Stat 3701 Lecture Notes: Bootstrap

Charles J. Geyer

April 17, 2017

1 License

2 R

- The version of R used to make this document is 3.3.3.
- The version of the `rmarkdown` package used to make this document is 1.4.
- The version of the `knitr` package used to make this document is 1.15.1.
- The version of the `bootstrap` package used to make this document is 2017.2.

3 Relevant and Irrelevant Simulation

3.1 Irrelevant

Most statisticians think a statistics paper isn't really a statistics paper or a statistics talk isn't really a statistics talk if it doesn't have simulations demonstrating that the methods proposed work great (at least in some toy problems).

IMHO, this is nonsense. Simulations of the kind most statisticians do *prove* nothing. The toy problems used are often very special and do not stress the methods at all. In fact, they may be (consciously or unconsciously) chosen to make the methods look good.

In scientific experiments, we know how to use randomization, blinding, and other techniques to avoid biasing the results. Analogous things are never AFAIK done with simulations.

When all of the toy problems simulated are *very different* from the statistical model you intend to use for your data, what could the simulation study possibly tell you that is relevant? Nothing.

Hence, for short, your humble author calls all of these millions of simulation studies statisticians have done *irrelevant simulation*.

3.2 Relevant

But there is a well-known methodology of *relevant simulation*, except that it isn't called that. It is called the *bootstrap*.

It idea is, for each statistical model and each data set to which it is applied, one should do a simulation study of this model on data of this form.

But there is a problem: the fundamental problem of statistics, that $\hat{\theta}$ is not θ. To be truly relevant we should simulate from the true unknown distribution of the data, but we don't know what that is. (If we did, we wouldn't need statistics.)

So as a second best choice we have to simulate from our best estimate of the true unknown distribution, the one corresponding to the parameter value $\hat{\theta}$ if that is the best estimator we know.

But we know that is the Wrong Thing. So we have to be sophisticated about this. We have to arrange what we do with our simulations to come as close to the Right Thing as possible.

And bootstrap theory and methods are extraordinarily sophisticated with many different methods of coming very close to the Right Thing.

4 R Packages and Textbooks

There are two well known R packages concerned with the bootstrap. They go with two well known textbooks.

- R package `boot` is an R recommended package that is installed by default in every installation of R. As the package description says, it goes with the textbook Davison and Hinkley (1997).
- The CRAN package `bootstrap` goes with, as its package description says, the textbook Efron and Tibshirani (1993).

  The package description also says that "new projects should preferentially use the recommended package '`boot`' ". But I do not agree. The package maintainer is neither of Efron or Tibshirani, and I do not think they would agree. Whatever the politics of the R core team that make the `boot` package "recommended", they have nothing to do with the quality of the package or with the quality of the textbook they go with. If you like Efron and Tibshirani (1993), you should be using the R package `bootstrap` that goes with it.

These authors range from moderately famous (for a statistician) to very, very famous (for a statistician). Efron is the inventor of the term *bootstrap* in its statistical meaning.

5 The Bootstrap Analogy

5.1 The Name of the Game

The term "bootstrap" recalls the English idiom "pull oneself up by one's bootstraps".

The literal meaning of "bootstrap" in non-technical language is leather loops at the top of boots used to pull them on. So the literal meaning of "pull oneself up by one's bootstraps" is to reach down, grab your shoes, and lift yourself off the ground — a physical impossibility. But, idiomatically, it doesn't mean do the physically impossible; it means something like "succeed by one's own efforts", especially when this is difficult.

The technical meaning in statistics plays off this idiom. It means to get a good approximation to the sampling distribution of an estimator without using any theory. (At least not using any theory in the computation. A great deal of very technical theory may be used in justifying the bootstrap in certain situations.)

5.2 Introduction

The discussion in this section (all of Section 5) is stolen from Efron and Tibshirani (1993, Figure 8.1 and the surrounding text).

To understand the bootstrap you have to understand a simple analogy. Otherwise it is quite mysterious. I recall being mystified about it when I was a graduate student. I hope the students I teach are much less mystified because of this analogy. This appears to the untutored to be impossible or magical. But it isn't really. It is sound statistical methodology.

5.3 The Nonparametric Bootstrap

The *nonparametric bootstrap* (or, to be more precise, Efron's original nonparametric bootstrap, because others have been proposed in the literature, although no other is widely used AFAIK) is based on a nonparametric estimate of the true unknown distribution of the data.

This nonparametric estimate is just the sample itself, thought of as a finite population to sample from. Let P denote the true unknown probability distribution that we assume the data are an IID sample from, and let $\widehat{P}_n$ denote probability model that samples IID from the original sample thought of as a finite population to sample from.

As we said above, this is the Wrong Thing with a capital W and a capital T. The sample is not the population. But it will be close for large sample sizes. Thus all justification for the nonparametric bootstrap is asymptotic. It only works for large sample sizes. We emphasize this because many naive users have picked up the opposite impression somewhere. The notion that the bootstrap (any kind of bootstrap) is an exact statistical method seems to be floating around in the memeosphere and impossible to stamp out.

The bootstrap makes an analogy between the real world and a mythical bootstrap world.

	real world	bootstrap world
true unknown distribution	P	$\widehat{P}_n$
true unknown parameter	$\theta = r(P)$	$\hat{\theta}_n = r(\widehat{P}_n)$
data	$X_1, \ldots, X_n$ IID P	$X_1^*, \ldots, X_n^*$ IID $\widehat{P}_n$
estimator	$\hat{\theta}_n = t(x_1, \ldots, x_n)$	$\theta_n^* = t(x_1^*, \ldots, x_n^*)$
estimated standard error	$\hat{s}_n = s(x_1, \ldots, x_n)$	$s_n^* = s(x_1^*, \ldots, x_n^*)$
approximate pivotal quantity	$(\hat{\theta}_n - \theta)/\hat{s}_n$	$(\theta_n^* - \hat{\theta}_n)/s_n^*$

The explanation.

- In the real world we have the true unknown distribution of the data P. In the bootstrap world we have the "true" pretend unknown distribution of the data $\widehat{P}_n$. Actually the distribution $\widehat{P}_n$ is known, and that's a good thing, because it allows us to simulate data from it. But we pretend it is unknown when we are reasoning in the bootstrap world. It is the analog in the bootstrap world of the true unknown distribution P in the real world.

- In the real world we have the true unknown parameter θ. It is the aspect of P that we want to estimate. In the bootstrap world we have the "true" pretend unknown parameter $\hat{\theta}_n$. Actually the parameter $\hat{\theta}_n$ is known, and that's a good thing, because it allows to see how close estimators come to it. But we pretend it is unknown when we are reasoning in the bootstrap world. It is the analog in the bootstrap world of the true unknown parameter θ in the real world.

  $\hat{\theta}_n$ is the same function of $\widehat{P}_n$ as θ is of P.

  - If θ is the population mean, then $\hat{\theta}_n$ is the sample mean.
  - If θ is the population median, then $\hat{\theta}_n$ is the sample median.

  and so forth.

- In the real world we have data $X_1, \ldots, X_n$ that are assumed IID from P, whatever it is. In the bootstrap world we simulate data $X_1^*, \ldots, X_n^*$ that are IID from $\widehat{P}_n$.

The way we simulate IID $\widehat{P}_n$ is to take samples from the original data considered as a finite population to sample. These are samples *with replacement* because that is what IID requires.

Sometimes the nonparametric bootstrap is called “resampling” because it samples from the sample, called resampling for short. But this terminology misdirects the naive. What is important is that we have the correct analogy on the “data” line of the table.

- We have some estimator of θ, which must be a *statistic*, that is some function of the data that does not depend on the unknown parameter. In order to have the correct analogy in the bootstrap world, our estimate there must be the *same function* of the *bootstrap data.*
- Many procedures require some estimate of standard error of $\hat{\theta}_n$. Call that $\hat{s}_n$. It too must be a *statistic*, that is some function of the data that does not depend on the unknown parameter. In order to have the correct analogy in the bootstrap world, our estimate there must be the *same function* of the *bootstrap data.*
- Many procedures use so-called *pivotal quantities*, either exact or approximate.

  An exact pivotal quantity is a function of the data and the parameter of interest whose distribution *does not depend on any parameters.* The prototypical example is the t statistic
  $$\frac{\overline{X}_n - \mu}{s_n / \sqrt{n}}$$
  which has, when the data are assumed to be exactly normal, an exact t distribution on $n - 1$ degrees of freedom (which does not depend on the unknown parameters μ and σ of the distribution of the data). Note that the pivotal quantity is *a function of* μ but the sampling distribution of the pivotal quantity *does not depend on* μ *or* σ: the t distribution with $n - 1$ degrees of freedom does not does not have any unknown parameters.

  An asymptotic pivotal quantity is a function of the data and the parameter of interest whose asymptotic distribution *does not depend on any parameters.* The prototypical example is the z statistic
  $$\frac{\overline{X}_n - \mu}{s_n / \sqrt{n}}$$
  (actually the same function of data and parameters as the t statistic discussed above), which has, when the data are assumed to have any distribution with finite variance, an asymptotic standard normal distribution (which does not depend on the unknown the distribution of the data). Note that the pivotal quantity is *a function of* μ but the sampling distribution of the pivotal quantity *does not depend on the unknown distribution of the data*: the *standard* normal distribution does not does not have any unknown parameters.

  An approximate pivotal quantity is a function of the data and the parameter of interest whose sampling distribution does not depend on the unknown distribution of the data, at least not very much. Often such quantities are made by standardizing in a manner similar to those discussed above: by standardization. Any time we have some purported standard errors of estimators, we can use them to make approximate pivotal quantities.
  $$\frac{\hat{\theta}_n - \theta}{\hat{s}_n}$$
  as in the bottom left cell of the table above.

  The importance of pivotal quantities in (frequentist) statistics cannot be overemphasized. They are what allow valid exact or approximate inference. When we invert the pivotal quantity to make confidence intervals, for example,
  $$\hat{\theta}_n \pm 1.96 \cdot \hat{s}_n$$
  this is (exactly or approximately) valid *because the sampling distribution does not depend on the true unknown distribution of the data, at least not much.* If it did depend strongly on the true distribution of the data, then our coverage could be way off, because our estimated sampling distribution of the pivotal quantity might be far from its correct sampling distribution.

  As we shall see, even when we have no $\hat{s}_n$ available, the bootstrap can find one for us.

5.3.1 Cautions

5.3.1.1 Use the Correct Analogies

In the bottom right cell of the table above there is a strong tendency for naive users to replace $\hat{\theta}_n$ with θ. But this is clearly incorrect. What plays the role of true unknown parameter value in the bootstrap world is $\hat{\theta}_n$ not θ.

5.3.1.2 Hypothesis Tests are Problematic

Any hypothesis test calculates critical values or P-values using the distribution *under the null hypothesis.* But the bootstrap does not sample that unless the null hypothesis happens to be correct. Usually, we want to reject the null hypothesis, meaning we hope it is *not correct.* And in any case, we would not be doing a hypothesis test unless we did not know whether the null hypothesis is correct.

Thus the obvious naive way to calculate a bootstrap P-value, which has been re-invented time and time again by naive users, is completely bogus. It says, if $w(X_1, \ldots, X_n)$ is the test statistic of the test, then the naive bootstrap P-value is the fraction of simulations of bootstrap data in which $w(X_1^*, \ldots, X_n^*) \geq w(X_1, \ldots, X_n)$. This test typically has no power. It rejects at level α with probability α no matter how far the true unknown distribution of the data is from the null hypothesis. This is because the bootstrap samples (approximately, for large n) from the true unknown distribution, not from the null hypothesis.

Of course, there are non-bogus ways of doing bootstrap tests, but one has to be a bit less naive. For example, any valid bootstrap confidence interval also gives a valid bootstrap test. The test rejects $H_0 : \theta = \theta_0$ (two-tailed) at level α if and only if a valid confidence interval with coverage probability $1 - \alpha$ does not cover θ_0.

We won't say any more about bootstrap hypothesis tests. The textbooks cited above each have a chapter on the subject.

5.3.1.3 Regression is Problematic

If we consider our data to be IID pairs (X_i, Y_i), then the naive bootstrap procedure is to resample pairs (X_i^*, Y_i^*) where each $(X_i^*, Y_i^*) = (X_j, Y_j)$ for some j. But this mimics the *joint* distribution of X and Y and regression is about the *conditional* distribution of X and Y. So again the naive bootstrap samples the wrong distribution.

A solution to this problem is to resample *residuals* rather than data. Suppose we are assuming a parametric model for the regression function but are being nonparametric about the error distribution, as in Section 3.4.1 of the course notes about models, Part I. Just for concreteness, assume the regression function is simple $\alpha + \beta x$. Then the relation between the bootstrap world and the real world changes as follows.

	real world	bootstrap world
true unknown error distribution	P	$\widehat{P}_n$
errors	$E_1, \ldots, E_n$ IID P	$E_1^*, \ldots, E_n^*$ IID $\widehat{P}_n$
true unknown parameters	α and β	$\hat{\alpha}_n$ and $\hat{\beta}_n$
true unknown regression equation	$Y_i = \alpha + \beta x_i + E_i$	$Y_i^* = \hat{\alpha}_n + \hat{\beta}_n x_i + E_i^*$
estimators	$\hat{\alpha}_n$ and $\hat{\beta}_n$	α_n^* and β_n^*

The table is not quite as neat as before because there is no good way to say that $\hat{\alpha}_n$ and $\hat{\beta}_n$ are the same function of the regression data, thought of as a finite population to sample, as α and β are of the population, and similarly that α_n^* and β_n^* are the same function of the bootstrap data as $\hat{\alpha}_n$ and $\hat{\beta}_n$ are of the original data.

The textbooks cited above each have a chapter on this subject.

5.4 The Parametric Bootstrap

The parametric bootstrap was also invented by Efron.

Now we have a parametric model. Let P_θ denote the true unknown probability distribution that we assume the data are an IID sample from,

The bootstrap makes an analogy between the real world and a mythical bootstrap world.

	real world	bootstrap world
true unknown distribution	P_θ	$P_{\hat{\theta}_n}$
true unknown parameter	θ	$\hat{\theta}_n$
data	$X_1, \ldots, X_n$ IID P_θ	$X_1^*, \ldots, X_n^*$ IID $P_{\hat{\theta}_n}$
estimator	$\hat{\theta}_n = t(x_1, \ldots, x_n)$	$\theta_n^* = t(x_1^*, \ldots, x_n^*)$
estimated standard error	$s(\hat{\theta}_n)$	$s(\theta_n^*)$
approximate pivotal quantity	$(\hat{\theta}_n - \theta)/s(\hat{\theta}_n)$	$(\theta_n^* - \hat{\theta}_n)/s(\theta_n^*)$

We won't be so detailed in our explanation as above. The main point is that everything is the same except as with the nonparametric bootstrap except that we are using parametric estimates of distributions rather than nonparametric.

The same caution about being careful about the analogy applies as with the nonparametric bootstrap. But the other cautions do not apply. Neither hypothesis tests nor regression are problematic with the parametric bootstrap. One simply samples from the correct parametric distribution. For hypothesis tests, one estimates the parameters under the null hypothesis and then simulates that distribution. For regression, one estimates the parameters and then simulates new response data from the estimated conditional distribution of the response given the predictors.

6 Examples

6.1 Nonparametric Bootstrap

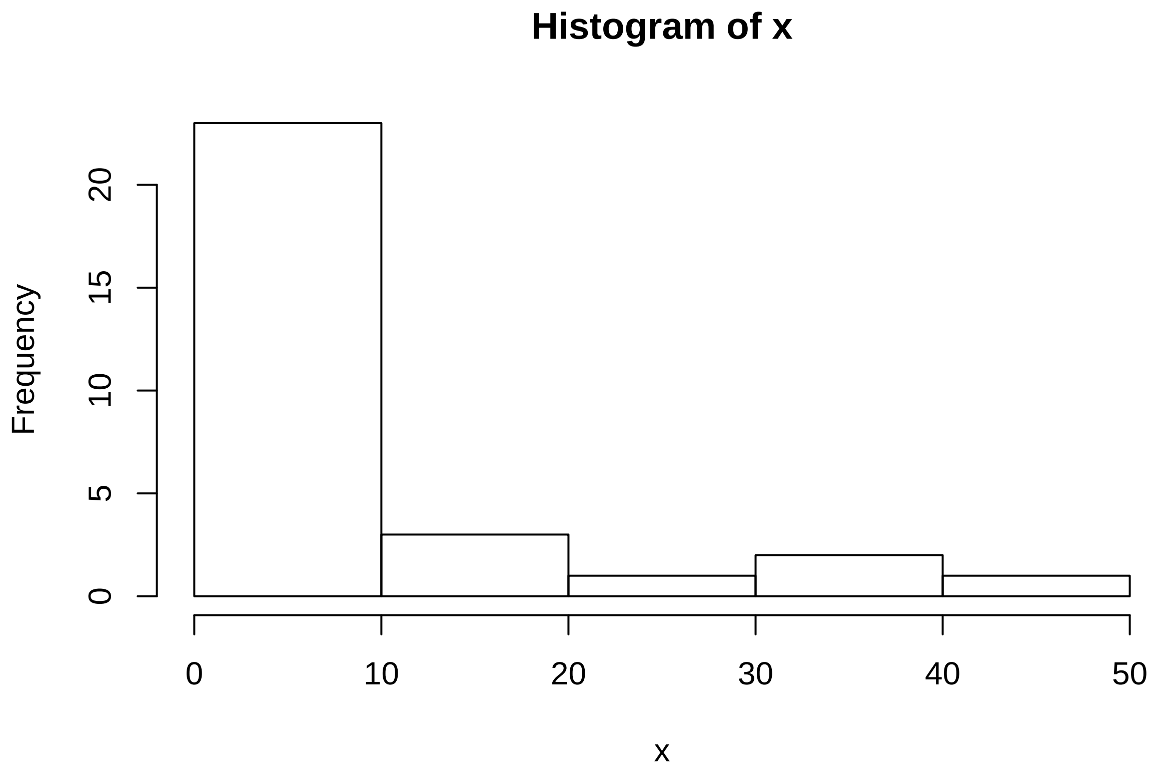
6.1.1 Data

We will use the following highly skewed data.

```r
x <- read.csv("http://www.stat.umn.edu/geyer/3701/data/boot1.csv")$x
length(x)
```

```
## [1] 30
```

```r
hist(x)
```



Suppose we wish to estimate the population mean using the sample mean as its estimator. We have the asymptotically valid confidence interval

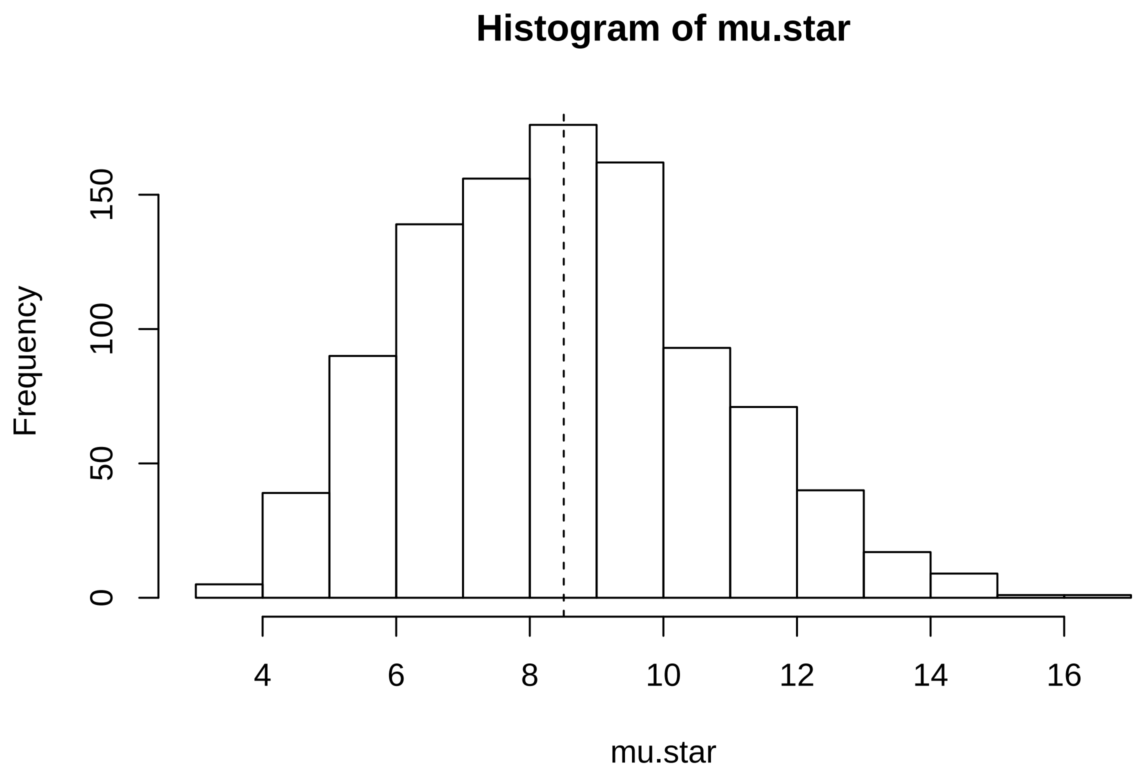
$$\bar{x}_n \pm \text{critical value} \cdot \frac{s_n}{\sqrt{n}}$$

where s_n is the sample standard deviation. We also have the rule of thumb widely promulgated by intro statistics books that this interval is valid when $n \geq 30$. That is, according to intro statistics books, $30 = \infty$. These data show how dumb that rule of thumb is.

6.1.2 Bootstrap

So let us bootstrap these data. There is an R function `boot` in the R recommended package of the same name that does bootstrap samples, but we find it so complicated as to be not worth using. We will just use a loop.

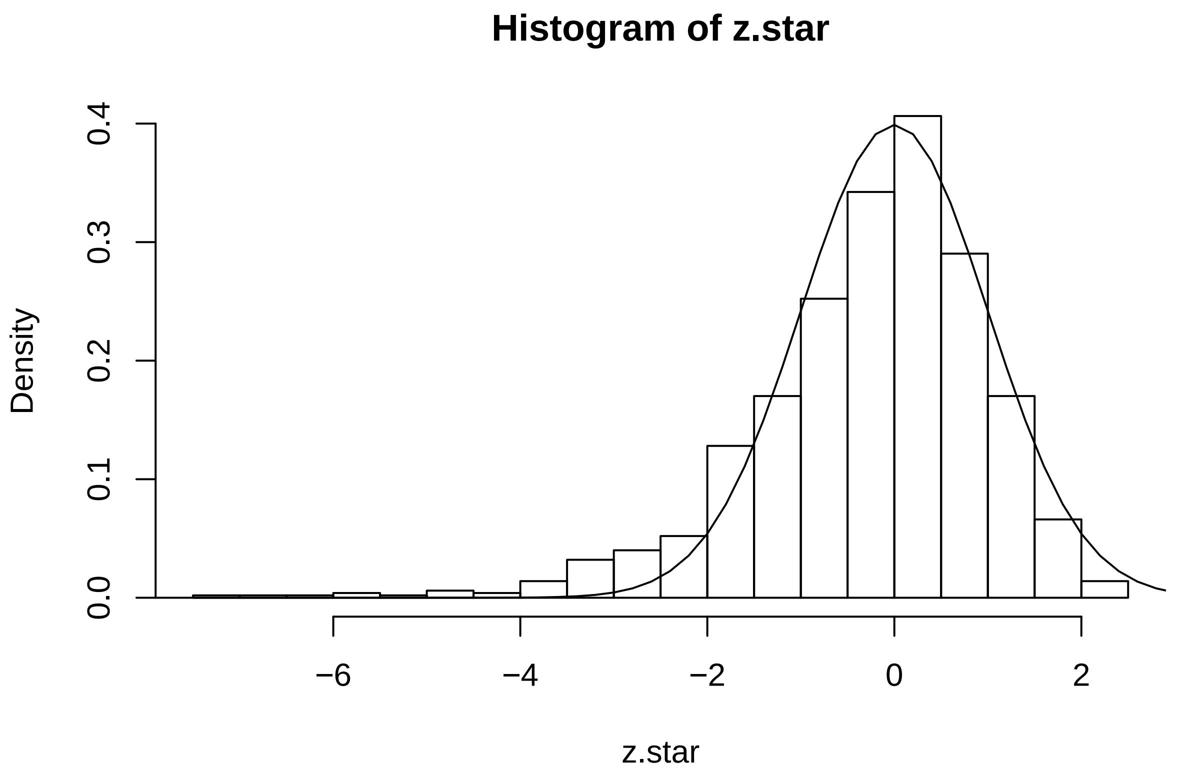
```r
mu.hat <- mean(x)
nboot <- 999
set.seed(42)
mu.star <- double(nboot)
s.star <- double(nboot)
for (iboot in 1:nboot) {
    xstar <- sample(x, replace = TRUE)
    mu.star[iboot] <- mean(xstar)
    s.star[iboot] <- sd(xstar)
}
hist(mu.star)
abline(v = mu.hat, lty = 2)
```



As the histogram shows, the sampling distribution of our estimator is also skewed (the vertical line shows $\hat{\mu}_n$).

We want to use the method of pivotal quantities here using the sample standard deviation as the standardizer.

```r
n <- length(x)
z.star <- (mu.star - mu.hat) / (s.star / sqrt(n))
hist(z.star, probability = TRUE, ylim = c(0, dnorm(0)), breaks = 20)
curve(dnorm, from = -10, to = 10, add = TRUE)
```



We can see that the distribution of `z.star` which is supposed to be standard normal (it would be standard normal when $n = \infty$) is actually for these data far from standard normal.

6.1.3 Bootstrap Confidence Interval

But since we have the bootstrap estimate of the actual sampling distribution we can use that to determine critical values.

I chose `nboot` to be 999 (a round number minus one) in order for the following trick to work. Observe that n values divide the number line into $n + 1$ parts. It can be shown by statistical theory that each part has the same sampling distribution of when stated in terms of fraction of the population distribution covered. Thus sound estimates of the quantiles of the distribution are `z[k]` estimates the $k/(n + 1)$ quantile. So we want to arrange the bootstrap sample size so that `(nboot + 1) * alpha` is an integer, where `alpha` is the probability for the critical value we want.

```
alpha <- 0.025
k <- round(c(alpha, 1 - alpha) * (nboot + 1))
k
```

```
## [1]  25 975
```

```
crit <- sort(z.star)[k]
crit
```

```
## [1] -3.205279  1.636923
```

```
quantile(z.star, probs = c(alpha, 1 - alpha))
```

```
##      2.5%     97.5%
## -3.160355  1.634057
```

The last command (the result of which we don't bother to save) shows that we are doing (arguably) the right thing. And we don't have to decide among the 9 different "types" of quantile estimator that the R function `quantile` offers. The recipe used here is unarguably correct so long as `(nboot + 1) * alpha` is an integer.

Note that our critical values are very different from

```
qnorm(c(alpha, 1 - alpha))
```

```
## [1] -1.959964  1.959964
```

which asymptotic (large sample) theory would have us use.

Our confidence interval is now

$$c_1 < \frac{\bar{x}_n - \mu}{s_n / \sqrt{n}} < c2$$

where c_1 and c_2 are the critical values. We "solve" these inequalities for μ as follows.

$$\begin{gathered}
c_1 \cdot \frac{s_n}{\sqrt{n}} < \bar{x}_n - \mu < c_2 \cdot \frac{s_n}{\sqrt{n}} \\
c_1 \cdot \frac{s_n}{\sqrt{n}} - \bar{x}_n < -\mu < c_2 \cdot \frac{s_n}{\sqrt{n}} - \bar{x}_n \\
\bar{x}_n - c_2 \cdot \frac{s_n}{\sqrt{n}} < \mu < \bar{x}_n - c_1 \cdot \frac{s_n}{\sqrt{n}}
\end{gathered}$$

(in going from the second line to the third, multiplying an inequality through by -1 reverses the inequality).

Now we use the last line of the nonparametric bootstrap analogy table. We suppose that the critical values are the same for both distributions on the bottom line (in the real world and in the bootstrap world).

Thus the bootstrap 95% confidence interval is

```
mu.hat - rev(crit) * sd(x) / sqrt(n)
```

```
## [1]  4.852995 15.666178
```

which is very different from

```
mu.hat - rev(qnorm(c(alpha, 1 - alpha))) * sd(x) / sqrt(n)
```

```
## [1]  4.131608 12.885250
```

6.1.4 Using `boott`

There is an R function `boott` in the CRAN package `bootstrap` that does this whole calculation for us.

```r
library(bootstrap)
boott(x, theta = mean, sdfun = function(x, nbootsd, theta, ...) sd(x),
    nboott = nboot, perc = c(alpha, 1 - alpha))$confpoints
```

```
##         0.025    0.975
## [1,] 4.525321 15.61439
```

where the weird signature of the `sdfun`

```
function(x, nbootsd, theta, ...)
```

is required by `boott` as `help(boott)` explains. Even though we have no use for the arguments `nbootsd` and `theta`, we have to have them in the function arguments list because the function is going to be passed them by `boott` whether we need them or not.

And what if you cannot think up a useful standardizing function? Then `boott` can find one for you using the bootstrap to the standard deviation of the sampling distribution of the estimator. So there is another bootstrap inside the main bootstrap. We call this a double bootstrap.

```r
boott(x, theta = mean, nboott = nboot, perc = c(alpha, 1 - alpha))$confpoints
```

```
##         0.025    0.975
## [1,] 4.209999 18.02688
```

Pretty cool.

6.1.5 Bootstrapping the Bootstrap

So how much better is the bootstrap confidence interval than the asymptotic confidence interval? We should do a simulation study to find out. But we don't have any idea what the population distribution is, and anyway, as argued in Section 3 above, simulations are irrelevant unless they are instances of the bootstrap. So we should check using the bootstrap. In order to not have our code too messy, we will use `boott`.

```r
boott.interval <- matrix(NaN, nboot, 2)
asymp.interval <- matrix(NaN, nboot, 2)
for (iboot in 1:nboot) {
    xstar <- sample(x, replace = TRUE)
    boott.interval[iboot, ] <- boott(xstar, theta = mean,
        sdfun = function(x, nbootsd, theta, ...) sd(x),
        nboott = nboot, perc = c(alpha, 1 - alpha))$confpoints
    asymp.interval[iboot, ] <- mean(xstar) -
        rev(qnorm(c(alpha, 1 - alpha))) * sd(xstar) / sqrt(n)
}
```

This is the first thing that actually takes more than a second of computing time, but it is still not very long.

```r
boott.miss.low <- mean(mu.hat < boott.interval[ , 1])
boott.miss.hig <- mean(mu.hat > boott.interval[ , 2])
asymp.miss.low <- mean(mu.hat < asymp.interval[ , 1])
asymp.miss.hig <- mean(mu.hat > asymp.interval[ , 2])
boott.miss.low
```

```
## [1] 0.01801802
```

```
boott.miss.hig
```

```
## [1] 0.02302302
```

```
boott.miss.low + boott.miss.hig
```

```
## [1] 0.04104104
```

```
asymp.miss.low
```

```
## [1] 0.01101101
```

```
asymp.miss.hig
```

```
## [1] 0.07807808
```

```
asymp.miss.low + asymp.miss.hig
```

```
## [1] 0.08908909
```

The bootstrap is apparently quite a bit better, but we can't really say that until we look at MCSE. For this kind of problem where we are looking at a dichotomous result (hit or miss), we know from intro stats how to calculate standard errors. This is the same as the problem of estimating a population proportion. The standard error is

$$\sqrt{\frac{\hat{p}_n(1 - \hat{p}_n)}{n}}$$

```
foompter <- rbind(c(boott.miss.low, boott.miss.hig,
    boott.miss.low + boott.miss.hig), c(asymp.miss.low, asymp.miss.hig,
    asymp.miss.low + asymp.miss.hig))
rownames(foompter) <- c("bootstrap", "asymptotic")
colnames(foompter) <- c("miss low", "miss high", "miss either")
foompter.se <- sqrt(foompter * (1 - foompter) / nboot)
library(knitr)
kable(foompter, digits=4, align="lcc", caption="Bootstrap Estimates")
```

Table 4: Bootstrap Estimates

	miss low	miss high	miss either
bootstrap	0.018	0.0230	0.0410
asymptotic	0.011	0.0781	0.0891

```
kable(foompter.se, digits=4, align="lcc",
    caption="Bootstrap Standard Errors")
```

Table 5: Bootstrap Standard Errors

	miss low	miss high	miss either
bootstrap	0.0042	0.0047	0.0063
asymptotic	0.0033	0.0085	0.0090

We say "bootstrap estimates" and "bootstrap standard errors" here rather than "Monte Carlo estimates" and "Monte Carlo standard errors" or "simulation estimates" and "simulation standard errors" because, of course, we are not doing the Right Thing (with a capital R and a capital T) which is simulating from the true unknown population distribution because, of course, we don't know what that is.

6.1.6 A Plethora of Bootstrap Confidence Intervals

The recipe for bootstrap confidence intervals illustrated here is a good one but far from the only good one. There are, in fact, a plethora of bootstrap confidence intervals covered in the textbooks cited above and even more in statistical journals.

Some of these are covered in the course materials for your humble author's version of STAT 5601. So a lot more could be said about the bootstrap. But we won't.

6.1.7 The Moral of the Story

The bootstrap can do even better than theory. Theory needs n to be large enough for theory to work. The bootstrap needs n to be large enough for the bootstrap to work. The n for the latter can be smaller than the n for the former.

This is well understood theoretically. Good bootstrap confidence intervals like the so-called bootstrap t intervals illustrated above, have the property called *higher-order accuracy* or *second-order correctness*. Asymptotic theory says that the coverage error of the asymptotic interval will be of order $n^{-1/2}$. Like everything else in asymptotics it too obeys the square root law. The actual coverage probability of the interval will differ from the nominal coverage probability by an error term that has approximate size $c/\sqrt{n}$ for some constant c (which we usually do not know, as it depends on the true unknown population distribution). For a second-order correct bootstrap interval the error will have approximate size c/n for some different (and unknown) constant c. The point is that $1/n$ is a lot smaller than $1/\sqrt{n}$.

We expect second-order correct bootstrap intervals to do better than asymptotics.

And we don't need to do any theory ourselves! The computer does it for us!

6.2 Parametric Bootstrap

We are going to use the same data to illustrate the parametric bootstrap.

But we now need a parametric model for these data. It was simulated from a gamma distribution, so we will use that.

6.2.1 The Gamma Distribution

The gamma distribution is a continuous distribution of a strictly positive random variable having PDF

$$f_{\alpha,\lambda}(x) = \frac{1}{\beta^\alpha \Gamma(\alpha)} x^{\alpha-1} e^{-x/\beta}, \qquad 0 < x < \infty,$$

where α and β are unknown parameters that are strictly positive.

It has a lot of appearances in theoretical statistics. The chi-square distribution is a special case. So is the exponential distribution, which is a model for failure times of random thingummies that do not get worse as they age. Also random variables having the F distribution can be written as a function of independent gamma random variables. In Bayesian statistics, it is the conjugate prior for several well-known families of distributions. But here we are just using it as a statistical model for data.

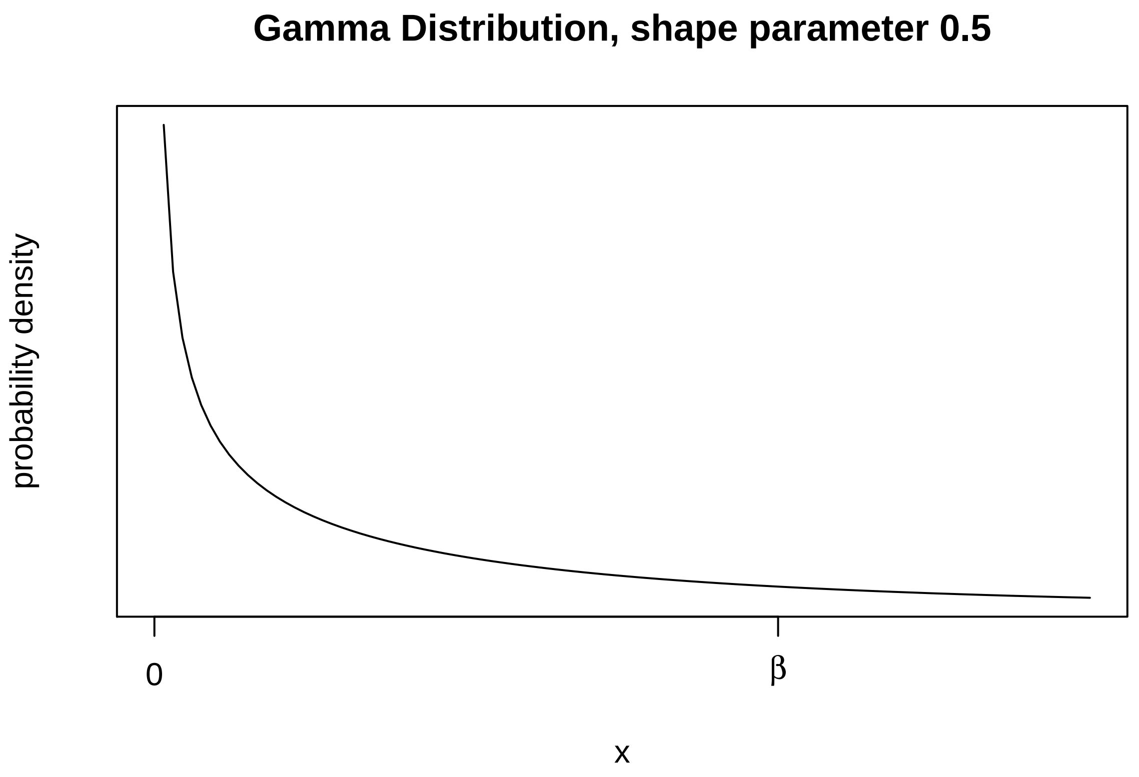
The function Γ is called the *gamma function*. It gives the probability distribution its name. If you haven't heard of it, don't worry about it. Just think of $\Gamma(\alpha)$ as a term that has to be what it is to make the PDF integrate to one.

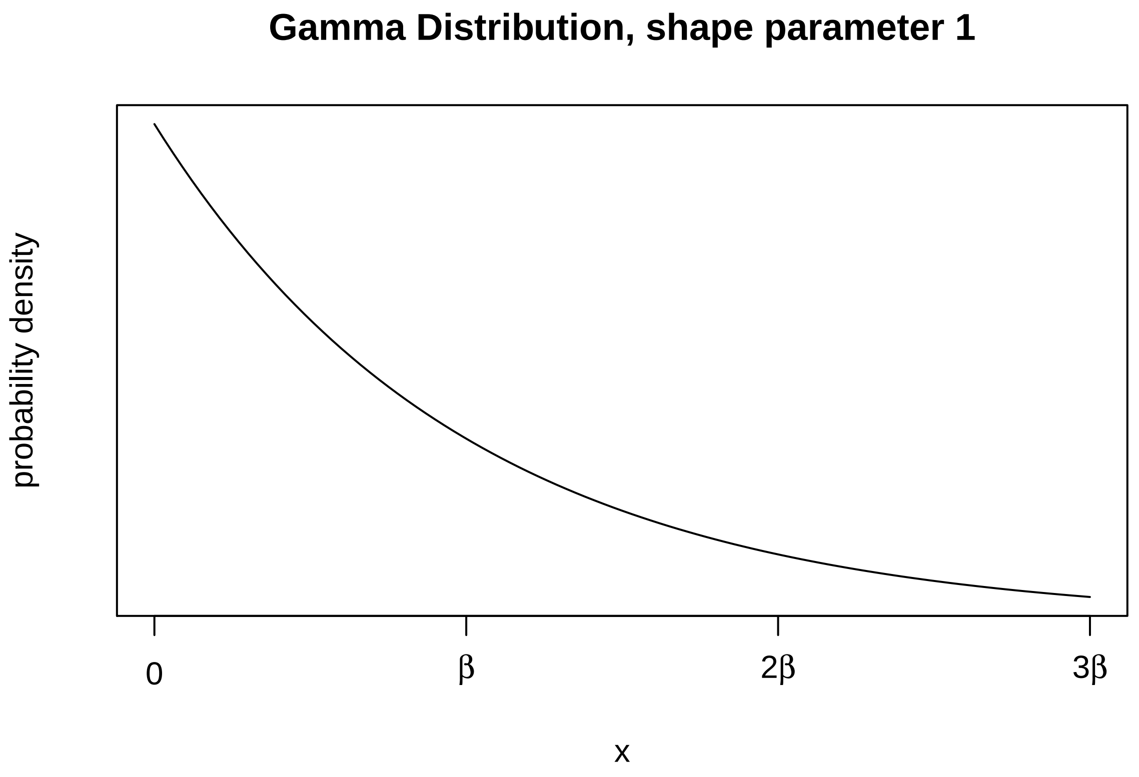
The parameter α is called the shape parameter because different α correspond to distributions of different shape. In fact, radically different.

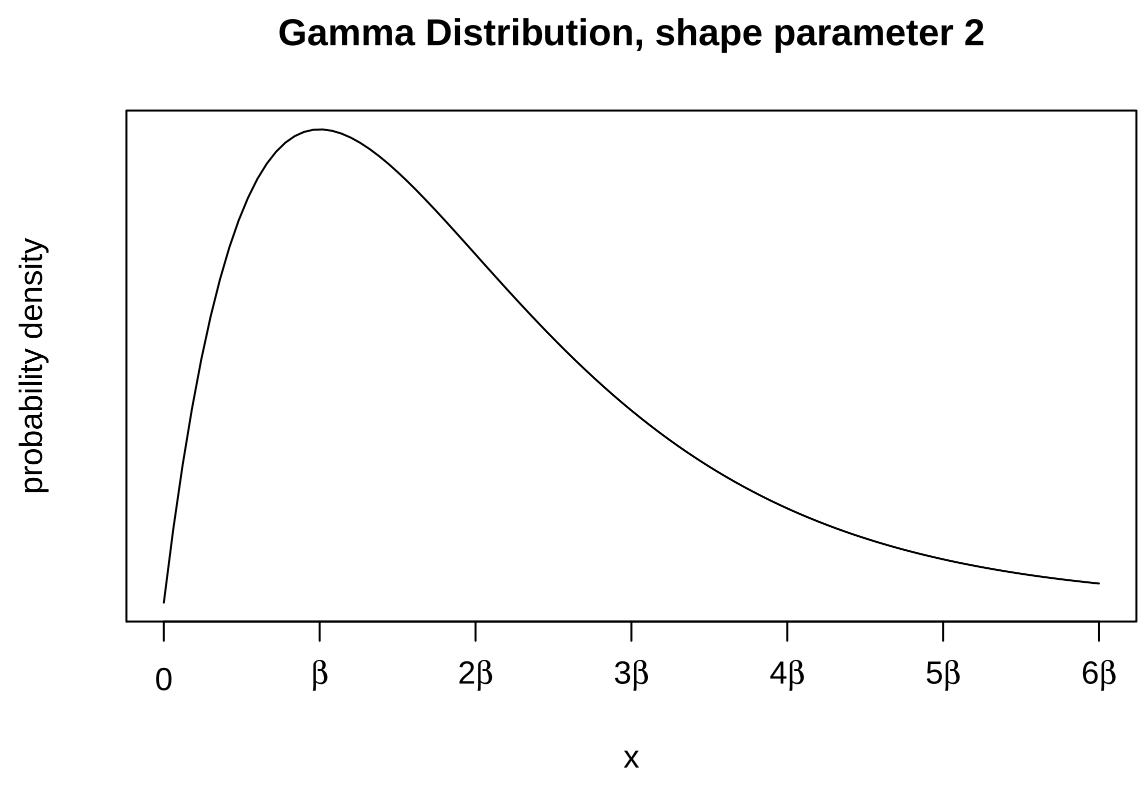
- For $\alpha < 1$ the PDF goes to infinity as $x \to 0$.
- For $\alpha > 1$ the PDF goes to zero as $x \to 0$.
- For $\alpha = 1$ the PDF goes to λ as $x \to 0$.

The parameter β is called the scale parameter because it is one. If X has the gamma distribution with shape parameter α and scale parameter one, then βX has the gamma distribution with shape parameter α and scale parameter β. So changing β does not change the shape of the distribution. We could use the same plot for all β if we don't put numbers on the axes.

```r
for (alpha in c(1/2, 1, 2)) {
    curve(dgamma(x, shape = alpha), from = 0, to = 3 * alpha,
        axes=FALSE, xlab = "", ylab = "")
    title(main = paste("Gamma Distribution, shape parameter", alpha))
    title(ylab = "probability density", line = 2)
    title(xlab = "x")
    box()
    usr <- par("usr")
    i <- seq(0, usr[2])
    if (length(i) > 2) {
        labs <- paste(i[-(1:2)], "* beta")
        labs <- parse(text = labs)
        labs <- c(expression(0), expression(beta), labs)
    } else {
        labs <- c(expression(0), expression(beta))
    }
    axis(side=1, at = i, labels = labs)
}
```


Gamma Distribution, shape parameter 0.5
probability density
0
β
x


Gamma Distribution, shape parameter 1
probability density
0
β
2β
3β
x



The mean and variance are

$$E(X) = \alpha\beta$$
$$\text{var}(X) = \alpha\beta^2$$

6.2.2 Method of Moments Estimators

Solving the last two equations for the parameters gives

$$\alpha = \frac{E(X)^2}{\text{var}(X)}$$
$$\beta = \frac{\text{var}(X)}{E(X)}$$

This suggests the corresponding sample quantities as reasonable parameter estimates.

```
theta.start <- c(mean(x)^2 / var(x), var(x) / mean(x))
theta.start
```

```
## [1]  0.4839005 17.5830111
```

These are called *method of moments estimators* because expectations of polynomial functions of data are called *moments* (mean and variance are special cases).

6.2.3 Maximum Likelihood

```
mlogl <- function(theta, x) {
    stopifnot(is.numeric(theta))
    stopifnot(is.finite(theta))
    stopifnot(length(theta) == 2)
    alpha <- theta[1]
    beta <- theta[2]
    # stopifnot(alpha > 0)
    # stopifnot(beta > 0)
    sum(- dgamma(x, shape = alpha, scale = beta, log = TRUE))
}
oout <- optim(theta.start, mlogl, x = x, method = "BFGS")
```

```
## Warning in dgamma(x, shape = alpha, scale = beta, log = TRUE): NaNs
## produced

## Warning in dgamma(x, shape = alpha, scale = beta, log = TRUE): NaNs
## produced

## Warning in dgamma(x, shape = alpha, scale = beta, log = TRUE): NaNs
## produced

## Warning in dgamma(x, shape = alpha, scale = beta, log = TRUE): NaNs
## produced

## Warning in dgamma(x, shape = alpha, scale = beta, log = TRUE): NaNs
## produced

## Warning in dgamma(x, shape = alpha, scale = beta, log = TRUE): NaNs
## produced

## Warning in dgamma(x, shape = alpha, scale = beta, log = TRUE): NaNs
## produced
```

```
oout$convergence == 0
```

```
## [1] TRUE
```

```
oout$par
```

```
## [1]  0.2877807 29.4915638
```

Since we got warnings, redo.

```
oout <- optim(oout$par, mlogl, x = x, method = "BFGS")
oout$convergence == 0
```

```
## [1] TRUE
```

```
oout$par
```

```
## [1]  0.2877506 29.4915650
```

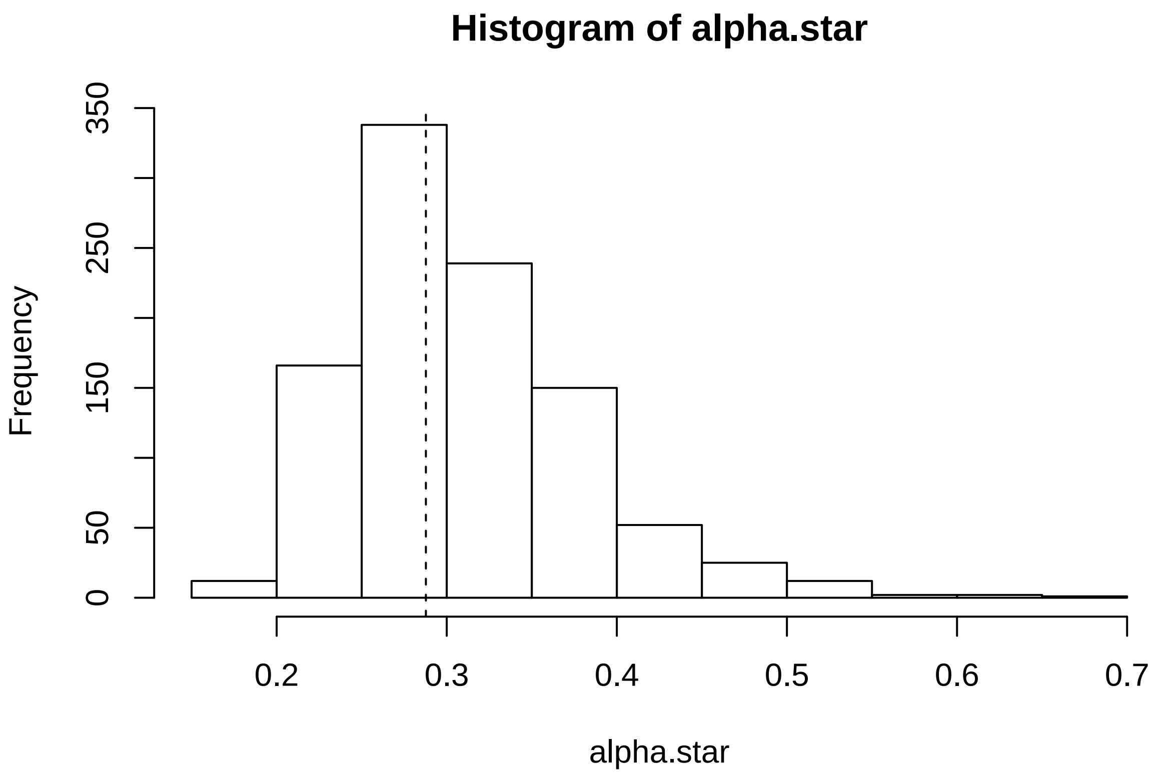
We commented out the checks that the parameter values are strictly positive because `optim` often goes outside the parameter space but then gets back on track, as it does here. We seem to have gotten the correct answer despite the warnings.

So now we are ready to bootstrap. Let us suppose we want a confidence interval for α.

6.2.4 Bootstrap

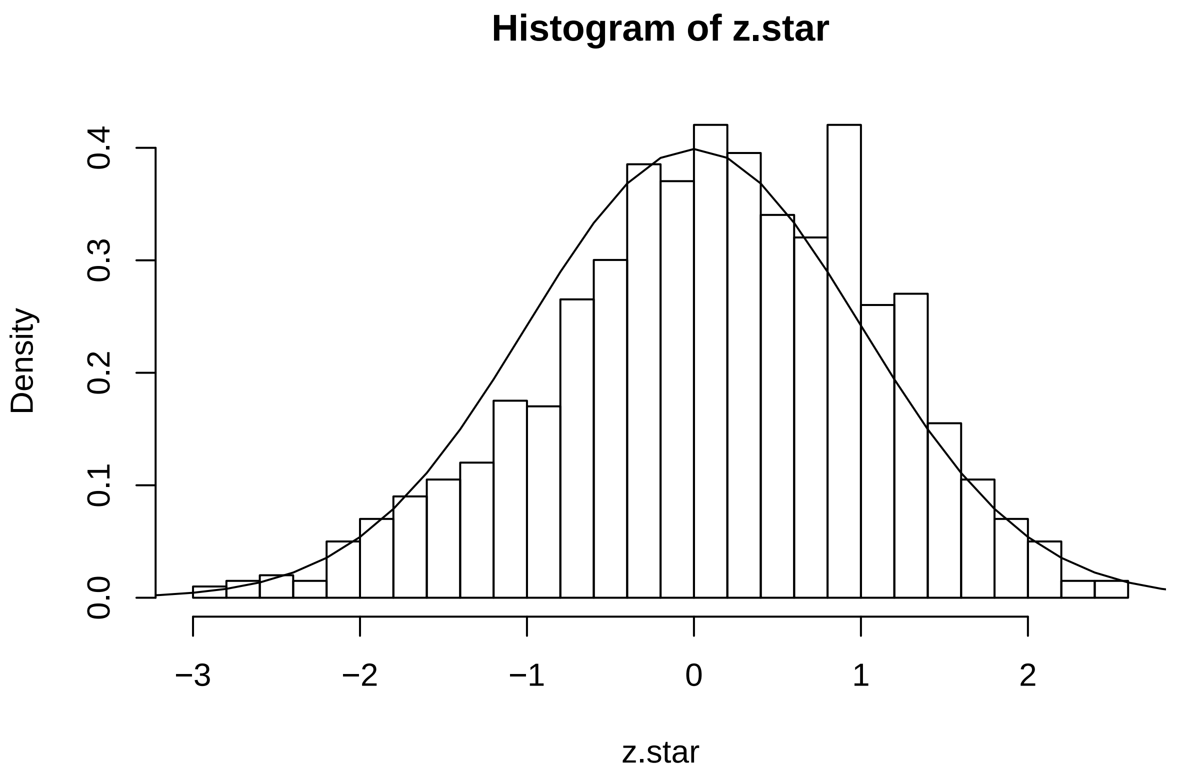
The parametric bootstrap simulates from the MLE distribution.

```
theta.hat <- oout$par
alpha.hat <- theta.hat[1]
beta.hat <- theta.hat[2]

starter <- function(x) c(mean(x)^2 / var(x), var(x) / mean(x))

alpha.star <- double(nboot)
alpha.star.asymp.var <- double(nboot)
for (iboot in 1:nboot) {
    xstar <- rgamma(length(x), shape = alpha.hat, scale = beta.hat)
    oout <- suppressWarnings(optim(starter(xstar), mlogl, x = xstar,
        method = "BFGS", hessian = TRUE))
    while (oout$convergence != 0)
        oout <- suppressWarnings(optim(oout$par, mlogl, x = xstar,
            method = "BFGS", hessian = TRUE))
    alpha.star[iboot] <- oout$par[1]
    alpha.star.asymp.var[iboot] <- solve(oout$hessian)[1, 1]
}
hist(alpha.star)
abline(v = alpha.hat, lty = 2)
```



We now follow the same "bootstrap t" idea with the parametric bootstrap that we did for the nonparametric.

```r
n <- length(x)
z.star <- (alpha.star - alpha.hat) / sqrt(alpha.star.asymp.var)
hist(z.star, probability = TRUE, breaks = 20)
curve(dnorm, from = -10, to = 10, add = TRUE)
```



This tells the asymptotics is working pretty well at $n = 30$. Perhaps the bootstrap is unnecessary. (But we didn't know that without using the bootstrap to show it.)

```r
alpha <- 0.025
k <- round(c(alpha, 1 - alpha) * (nboot + 1))
crit <- sort(z.star)[k]
crit
```

```
## [1] -1.961735  1.887759
```

```r
qnorm(c(alpha, 1 - alpha))
```

```
## [1] -1.959964  1.959964
```

Not a lot of difference in the critical values from the standard normal ones.

Since we forgot about the Hessian when estimating the parameters for the real data, we have to get it now.

```r
oohess <- optimHess(theta.hat, mlogl, x = x)
oohess
```

```
##           [,1]        [,2]
## [1,] 396.79943 1.017240017
## [2,]   1.01724 0.009977221
```

```r
alpha.hat.se <- sqrt(solve(oohess)[1, 1])
alpha.hat - rev(crit) * alpha.hat.se
```

```
## [1] 0.1774826 0.4023396
```

```
alpha.hat - rev(qnorm(c(alpha, 1 - alpha))) * alpha.hat.se
```

```
## [1] 0.1732649 0.4022362
```

6.2.5 The Moral of the Story

The moral of the story here is different from the nonparametric story above. Here we didn't need the bootstrap, and the confidence interval it made wasn't any better than the interval derived from the usual asymptotics of maximum likelihood.

But we didn't know that would happen until we did it. If anyone ever asks you "How do you know the sample size is large enough to use asymptotic theory?", this is the answer.

If the asymptotics agrees with the bootstrap, then both are correct. If the asymptotics does not agree with the bootstrap, use the bootstrap.