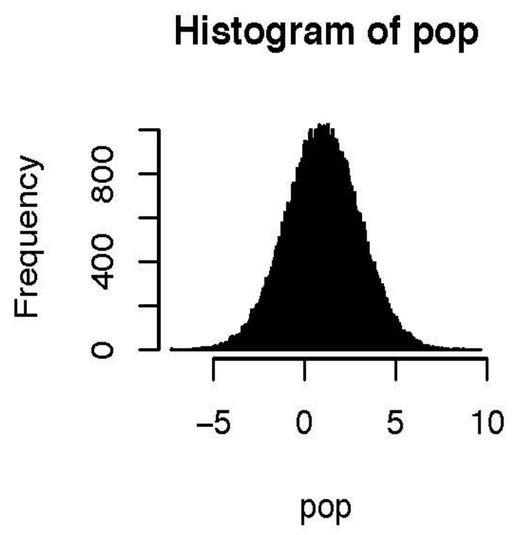
4 Sampling Distributions of the Sample Mean and Median

4.1 Sampling Distribution of the Mean

4.1.1 Normal Population

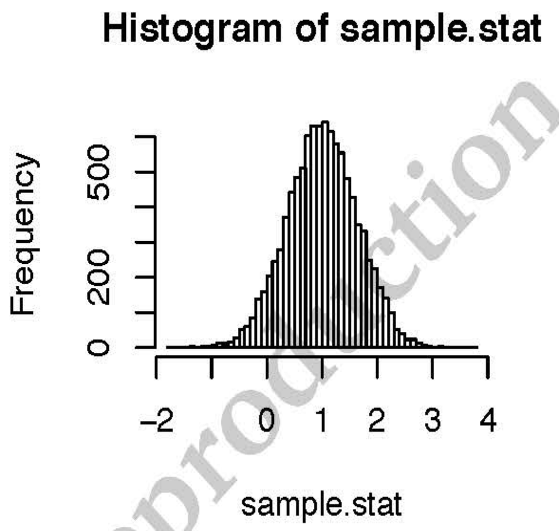
Instead of taking samples from a normal population, using rnorm(), we are going to take ONE huge sample from a normal population, using rnorm(), and then just treat **it** as our population. The main reason for this is mostly to set the stage for something called "bootstrapping," which we will study later.

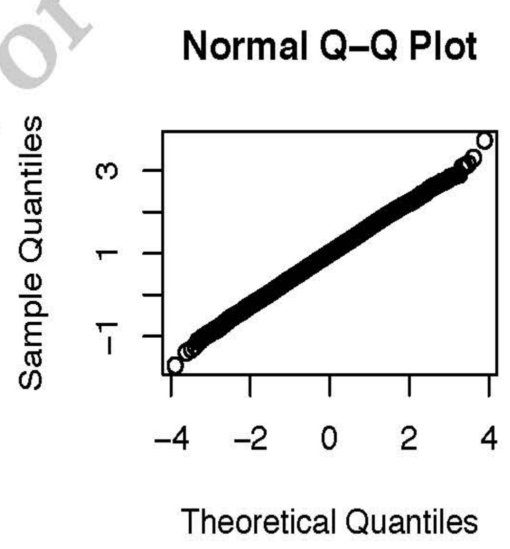
```
N <- 100000  # Let N be the population size.
pop <- rnorm(N, 1, 2)  # Take a random sample and treat it as pop.
pop.mean <- mean(pop)  # This is mu, the population mean.
pop.sd <- sd(pop)  # This is sigma, the pop standard deviation.
pop.median <- median(pop)  # Get the population median, for later.
c(pop.mean, pop.sd, pop.median)  # Print them for comparison, below.

[1] 1.002 2.000 1.002

hist(pop, breaks = 400)  # This shows that the population is pretty normal.

# Experiment underlying the sampling distribution.
n.trial <- 10000  # Take 10000 samples of
sample.size <- 10  # size 10 (i.e., small) from the population.
sample.stat <- numeric(n.trial)  # Create space to store the 10000 sample means.

for (i in 1:n.trial) {
  samp <- sample(pop, sample.size, replace = T)  # Take a sample (with replacement).
  sample.stat[i] <- mean(samp)  # Compute each sample's mean.
}
mean(sample.stat)  # Compare mean of sample means

[1] 1.003

pop.mean  # with the population mean.

[1] 1.002

sd(sample.stat)  # Compare the standard deviation of sample means

[1] 0.6285

pop.sd  # with the pop standard deviation.

[1] 2

pop.sd / sqrt(sample.size)  # But compare with (pop std dev)/sqrt(n)

[1] 0.6324
```



According to the Central Limit Theorem (CLT), the sampling distribution of the sample mean should be normal. To confirm:

```
hist(sample.stat,breaks = 40)
qqnorm(sample.stat)  # Pretty normal.
```



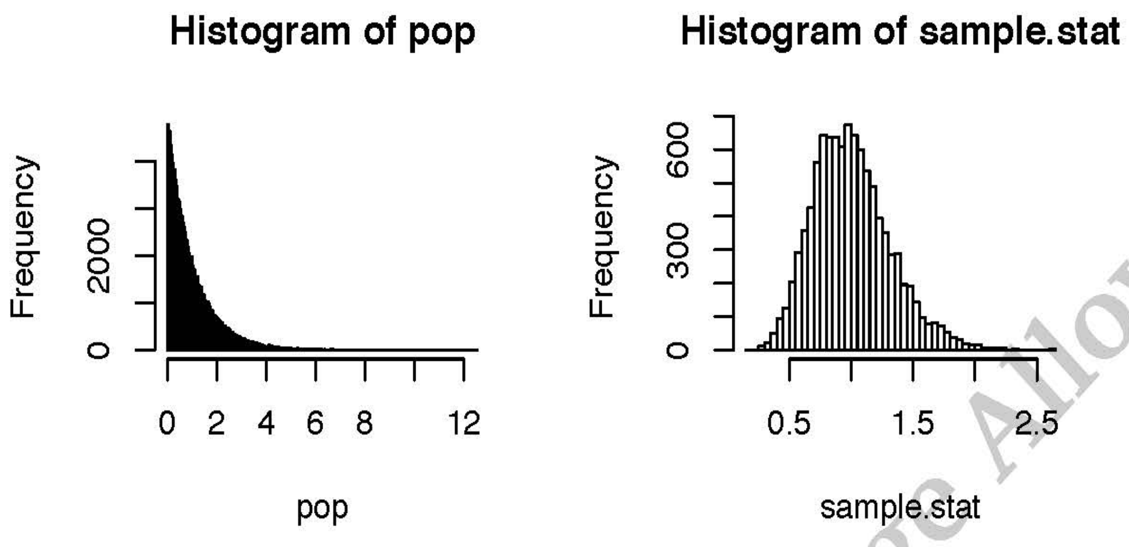


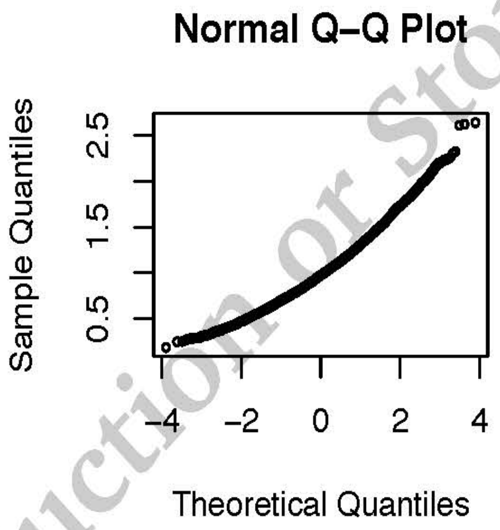
As the sample size increases, the mean of the sample means gets pretty close to the population mean, and the standard deviation of the sample means gets pretty close to the $\frac{sd(pop)}{\sqrt{n}}$. So, the CLT is confirmed.

4.1.2 Non-normal Population

```
N <- 100000
pop <- rgamma(N, 1, 1)
```

```
pop.mean <- mean(pop)
pop.sd <- sd(pop)
pop.median <- median(pop)
c(pop.mean, pop.sd, pop.median)

[1] 1.0036 1.0005 0.6957

hist(pop, breaks = 400)  # The distribution of sample means looks non-normal.

n.trial <- 10000  # Take 10000 samples of
sample.size <- 10  # size 10 (i.e., small) from the population.
sample.stat <- numeric(n.trial)  # Space for storing the 10000 sample means.

for (i in 1:n.trial) {
  samp <- sample(pop, sample.size, replace=T)  # Take a sample (with replacement).
  sample.stat[i] <- mean(samp)  # and compute each sample's mean.
}

mean(sample.stat)  # Compare mean of sample means with population mean.

[1] 1.003

pop.mean

[1] 1.004

sd(sample.stat)  # Compare the sd of sample means with population sd.

[1] 0.3129

pop.sd

[1] 1

pop.sd / sqrt(sample.size) # Compare with (pop sd)/root(n).

[1] 0.3164

hist(sample.stat, breaks = 40)
qqnorm(sample.stat, cex = 0.5)
```





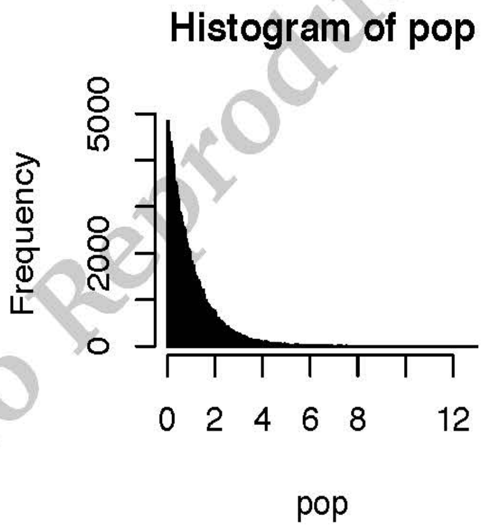
When the population is NOT normal, for small samples (10) the sampling distribution of the sample mean resists looking normal; but with larger samples (100), it is normal even though the population is not normal.

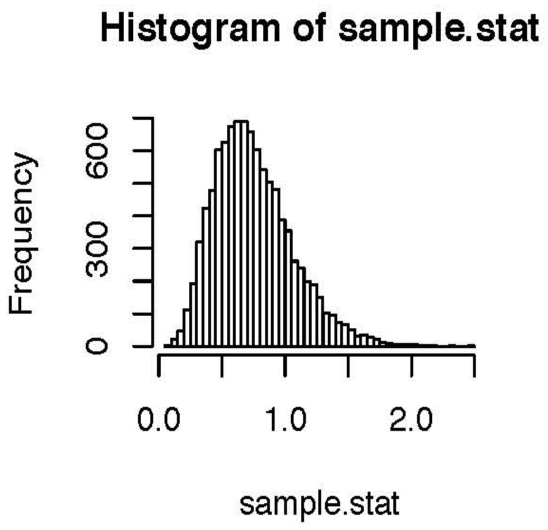
4.2 Sampling Distribution of Median

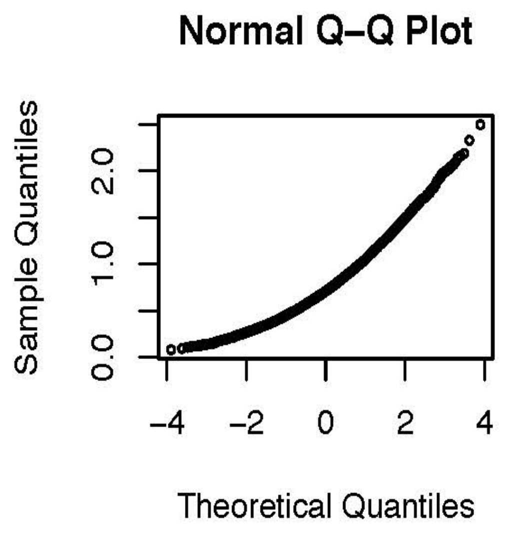
4.2.1 Non-normal Population

```
N <- 100000
pop <- rgamma(N,1,1)
pop.mean <- mean(pop)
pop.sd <- sd(pop)
pop.median <- median(pop)
c(pop.mean, pop.sd, pop.median)
```

```
[1] 1.0048 1.0077 0.6973

hist(pop, breaks = 400)

n.trial <- 10000  # Take 10000 samples of
sample.size <- 10  # size 10 (i.e., small) from the population.
sample.stat <- numeric(n.trial)  # Space for storing the 10000 sample medians.

for (i in 1:n.trial) {
  samp <- sample(pop, sample.size, replace = T)  # Take a sample (with replacement).
  sample.stat[i] <- median(samp)  # Compute each sample's MEDIAN.
}

mean(sample.stat)  # Compare the MEAN of sample MEDIANS with pop MEDIAN.

[1] 0.7512

pop.median

[1] 0.6973

sd(sample.stat)  # Compare the sd of sample MEDIANS with population sd.

[1] 0.3105

pop.sd

[1] 1.008

# Note that the formula sigma/root(n) applies only to the sample MEAN.

hist(sample.stat, breaks = 40)
qqnorm(sample.stat, cex = 0.5)
```







The sampling distribution doesn't look too normal. But if the sample size is relatively large, the distribution of a bunch of sample medians, taken from even a non-normal population, is still normal. Most statistics (e.g., sample mean, sample median, sample standard deviation, ...) ultimately end up having a normal distribution, but some require a larger sample size.

5 Confidence Interval and Bootstrapping

Now, we're going to move on from the sampling distribution, and develop the notion of a Confidence Intervals (CI). First, we will show that the formula for the CI for the population mean actually does what it is designed to do. Recall that the formula for the confidence interval (CI) for the population mean is given by:

$$\bar{x} \pm z^* \cdot \frac{\sigma}{\sqrt{n}} \tag{1}$$

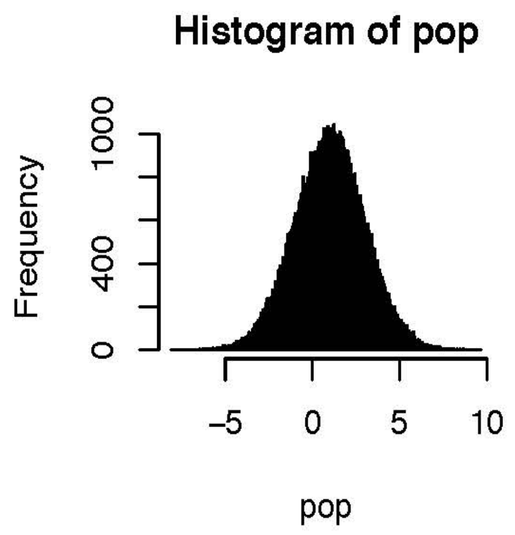
and it is designed to cover the population mean in 95% of samples taken from the population. One nontrivial part of this formula is the $\frac{\sigma}{\sqrt{n}}$, also called the standard error (std err) of the sample mean. It's nontrivial because, first we have to approximate the population std (σ) with sample std, but more importantly, we have to use math to derive it. For many statistics (other than the sample mean), the std error is difficult to derive mathematically. The other nontrivial part of the CI formula is the z^*, because it is based on the fact that the sample mean has a normal distribution. Some statistics do not.

The second task is to show that we can actually get similar answers, WITHOUT using the formula for the std err of the mean, nor the assumption of normality. This is important when simple formulas for the std err do not exist, e.g., for sample median. The main idea is called The Bootstrap: We basically treat the single sample that we have in a realistic situation as if it were the population! So, instead of sampling from the population (i.e., what we did above), bootstrap re-samples from the *sample.* It's like magic, but you'll see how it works below.

There are different kinds of CI, e.g., 1-sample, 2-sample, 1-sided, 2-sided, large-sample, small-sample, etc. And yet other kinds of CI will be covered as we proceed forward into chapters 8, 9, 10, and 11. In the following, you will also come across words like ”p-value,” or “hypothesis.” For now, you may simply ignore them. Ch8 will introduce that method, which is equivalent to the CI method.

5.1 Confidence Interval for Population Mean

```
# Create a normal population.
rm(list = ls(all = TRUE))
set.seed(1)
N <- 100000  # Sample size = 100000.
pop <- rnorm(N, 1, 2)  # Draw N samples from a normal distribution with
                       # mu = 1 and sigma = 2.
pop.mean <- mean(pop)
pop.sd <- sd(pop)
pop.median <- median(pop)
c(pop.mean, pop.sd, pop.median)

[1] 0.9955 2.0070 1.0016

hist(pop, breaks = 400)

sample.size <- 200  # Sample size.
sample.trial <- sample(pop, sample.size, replace = T)  # Here is a sample.
```



5.1.1 Calculating CI Using Formula

```
sample.stat <-  mean(sample.trial)  # Sample mean.
std.err <- sd(sample.trial) / sqrt(sample.size)  # Calculate the standard error.
sample.stat - abs(qnorm(.05 / 2)) * std.err  # Note z_star .

[1] 0.7895

sample.stat + abs(qnorm(.05 / 2)) * std.err  # Sign is correct!

[1] 1.32

# sample.stat + qt(0.05 / 2, sample.size - 1) * std.err  # For use later
# sample.stat + qt(1 - 0.05 / 2, sample.size - 1) * std.err
```

5.1.2 Calculating CI Using Built-in Function

```
t.test(sample.trial, alternative = "two.sided", conf.level = 0.95)


One Sample t-test

data:  sample.trial
t = 7.8, df = 200, p-value = 4e-13
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
 0.7878 1.3219
sample estimates:
mean of x
    1.055
```

```
# Sample.trial contains the data/measurements of x.
# "two-sided" specifies a 2-sided CI, and 0.95 is the confidence level.

# To get the confidence interval, use the following command:
t.test(sample.trial)$conf.int[1:2]

[1] 0.7878 1.3219
```

Note that answers from the two methods (by formula and by computer) are very similar. The interpretation of C.I. is that we can be 95% confident that the true mean resides in this interval.

5.2 Coverage of a Confidence Interval

In practice, you will have only one sample (samp) (and not the population (pop)), and so you will build only one CI. But here we want to confirm that the CI, the way we compute it (i.e., with our formulas or with t.test()) covers the population mean the correct percentage of time. **This is what a CI is designed to do: to have the correct coverage.**

To do so, we will draw n.trial = 100 samples of size sample.size = 90 from the normal population, above. For each sample, we will construct the 95% CI and we will make a plot that shows all 100 CIs then count how many of them cover the population mean.

```
rm(list = ls(all = TRUE))
set.seed(1)
N <- 100000
pop <- rnorm(N, 1, 2)
pop.mean <- mean(pop)
pop.sd <- sd(pop)
pop.median <- median(pop)
c(pop.mean, pop.sd, pop.median)

[1] 0.9955 2.0070 1.0016

hist(pop, breaks = 400, main = 'Histogram of Population')

n.trial <- 100  # Number of samples to draw from population.
sample.size <- 90 # Size of each sample = 90.
CI <- matrix(nrow = n.trial, ncol = 2)  # Create space to store n.trial CIs.

for (i in 1:n.trial) {
  sample.trial <- sample(pop, sample.size)  # For each sample/trial,
  CI[i, ] <- t.test(sample.trial)$conf.int[1:2]  # compute (and keep) only CI.
}

count <- 0  # Count number of CIs that cover mu.
for (i in 1:n.trial) {
  if (CI[i, 1] <= pop.mean && CI[i, 2] >= pop.mean) {
    count <- count + 1
  }
}
count

[1] 96
```

```
attend <- c(9.0, 14.0, 15.0, 12.5, 13.5, 14.5, 12.5, 8.5, 17.5, 9.5, 12.0, 11.0,
            14.0, 14.5, 14.0, 21.5, 12.5, 10.5, 17.5, 5.0, 10.5, 17.5, 16.5, 19.0,
            18.0, 15.5, 13.5, 21.5, 10.5, 17.0, 18.5, 12.0, 15.0, 17.5, 11.5,
            15.5, 17.0, 17.0, 20.0, 15.5, 12.0, 13.0, 23.0, 11.5, 14.0, 13.0, 22.5,
            8.5, 11.0, 9.5, 11.5, 17.0, 11.5, 17.5, 7.5, 8.0, 14.5, 9.5, 19.0,
            16.5, 18.5, 10.5, 16.5, 14.5, 13.5, 14.5, 12.0, 17.0, 13.0, 11.0, 12.5,
            9.0, 19.0, 15.0, 16.0, 11.0, 7.0, 22.0, 13.0, 7.5, 14.5, 13.0, 18.5,
            13.0, 18.5, 10.0, 20.5, 10.5, 17.5, 13.0, 19.5, 10.0, 13.0, 19.5, 10.5,
            14.5, 11.0, 14.5, 7.0, 7.0, 9.0, 16.0, 13.0, 19.5, 15.0, 17.0, 18.0,
            10.5, 15.0, 8.5, 10.0, 14.0, 16.0, 12.5, 13.5, 17.0)
non.attend <- c(3.0, 12.5, 8.5, 18.5, 5.5, 18.5, 7.5, 13.5, 6.5, 17.0, 11.5, 13.0,
                13.0)
```

To see if the data provide evidence for the claim that μ_1 = mean of attend is higher than μ_2 = mean of non.attend, the appropriate CI is a lower confidence bound for $\mu_1 - \mu_2$, which is equivalent to testing

$$H_0 : \mu_1 - \mu_2 \leq 0$$
$$H_1 : \mu_1 - \mu_2 > 0$$

```
t.test(attend, non.attend, alternative = "greater", conf.level = 0.95)


Welch Two Sample t-test

data:  attend and non.attend
t = 1.8, df = 14, p-value = 0.05
alternative hypothesis: true difference in means is greater than 0
95 percent confidence interval:
 0.04549     Inf
sample estimates:
mean of x mean of y
    13.98     11.42
```

This means that we can be 95% confident that the true (i.e. population) mean grade of the attending students is higher than that of the non-attending students by at least 0.045. Because 0 is not included in the CI, the "corollary" conclusion is that the mean grade of attending students **is** higher than that of the non-attending students. One often says that the difference is "statistically significant." (The same conclusion follows from the p-value; it's smaller than $\alpha = 0.05$, and so we can reject $H_0 : \mu_1 \leq \mu_2$ in favor of $H_1 : \mu_1 > \mu_2$.)

The result is statistically significant, but is it physically significant? That's a different question! In other words, how much higher is the mean of the attendees, and do we care? To answer that, look at the sample means of the two groups (last line of the output). The attending students' grade is $\frac{13.98-11.42}{11.42} \cdot 100 \approx 22\%$ higher than that of the non-attending students. That's big enough to be considered physically significant.

It's important to note that **statistical significance** and **physical significance** are two different concepts. The difference between the two means may be statistically significant, but it may be so small that no one really cares about it, i.e., it may be physically non-significant.

5.5 Bootstrap: CI without formulas

We have confirmed in 4.2 that the CI computed with the formula $\bar{x} \pm z^* \cdot \frac{\sigma}{\sqrt{n}}$ has the correct coverage property (about 95% of such CIs cover the true mean). But that conclusion is based on several assumptions:

1. We know what z^* to use in the formula.

2. We can approximate σ with the sample standard deviation.

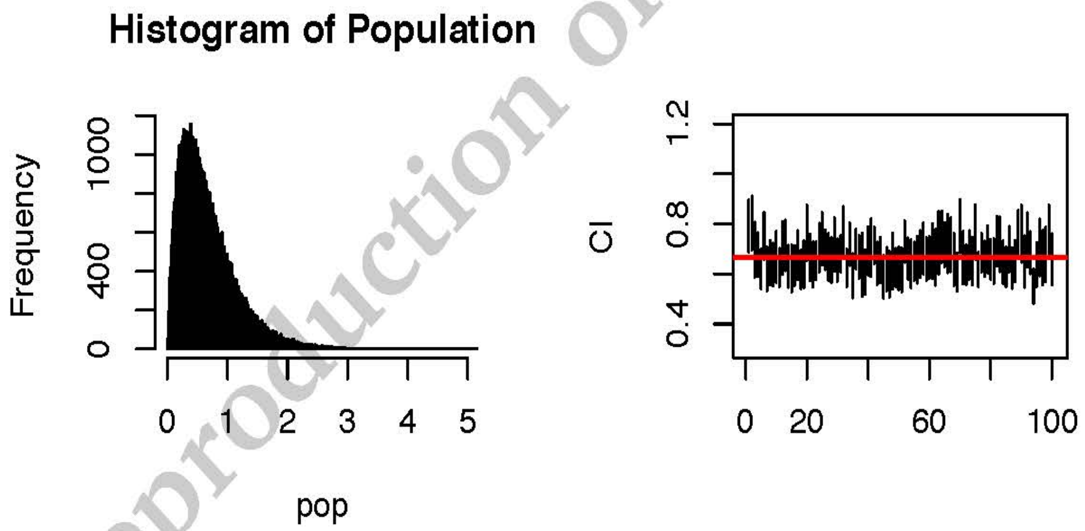
But for some statistics (e.g., sample mean) we don't even have a formula for a CI. One solution to that problem is Bootstrapping.

Example: Producing the Correct CI for Mean

Instead, of using the formula $\frac{sd(sample)}{\sqrt{n}}$ for the standard deviation of the sampling distribution of the sample mean, we can actually build (though approximately) the sampling distribution itself. This is done by taking multiple samples - called bootstrap samples - from the single observed sample! The theory behind bootstrap argues that the std dev of this "sampling distribution" is a pretty good estimate of the standard dev of the sampling distribution of the sample mean. Armed with this approximation to the sampling distribution, we can take its appropriate quantiles to give us CI; after all, $\overline{x} \pm 1.96 \cdot \frac{\sigma}{\sqrt{n}}$ mark quantiles of the true sampling distribution). So, that's the idea: to build a histogram of the sample statistic of interest by treating the sample as if it were the population.

Now, when it comes to testing the coverage properties of a CI for some parameter, recall that we take multiple samples from a population already. So, in the bootstrap approach, we will have to take multiple (bootstrap) samples from each of the samples taken from the population. For technical reasons that we won't go into, the bootstrap samples must be taken with replacement.

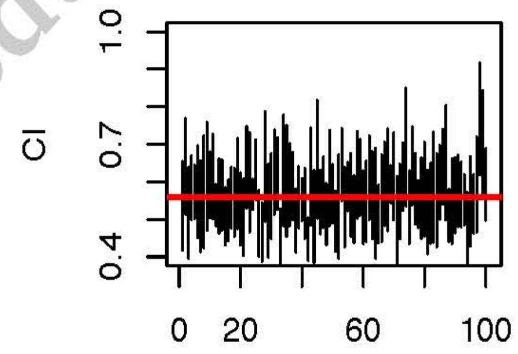
```
rm(list = ls(all = TRUE))
set.seed(1)
N <- 100000
pop <- rgamma(N, 2, 3)  # Draw from gamma instead of normal.
pop.mean <- mean(pop)
pop.sd <- sd(pop)
pop.median <- median(pop)
c(pop.mean, pop.sd, pop.median)

[1] 0.6659 0.4705 0.5590

hist(pop, breaks = 400, main = 'Histogram of Population')

n.trial <- 100
sample.size <- 90
CI <- matrix(nrow = n.trial, ncol = 2)
for (i in 1:n.trial) {
  sample.trial <- sample(pop, sample.size)  # Take a sample.
  # Now, the bootstrap block (which you have to type in):
  # For each sample, take a bootstrap sample (with replacement), and compute
  # the sampling distribution of the sample means. The appropriate quantiles
  # of this sampling distribution give the confidence interval.
  n.boot <- 100  # Number of bootstrap samples, from each sample.
  boot.stat <- numeric(n.boot)
    for (j in 1:n.boot) {
      boot.sample <- sample(sample.trial, sample.size, replace = T)
      # With replacement.
      boot.stat[j] <- mean(boot.sample)  # Store the means.
      }  # End of loop over bootstrap.
  CI[i, ] <- quantile(boot.stat, c(0.05 / 2, (1 - 0.05 / 2)))
  # CI[i,] <- c(mean(sample.trial) + qnorm(.05/2) * pop.sd / sqrt(sample.size),
  #              mean(sample.trial) - qnorm(.05/2)*pop.sd / sqrt(sample.size))
  # For small sample, replace qnorm(.05/2) with qt(0.05/2, sample.size - 1).
  # CI[i, ] <- sort(boot.stat)[(n.boot + 1) * c(0.05/2, 0.95/2)]  # See Geyer.
}  # End of loop over samples.

count <- 0
for (i in 1:n.trial) {
  if (CI[i, 1] <= pop.mean && CI[i, 2] >= pop.mean)
    count <- count + 1
    }
count

[1] 94

plot(c(1, 1), CI[1, ], ylim = c(0.3, 1.2), xlim = c(0, 101), ylab="CI", xlab = '',
     type = "l")
for (i in 2:n.trial) {
  lines(c(i, i), CI[i, ])  # Draw CIs (vertically).
}
abline(h = pop.mean, col = "red", lwd = 2)  # Draw the population mean
                                            # (horizontally).
```



It may seem like the bootstrap method makes no assumptions, and that it will work all the time. However, it turns out that it does have some problems. Some of the problems are addressed by Schenker (1985). For example, he shows that the particular version we use above (called percentile bootstrap) gives CIs which cover the population parameter less frequently than they should, especially for small samples. For example, with a sample size of 20, a 90% CI will cover the pop mean around 78% of the time.

5.5.1 Confidence Interval for Sample Median

```
n.trial <- 100
sample.size <- 90
CI <- matrix(0, n.trial, 2)
for (i in 1:n.trial) {
  sample.trial <- sample(pop,sample.size,replace=F)
  n.boot <- 100
  boot.stat <- numeric(n.boot)
  for (j in 1:n.boot) {
    boot.sample <- sample(sample.trial, sample.size, replace = T)
    boot.stat[j] <- median(boot.sample)  # Median
    }
  CI[i, ] <- quantile(boot.stat, c(0.05 / 2, (1 - 0.05 / 2)))
}

count <- 0
for (i in 1:n.trial) {
  if (CI[i, 1] <= pop.median && CI[i, 2] >= pop.median)
    count <- count + 1
}
count

[1] 96

plot(c(1, 1), CI[1, ], ylim = c(0.4, 1), xlim = c(0, 101), xlab = '', ylab = "CI",
     type = "l")
for (i in 2:n.trial) {
  lines(c(i, i), CI[i, ])
  abline(h = pop.median, col = "red", lwd = 2)
}
```



Note that the number of times that the confidence interval covers the true median is close to 95. In other words, the way we are computing a confidence interval for a population median gives us confidence intervals that cover the population median the expected number of times. In practice, when you have a **single** sample, and no population, you can use this bootstrap method to build a confidence interval for the population median.

A quick partial fix to the problem of under-coverage is proposed by Charles Geyer:

http://www.stat.umn.edu/geyer/old/5601/examp/percent.html

and it involves revising the CI line just a bit. The commented line in in the above code will let you test this idea.

Reference

Schenker, Nathaniel (1985): Qualms About Bootstrap Confidence Intervals Journal of the American Statistical Association, Vol. 80, No. 390 (Jun., 1985), pp. 360-361.