Lab Exercise Sampling 2

Daniel Kiser

February 21, 2018

Problem 1

1) If $x = y^{1/n}$ then we take the derivative of both sides and get $dx = \frac{1}{n} y^{1/n-1}$. Since

$$f(y)dy = f(x)dx$$

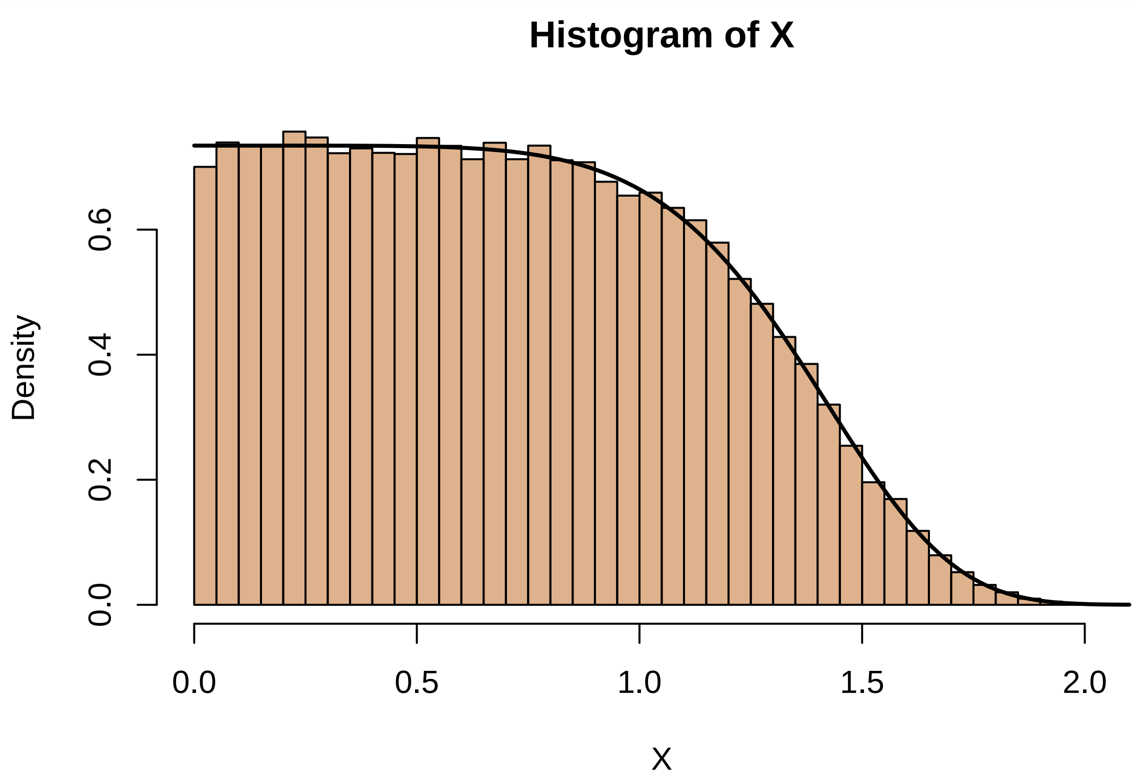
Then

$$f_y(y) = f_x(y^{1/n}) \frac{dx}{dy} \alpha \frac{1}{n} y^{1/n-1} e^{-ky}$$

2) If we compare the density of y with the Gamma function, we see that $\alpha = 1/n$ and $\lambda = k$.

3)

```
set.seed(1234)
Y <- rgamma(1e5, shape = 1/6, rate = 1/10)
X <- Y^(1/6)
hist(X, breaks = 30, freq = F, col = rgb(0.75,0.4,0.1,0.5)) # X is your sample
lambda = 1/10; alpha = 1/6
target <- function(x){exp(-lambda*x^(1/alpha))/
integrate(function(x) exp(-lambda*x^(1/alpha)),0,Inf)$value}
curve(target,lwd=2,add=T)
```



Problem 2

1) To find the cdf, we integrate $\alpha x_m x^{-\alpha-1} = -xmx^{-\alpha}$ from t to infinity, which resolves to $(\frac{x_m}{x^\alpha})^{1/\alpha}$.

```
transform <- function(alpha, xm) { x <- (xm/runif(100))^(1/alpha); return(x) }
```

2) I would generate a sample from X with $x_m = 1$ and $\alpha = 3$, and then find the median of the sample:

```
set.seed(1234)
xm <- 1
alpha <- 3
sample <- transform(alpha, xm)
theoretical_median <- xm * 2^(1/alpha)
theoretical_median
```

```
## [1] 1.259921
```

```
median(sample)
```

```
## [1] 1.372391
```

```
dif <- theoretical_median - median(sample)
dif
```

```
## [1] -0.1124697
```

The median given be the sample is very close to the theoretical median, with a difference of only -0.1124697. 3)

```
theoretical_mean <- (xm * alpha)/(alpha - 1)
theoretical_mean
```

```
## [1] 1.5
```

```
mean(sample)
```

```
## [1] 1.568003
```

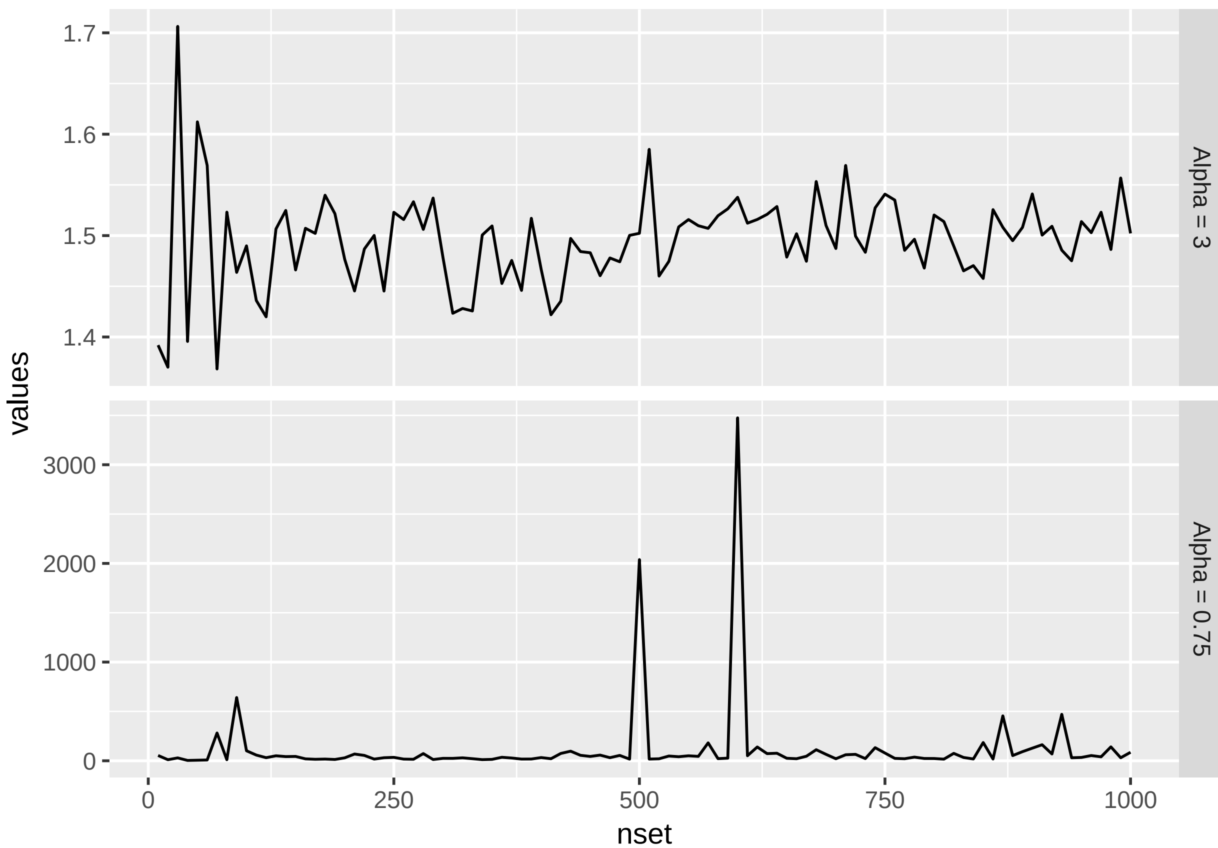
```
dif <- theoretical_mean - mean(sample)
dif
```

```
## [1] -0.06800322
```

The difference between the theoretical mean and the Monte Carlo mean is only -0.06800322. 4)

```
set.seed(1234)
transform <- function(alpha, xm, n) { x <- (xm/runif(n))^(1/alpha); return(x) }
xm <- 1
alpha1 <- 0.75
alpha2 <- 3
nset = 10*{c(1:1e2)}

paretomeans.1 = paretomeans.2 = NULL # initialize

for(i in 1:length(nset)){
  n = nset[i]
  ## mean(transform(alpha, xm, n)) returns the Monte Carlo mean

  paretomeans.1 = c(paretomeans.1,mean(transform(alpha1, xm, n)))
  paretomeans.2 = c(paretomeans.2,mean(transform(alpha2, xm, n)))
}
```

```r
pareto.data = rbind(data.frame(values = paretomeans.2,
                               type= "Alpha = 3"),
                    data.frame(values = paretomeans.1,
                               type= "Alpha = 0.75"))
pareto.data = cbind(pareto.data, nset)

library(ggplot2)
ggplot(pareto.data, aes(x = nset, y = values, group = type))+
  geom_line()+facet_grid(type~., scales = "free_y")
```



It appears that when alpha is greater than 3, the Monte Carlo mean converges to 1.5 as n increases. However, if alpha is less than 1 (such as 0.75), the Monte Carlo mean never converges.