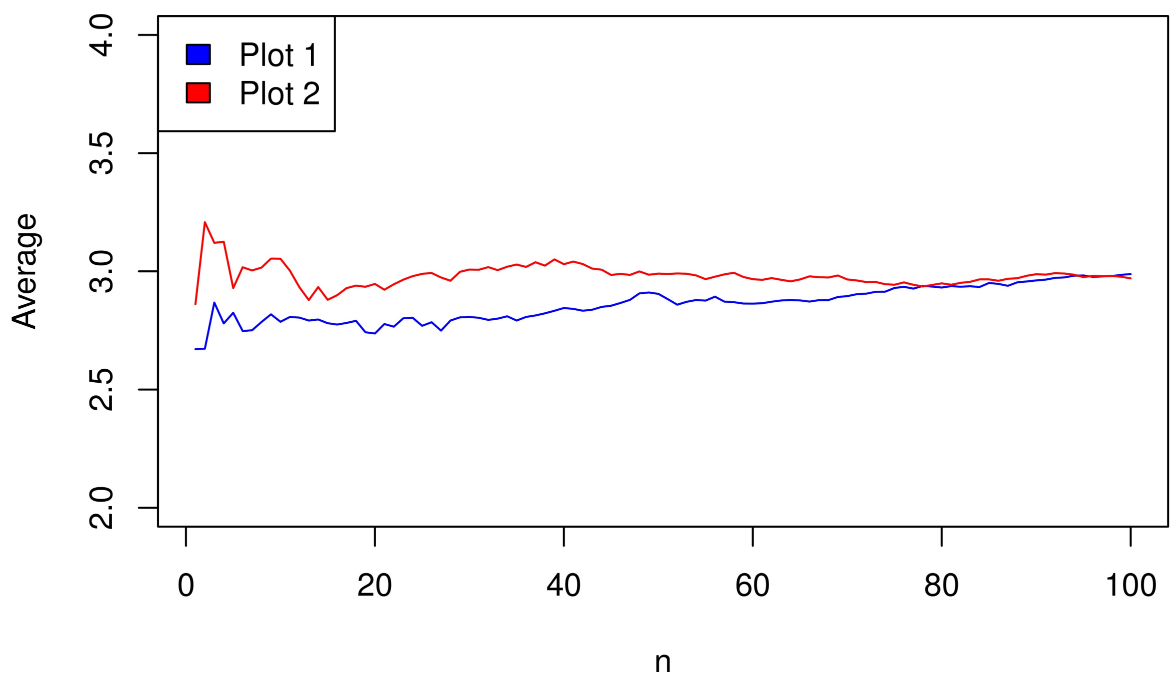
```
# show at least two simulations on the same plot with different colors
# if you're using the base R plotting package then you'll want to
# make sure that you have the same xlim,ylim, xlab, ylab for both.
```

J1.4:

In the 2 graphs plotted below, we see that although both these graphs start off at different values and have quite drastic changes in the beginning (from about 0 to 20), both the graphs start spiking significantly less as the values of n get higher, meaning they settle down at the mean value (which in this case is 3). A few subtle differences are where the graph initially starts and the size of the spikes in the beginning but as the graphs start to reach higher values, they start to follow a very similar trend.

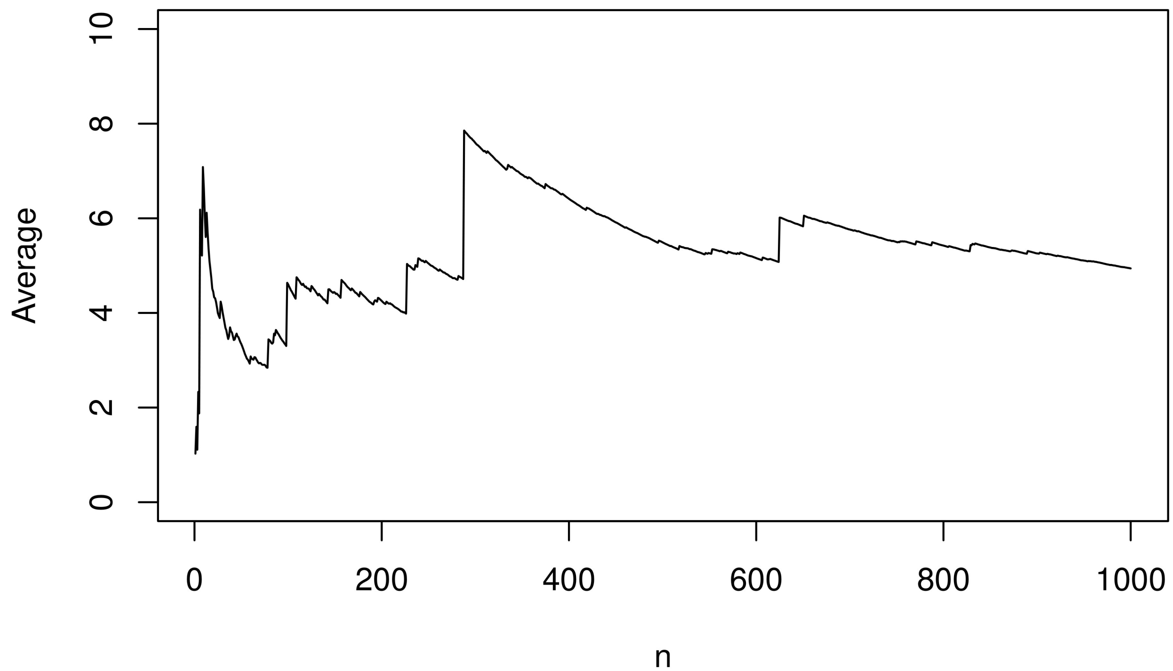
```
n1<-c(1:100)
weight<-rnorm(100, 3, 0.5)
s<-cumsum(weight)
plot(s/n1, xlab = "n", ylab = "Average", ylim = c(2,4), type="l", col="blue")
weight<-rnorm(100, 3, 0.5)
s<-cumsum(weight)
lines(s/n1, col="red")
legend("topleft", c("Plot 1", "Plot 2"), fill = c("blue", "red"))
```



J1.8:

We see that this simulation is quite similar to the one that was given to us in the fact that it has random spikes, meaning there are certain points at which the graph suddenly leaps up. This could be because of the fact that as the lecture highlighted, Cauchy random variables don't have a mean. Therefore, in contrast to the other graph in which the value of the mean started to settle down for larger values of n, in this case, the values don't converge to a single mean value.

```r
set.seed(14)
n<-c(1:1000)
y<-abs(rcauchy(1000))
s<-cumsum(y)
plot(s/n, xlab="n", ylab = "Average", ylim = c(0,10), type="l")
```



Basic Monte-Carlo Exercise:

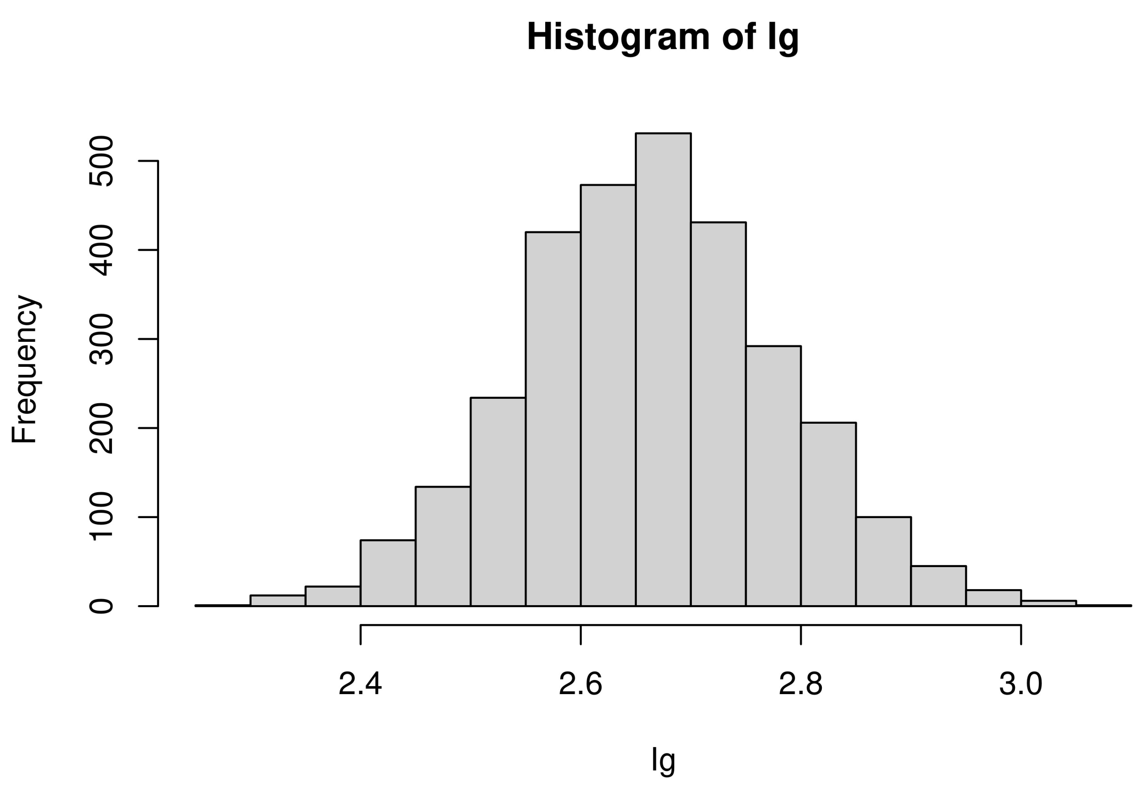
1. Use Simple Monte-Carlo integration with n = 400 to find $\displaystyle\int_0^2 x^2dx$, (make sure to include your code).

```
x<-runif(400, 0, 2)
g<-x^2
2*mean(g)
```

```
## [1] 2.623313
```

2. Repeat the process you did above 3000 times and give a histogram of the 3000 Simple Monte Carlo estimates of $\int_0^2 x^2dx$. Briefly describe where the Simple Monte Carlo estimates are centered and how spread out they are, using statistics.

```
Ig<-numeric(1000)
for(i in 1:3000){x<-runif(400, 0, 2);
g<-x^2;
Ig[i]<-2*mean(g);}
hist(Ig)
```




```
summary(Ig)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   2.297   2.584   2.662   2.663   2.739   3.051
```

```r
mean(Ig)
```

```
## [1] 2.663194
```

```r
var(Ig)
```

```
## [1] 0.01380284
```

```r
sd(Ig)
```

```
## [1] 0.1174855
```