Simulations and the central limit theorem

Patrick Breheny

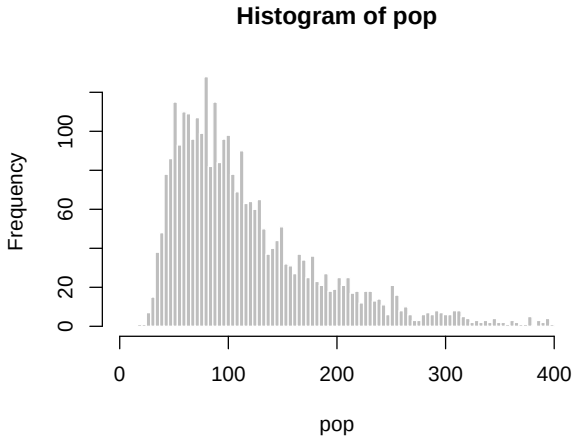
October 1, 2014

The purpose of today's lab is to look at the central limit theorem from a computational simulation perspective. In lecture we saw the theoretical result; simulations provide a powerful way to investigate how well the theory works in practice.

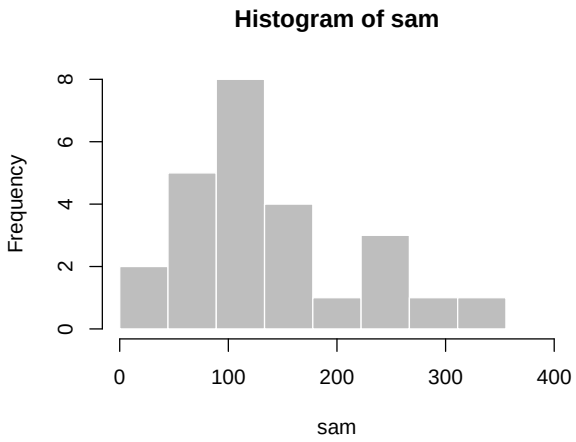
1 Simulation: NHANES lipid data

As part of the NHANES study, the triglyceride levels of 3,026 adult women were measured. Triglycerides, the main constituent of both vegetable oil and animal fat, have been linked to atherosclerosis, heart disease, and stroke. Let's consider this whole group of 3,026 women the "population" for the purposes of our simulation, and that we are going to conduct a study of this population by taking a small sample of, say, 25 women from it. So let's do this and take a look at the distribution of triglycerides in our population and in the sample:

```
> lipids <- read.delim("http://myweb.uiowa.edu/pbreheny/data/lipids.txt")
> pop <- lipids$TRG
> sam <- sample(pop, 25)
> hist(pop, col="gray", border="white", breaks=seq(0, 400, length=99))
```




```
> hist(sam, col="gray", border="white", breaks=seq(0, 400, length=10))
```

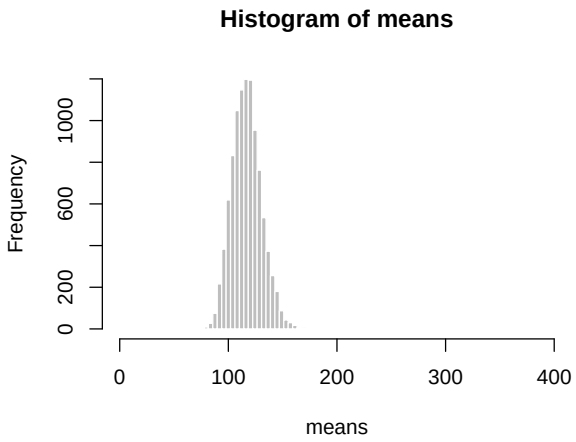



```
> mean(pop)

[1] 116.9

> mean(sam)

[1] 140.2
```

It's worth noting that (a) the distribution of triglycerides in the population is clearly right-skewed, (b) the sample looks reasonably representative of the population (as it should, since it's a random sample), and (c) the sample mean and population mean are reasonably close, but the sample mean is clearly off by a bit in terms of estimating the population mean. Of course, this is just one sample; the means of other random samples might be much further away from 116.9, or much closer.

What the central limit theorem deals with is the distribution of the sample mean. To see that distribution, we'll have to repeat the above sampling process many times and obtain many sample means. This can be done in `R` using a `for` loop:

```
> N <- 10000          ## Number of simulations to run
> n <- 25             ## Sample size
> means <- numeric(N) ## Setting up an empty vector
> for (i in 1:N) {
+   sam <- sample(pop, n)
+   means[i] <- mean(sam)
+ }
> hist(means, col="gray", border="white", breaks=seq(0, 400, length=99))
```

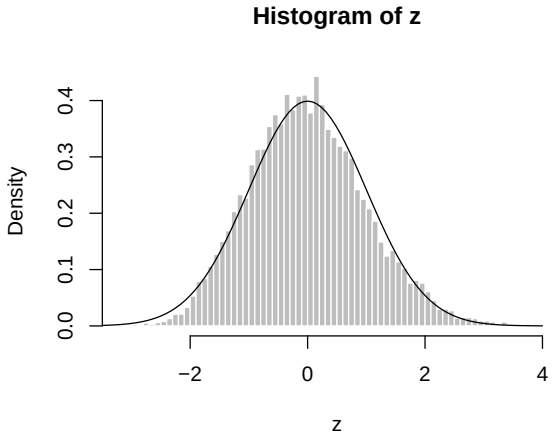


Note that, at least qualitatively, the central limit theorem seems to be holding up:

- The distribution of the means seems centered around the population mean of 117
- The spread of the distribution of the means is clearly much smaller than the spread in the original distribution of TRG values
- The shapes of the distributions are not the same; in particular, the distribution of means looks much less skewed and more normal-like

Let's put it through a more quantitative check, though, to see how exactly the CLT is working out:

```
> mean(means)

[1] 117

> SD <- sd(pop)
> SE <- SD/sqrt(n)
> sd(means)

[1] 13.5

> SE

[1] 13.59

> z <- sqrt(n) * (means-mean(pop)) / sd(pop)
> hist(z, col="gray", border="white", freq=FALSE, breaks=99)
> zz <- seq(-4, 4, length=101)
> lines(zz, dnorm(zz))
```



So the approximation seems pretty good – the distribution isn't *exactly* normal, but it's pretty close, and the expectation and SD calculations match up with our in-class derivations. Let's look at some specific distributional predictions with respect to probability:

```
> ## Probability that a sample mean is less than 100 mg/dL
> mean(means <= 100)

[1] 0.1003

> pnorm(100, mean(pop), SE)  ## Using the location-scale normal

[1] 0.1062

> pnorm((100-mean(pop))/SE) ## Using the standard normal

[1] 0.1062

> ## 90th percentile
> quantile(means, .9)

  90%
134.7

> qnorm(.9, mean(pop), SE) ## Using the location-scale normal

[1] 134.4

> qnorm(.9)*SE + mean(pop) ## Using the standard normal

[1] 134.4
```

Questions for discussion:

- Try checking the accuracy with respect to: "What's the probability that the sample mean will be between 100 and 150 mg/dL?"
- Each of the above answers is an approximation. Why? Which should you trust? What does the accuracy of each approximation depend on?

Additional exercises:

- Try re-running the above experiment(s) with different values for `N`, such as 100 and 1,000,000. What changes?
- Try re-running the above experiment(s) with different values for `n`, such as 5 and 1,000. What changes?

2 Simulation: Binomial CI coverage

Now that we've gotten the hang of simulations, let's carry out a simulation to see what the coverage of the Clopper-Pearson interval *really* is. From theory, it must have at least 95% coverage, but is the coverage exactly 95%, or it is, say, 99%.

Let's let π denote the true probability of success; we'll start off supposing that $\pi = 0.25$ and $n = 10$. Let's carry out the simulation. Note that with the `rbinom` function, we can actually do the sampling outside of a `for` loop, but we still need the `for` loop to calculate the CI:

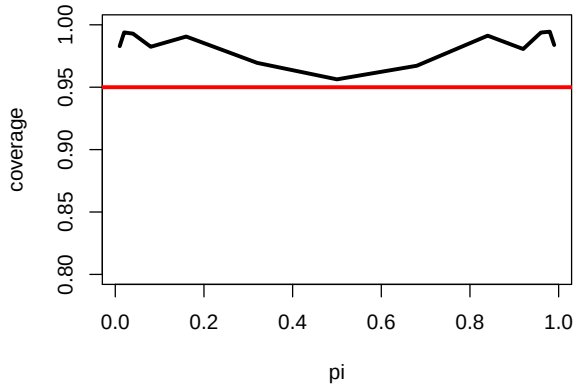
```
> N <- 10000
> n <- 10
> pi <- 0.25
> x <- rbinom(N, prob=0.25, size=n)
> covered <- numeric(N)
> for (i in 1:N) {
+   ci <- binom.test(x[i], n)$conf
+   covered[i] <- (ci[1] < pi) & (pi < ci[2])
+ }
> mean(covered)

[1] 0.9786
```

So our interval is actually fairly conservative here: its coverage is about 98%. What if we re-run the simulation with $n = 50$?

It would be interesting to see what happens to the coverage as a function of π. We can accomplish this with *nested* `for` loops. This actually takes a little while to run. It would be nice if we had a little progress bar, so I'm providing you with one:

```
> source("http://myweb.uiowa.edu/pbreheny/571/f14/labs/displayProgressBar.R")
> N <- 10000
> n <- 20
> pi <- c(0.01, 0.02, 0.04, 0.08, 0.16, 0.32, 0.5, 0.68, 0.84, 0.92, 0.96, 0.98, 0.99)
> coverage <- numeric(length(pi))
> for (j in 1:length(pi)) {
+   x <- rbinom(N, prob=pi[j], size=n)
+   covered <- numeric(N)
+   for (i in 1:N) {
+     ci <- binom.test(x[i], n)$conf
+     covered[i] <- (ci[1] < pi[j]) & (pi[j] < ci[2])
+   }
+   coverage[j] <- mean(covered)
+   displayProgressBar(j, length(pi))
+ }
```

```
Progress:
|              |
 *************

> plot(pi, coverage, type="l", lwd=3, ylim=c(0.8,1))
> abline(h=0.95, col="red", lwd=3)
```



So our Clopper-Pearson interval seems quite conservative when π is close to 0 or 1, but pretty close to the *nominal coverage* when π is close to 0.5. As guaranteed by theory, the coverage is indeed always at least 95%.

There are lots of directions we could head here:

- Re-run the simulation above with a different sample size
- Try a similar simulation, only keeping π fixed and plotting coverage vs. `n`
- Include one or both of the Bayesian intervals and see what their coverage looks like

I'm not sure how much time we'll have left in lab for all this, so we'll do what we can and pick up where we left off next time.