Bootstrap Example and Sample Code

1 Bootstrap Example

This section will demonstrate how the bootstrap can be used to generate confidence intervals.

Suppose we have a sample of data from an exponential distribution with parameter $\lambda$, i.e. our data come from a distribution with density

$$f(x|\lambda) = \lambda e^{-\lambda x}.$$

Recall that the MLE is given by $\hat{\lambda} = 1/\bar{X}$ and that $\hat{\lambda}$ has asymptotic variance equal to $\lambda^2/n$. Since we know that the MLE is asymptotically normally distributed, we can form a $(1 - \alpha)100\%$ confidence interval for $\lambda$ by using our estimate of the asymptotic variance based on our MLE and taking $[\hat{\lambda} - z(1 - \alpha/2)\hat{\lambda}/\sqrt{n}, \hat{\lambda} + z(1 - \alpha/2)\hat{\lambda}/\sqrt{n}]$.

Here is an example with R code generating the sample, calculating the MLE, and computing a 95% confidence interval for $\lambda$

```r
my.Test.Data<-rexp(100,3)
mean(my.Test.Data)
# [1] 0.30125: AN example of what happened
1/mean(my.Test.Data)
# [1] 3.32
empirical.lambda<-1/mean(my.Test.Data)

#Normal theory quantiles:
CIlow<-empirical.lambda-qnorm(0.975)*empirical.lambda/sqrt(100)
CIhigh<-empirical.lambda+qnorm(0.975)*empirical.lambda/sqrt(100)
#[2.67, 3.97]
```

We see that we generated a sample of size 100 from the exponential distribution with parameter $\lambda = 3$. The mean of our data set is 0.301, and therefore our maximum likelihood estimate for $\lambda$ is 3.32. Since we know the MLE is asymptotically normal, we can construct a confidence interval using our estimate of the asymptotic variance of our MLE and the quantiles of the standard normal distribution. In this case, we get [2.67, 3.97] for our confidence interval.

But suppose we were not working with the MLE, or suppose we did not want to use the asymptotic distribution to form our confidence intervals. Then we could use the bootstrap to estimate the distribution of $\hat{\lambda}$ and create bootstrap confidence intervals for $\lambda$.

First, we form our set of bootstrap estimates of our parameter by generating $B$ random samples of size $n = 100$ from the exponential distribution with parameter $\lambda$ and using each of these samples to get a new estimate of our model parameter, $\hat{\lambda}^{(b)}$. 
In general, we don’t necessarily know anything about the distribution of the new parameter estimates \( \hat{\lambda}^{(b)} \). But, although we may not have a perfect idea of the shape of this distribution, we can calculate quantiles \( q^*(\alpha) \) from it, where we have \( q^*(\alpha) = \hat{\lambda}^{(b)} \) such that a fraction \( \alpha \) of the other bootstrap parameter estimates are less than or equal to \( \hat{\lambda}^{(b)} \). [For example, if we had 1000 bootstrap samples, the quantile \( q^*(0.05) \) would be the 50th largest observation.] This means we can write

\[
P(q^*(\alpha/2) \leq \hat{\lambda}^{(b)} \leq q^*(1 - \alpha/2)) = 1 - \alpha
\]

Now suppose we want to look at the distribution of \( \hat{\lambda}^{(b)} - \hat{\lambda} \). From the expression above, we can see

\[
P(q^*(\alpha/2) - \hat{\lambda} \leq \hat{\lambda}^{(b)} - \hat{\lambda} \leq q^*(1 - \alpha/2) - \hat{\lambda}) = 1 - \alpha
\]

In addition, we can argue that we can estimate the distribution of \( \hat{\lambda} - \lambda \) by the distribution of \( \hat{\lambda}^{(b)} - \hat{\lambda} \). This makes sense if you think of the analogy that \( \hat{\lambda} \) arose from sampling from a distribution with parameter \( \lambda \), while \( \hat{\lambda}^{(b)} \) arose in exactly the same way from sampling from a distribution from parameter \( \hat{\lambda} \). This means we can say

\[
P((q^*(\alpha/2) - \hat{\lambda} \leq \hat{\lambda}^{(b)} - \hat{\lambda} \leq q^*(1 - \alpha/2) - \hat{\lambda}) = 1 - \alpha
\]

Now we can find our confidence interval for \( \lambda \) by rearranging terms to get

\[
P(2\hat{\lambda} - q^*(1 - \alpha/2) \leq \lambda \leq 2\hat{\lambda} - q^*(\alpha/2)) = 1 - \alpha.
\]

This means our bootstrap confidence interval for \( \lambda \) is \([2\hat{\lambda} - q^*(1 - \alpha/2), 2\hat{\lambda} - q^*(\alpha/2)]\).

Now some code demonstrating how to find a 95% confidence interval for \( \lambda \). Look at each line of code carefully to see what it is doing.

```r
# first, initialize a matrix that will receive the values of the
# estimate from each sample
boot.sampling.dist<-matrix(1,2000)

# Now create 2000 bootstrap samples and compute the value of the stat for each of them
# for (i in 1:2000){
#  boot.sampling.dist[i]<-1/mean(rexp(100,empirical.lambda))
# }
# look at the sampling distribution of the stat, according to parametric bootstrap:
windows()
hist(boot.sampling.dist,main="Estimate of sampling distribution of lambda",breaks=50)

# find the quantiles of this distribution
my.quantiles<-quantile(boot.sampling.dist,c(.025,0.975))

# calculate the bootstrap confidence interval boundaries
CIbootlow<-2*empirical.lambda-my.quantiles[2]
CIboothigh<-2*empirical.lambda-my.quantiles[1]
```
One more thing that the collection of bootstrap estimated parameters can be used for is to calculate an estimate of the standard error of $\hat{\lambda}$. The standard error of $\hat{\lambda}$ can be estimated by the sample standard error of the bootstrap parameters:

$$\text{boot.estimate.se} = \sqrt{\text{var(boot.sampling.dist)}}$$

## 2 Some Helpful Commands

### Working with Distributions

There are a set of four functions that are defined for almost any distribution that you will encounter on your homework. I will show you examples for the normal distribution, but analogous functions exist for other distributions as well, such as the exponential (used above), the binomial, etc. These four functions are `rnorm()`, `pnorm()`, `qnorm()` and `dnorm()`. Each of them takes a series of arguments.

- **`rnorm()`** is used to generate a set of random variables sampled from a normal distribution of your choice. You pass it three arguments, in this order: $n$, the number of observations you want it to generate; $\mu$, the mean of the distribution you want it to sample from; $\sigma^2$, the variance of the distribution you want it to sample from. [Note that the number of arguments may vary if you are working with a different distribution that has a different number of parameters. For example, we saw above that `rexp()` only takes two parameters.]

- **`pnorm()`** is used to give the cumulative density of the distribution you are working with. You pass it three arguments: $q$, the quantile below which you want the cumulative probability; and the parameters of the distribution you are working with. For example, $\text{pnorm}(0, 0, 1) = 0.5$, since half the area of the standard normal distribution is below 0.

- **`qnorm()`** is used to calculate a quantile, if you know the cumulative probability. You pass it three arguments: $p$, the cumulative probability; and the parameters of the distribution you are working with. For example, $\text{qnorm}(0.5, 3, 1) = 3$ since 3 is the 0.5th quantile of the normal distribution with mean 3 and variance 1.

- **`dnorm()`** is used to calculate the value of the density function at a given point. You pass it three arguments: $x$, the point at which you want to compute the density; and the parameters of the distribution you are working with.

### Histograms

There are a couple of useful arguments for the `hist()` function that will come in handy on your homework:

- **`breaks`**: Tells R how many columns you want your histogram to have. If you give it a number, it will make that many columns. Otherwise, you can pass it a vector of breakpoints and the column borders will occur at the breakpoints you specify.

- **`freq`**: If this argument is set to `FALSE`, the histogram bars will indicate density (percentages) rather than counts. This is useful if you want to compare the distribution of your data to another distribution. You can superimpose the density of the distribution you want to compare your data to.

- **`main`**: This argument is where you set the title for your histogram. Name it something descriptive so whoever is looking at it knows what they’re seeing.
When you want to compare the shape of your histogram to the shape of a certain density function (like if you’re trying to test whether your assumed distribution was reasonable), you may want to superimpose the plot of the density function on top of your histogram. To do this, you first create your histogram using the `hist()` command. Then you call the function `lines()`, passing it two arguments: a vector of values ranging over the width of your histogram and the density of your desired distribution at each of these values. I’ll show an example of this below.

**An Example**

We will generate a random sample from a certain density, create a histogram of our sample, and superimpose the original density on top of our histogram to check the fit.

```r
# set number of sample points to generate
n<-1000

# generate n sample points from exponential dist with parameter 1
expo.data<-rexp(n,1)

# make a histogram of the data
hist(expo.data,breaks=50,main="Histogram of exponential data",freq=FALSE)

# observing that our histogram ranges over the values 0 to 10
# we create a vector that also ranges over these values
xvals<-seq(0,10,by=0.01)

# we then superimpose the density function at these points on our # histogram
lines(xvals,dexp(xvals,1))

# to save a copy of this histogram you can execute
pdf("myhist.pdf", width=3, height=3, horizontal=F)

hist(expo.data,breaks=50,main="Histogram of exponential data",freq=FALSE)
lines(xvals,dexp(xvals,1))
dev.off()
```
Figure 1: Histogram of exponential random variables with density superimposed.