```

Problem 1

Use R to simulate samples from a normal distribution. Let Y be a random variable with chi-squared distribution with 5 degrees of freedom.

Part a

Since we can only sample from the standard normal distribution, I would use the transformation that the Chi-square distribution is really the sum of squares of standard normal variables. The degrees of freedom is the number of standard normal random variables that make up the Chi-square distribution. In this case we would use the following transformation:

$$\chi^2 = \Sigma_{i=1}^{5} Z_i^2$$

Part b

Variance of Y:

$$Var(Y) = (\int x f(x) dx)^2 - (\int x^2 f(x) dx)$$

Foruth moment of Y:

$$\int x^4 f(x) dx$$

We can approximate all these integrals using Monte Carlo, where $h(x) = x^n$ (depending on the moment) and $f(x)$ is the density of Y that we are sampling from. Therefore we can use our samples X_i from the standard normal and transform them (5 for 5 degrees of freedom) into a chi-square sample Y_i. These samples can give us the value $\frac{1}{n}\Sigma_{i=1}^{n} h(Y_i) = \frac{1}{n}\Sigma_{i=1}^{n} Y_i^k$, which can approximate the k^{th} moment integral.

Part c

Use R to estimate the above quantities using Monte Carlo with N = 10,000 samples and report the results.

```r
n <- 10000
df <- 5
norm_samples <- matrix(rnorm(n*df), nrow=n, ncol=df)
norm_samples <- apply(norm_samples, c(1,2), function(x) {x^2})
chi_samples <- rowSums(norm_samples)

# variance
second <- sum(chi_samples^2) / n
expect_square <- (sum(chi_samples) / n) ^ 2
variance <- second - expect_square
print(variance)
```

```
## [1] 9.983276
```

```r
# compare approx with sd function
print(sd(chi_samples)^2)
```

```
## [1] 9.984275
```

```r
# fourth moment
fourth <- sum(chi_samples^4) / n
print(fourth)
```

```
## [1] 3449.642
```

Problem 2

Part a

Express the probability as an expectation and as an integral. For this we can use an indication function $g(x)$, which is 1 when $|\overline{X}_n - \mu| < \epsilon$ and 0 when it is not . ($|\overline{X}_n - \mu| \geq \epsilon$) We can use this indication function to express the probaility as an expectation:

$$E[g(x)] = P(|\overline{X}_n - \mu| < \epsilon)$$

We represent this as an integral (formula for finding the first moment) where $|\overline{X}_n - \mu|$ is the value we plug in for x and $f(x)$ is the pdf of $|\overline{X}_n - \mu|$.

$$E[g(x)] = \int g(x)dP(x) = \int g(x)f(x)dx$$

Logically this makes sense as $P(|\overline{X}_n - \mu| < \epsilon)$ looks like the representation of a CDF, which is the integral we got.

Part b

Here $I_n = \int h(x)f(x)dx$ where $h(x)$ is the indicator function that we described above. This integral is in the form of an expectation of the indicator function, therefore all we need to do is sample from this indicator function random variable and find that sample mean, which is the Monte Carlo integration process.

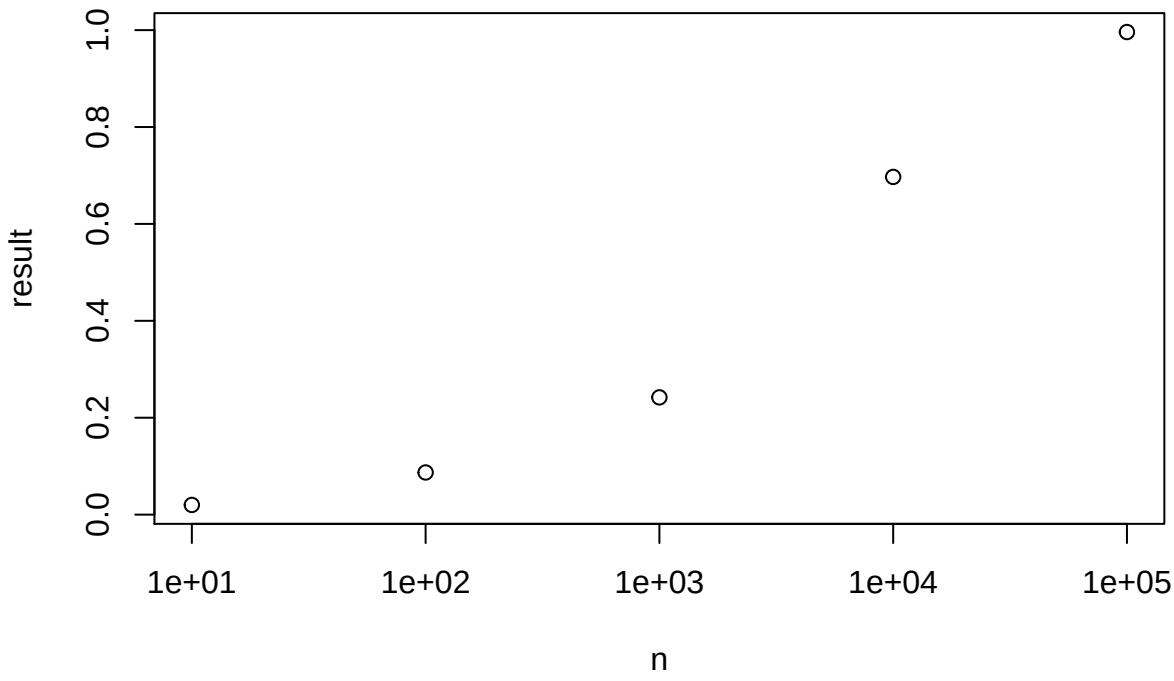
Part c

```
epsilon <- 0.01
m <- 1000
n <- c(10, 100, 1000, 10000, 100000)
result <- vector("numeric", length(n))

mu <- 2
sigma <- 1

for (i in 1:length(n)) {
  samples <- matrix(rnorm(n[i] * m, mu, sigma), nrow = m, ncol = n[i])
  samples <- rowSums(samples)
  x_n <- samples / n[i]
  comp_val <- abs(x_n - mu)
  result[i] <- sum(as.integer(comp_val < epsilon)) / m
}

plot(n, result, log='x')
```



Part d

As you can see in the plot above the correct behavior is shown. As the sample size of the sample mean gets larger, the expected value of our indicator function gets closer to 1, which means the numerical approximation

of the $P(|\overline{X}_n - \mu| < \epsilon)$ gets closer to 1. This is exactly what the limit states in the weak law of large numbers.

Problem 3

Part a