The Bootstrap

Mark M. Fredrickson (mfredric@umich.edu)

Computational Methods in Statistics and Data Science (Stats 406)

Blood Pressure and Low Dose Aspirin

National Health And Nutrition Examination Survey

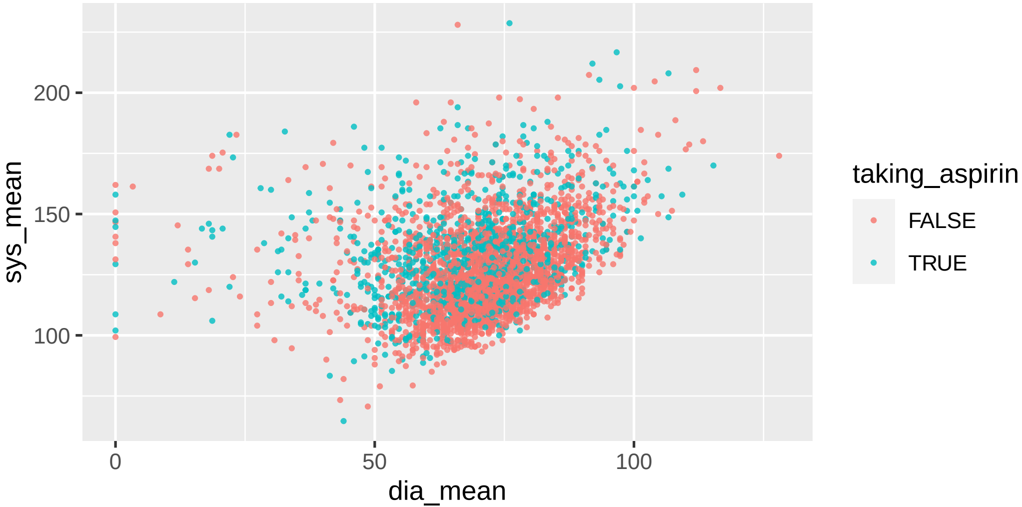
The **National Health And Nutrition Examination Survey (NHANES)** provides survey data on the dietary and health habits of people in the United States.

For many years, **low dose aspirin** was thought to be beneficial for those at risk of **heart disease**.

Variables:

- Respondent's self-reported use of low dose aspirin
- Diastolic and systolic blood pressure

See additional slides at end of lecture for data loading.



200
150
100
sys_mean
0
50
100
dia_mean
taking_aspirin
FALSE
TRUE

The Bootstrap

Set up

Suppose we have a sample of **size** n, which we will **assume are independent, identically distributed**.

$$X_i \sim F, \quad i = 1, \ldots, n, \quad \text{(independent)}$$

We want to estimate $\theta = \mathrm{E}\left(h(X)\right)$ using a **statistic** $T(X_1, \ldots, X_n) = T$ and construct a **confidence interval** for θ.

Usual Methods

We need to know the **sampling distribution of T** (the distribution of T if we picked many samples of size n).

- **Derive the distribution** of T from **assumptions** about the X_i.
- Find a **large sample approximation** for T and assume we have enough data for it to apply.
- **Fix a specific distribution** and use **Monte Carlo**.

Alternative: What if we could estimate the **distribution of X** and then use the **inversion method** to draw from our estimate?

Estimating F

Recall the definition of $F(x)$:

$$F(x) = \Pr(X \leq x) = \mathrm{E}\left(I(X \leq x)\right)$$

We developed tools to estimate things like $\mathrm{E}\left(I(X \leq x)\right)$:

$$\widehat{F}_n(x) = \frac{1}{n}\sum_{i=1}^{n} I(X_i \leq x)$$

This is the **empirical cumulative distribution function.**

Inversion method

Recall, the **inversion method** draws $U \sim U(0,1)$ and then generates $X = Q_X(U)$. Here, we will use the **empirical quantile function**.

Notice that $\widehat{F}_n(x)$ defines a **discrete distribution**, so the **empirical quantile function** is:

$$\hat{Q}_n(u) = x \text{ such that } \hat{F}_n(x - \epsilon) < u \leq \hat{F}_n(x), \forall \epsilon > 0$$

Notice: the only possible x values will be those that appear in the sample.

Claim: this is equivalent to picking an observation **uniformly at random**:

$$P(\hat{Q}(U) = X_i) = \frac{1}{n}, \quad \forall i$$

Proof

Breaking ties arbitrarily, relabel our data such that

$$X_{(1)} \leq X_{(2)} \leq \cdots \leq X_{(n)}$$

Observe: $\sum I(X_i \leq X_{(j)}) = j$

$$\begin{aligned}
P(\hat{Q}(U) = X_{(j)}) &= P\left(\hat{F}(X_{(j-1)}) < U \leq \hat{F}(X_{(j)})\right) \\
&= P\left(\frac{1}{n}\sum_{i=1}^{n} I(X_i \leq X_{(j-1)}) < U \leq \frac{1}{n}\sum_{i=1}^{n} I(X_i \leq X_{(j)})\right) \\
&= P((j-1)/n < U \leq j/n) \\
&= P(U \leq j/n) - P(U \leq (j-1)/n) \\
&= \frac{j}{n} - \frac{j-1}{n} = \frac{1}{n}
\end{aligned}$$

This process is equivalent to picking one of the X_i **uniformly at random**.

Key idea: estimate the **sampling distribution of T** by drawing many samples from the **estimated distribution for X**.

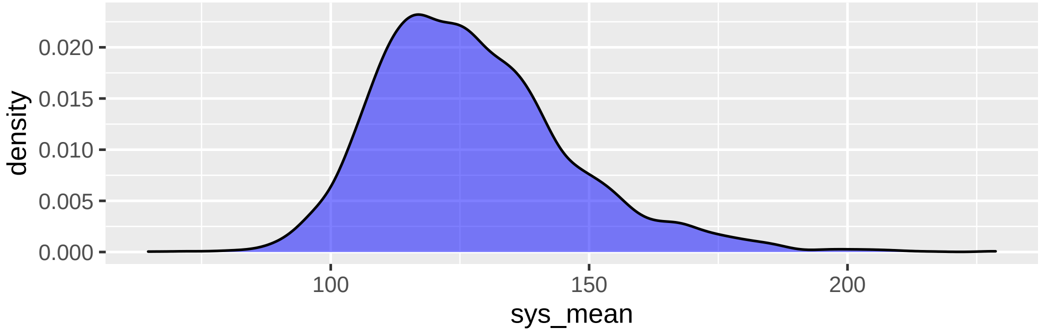
There are n^n possible samples, so we can't enumerate them all. So we will take a sample (**the bootstrap sample**).

Let $X_1^*, X_2^*, \ldots, X_n^*$ be a sample picked from the original n observations, taken with replacement. We will do this B times.

For each sample, compute $T(X_1^*, X_2^*, \ldots, X_n^*) = T^*$.

NHANES systolic measurements

Here's the distribution of systolic blood pressure (average) readings for the NHANES survey:



Let's estimate the **population mean of systolic blood pressure** (assuming NHANES is IID from US pop).

The trimmed mean

The previous distribution displayed a mild amount of **right-skew**.

While the CLT holds for large enough n, the **trimmed mean** might be a better choice.

Recall, $p \in [0, 1]$:

- Discard the lower $p/2$ and upper $p/2$ proportion of the data
- Compute the mean on the remaining observations

What is the **sampling distribution** of the trimmed mean for our data?

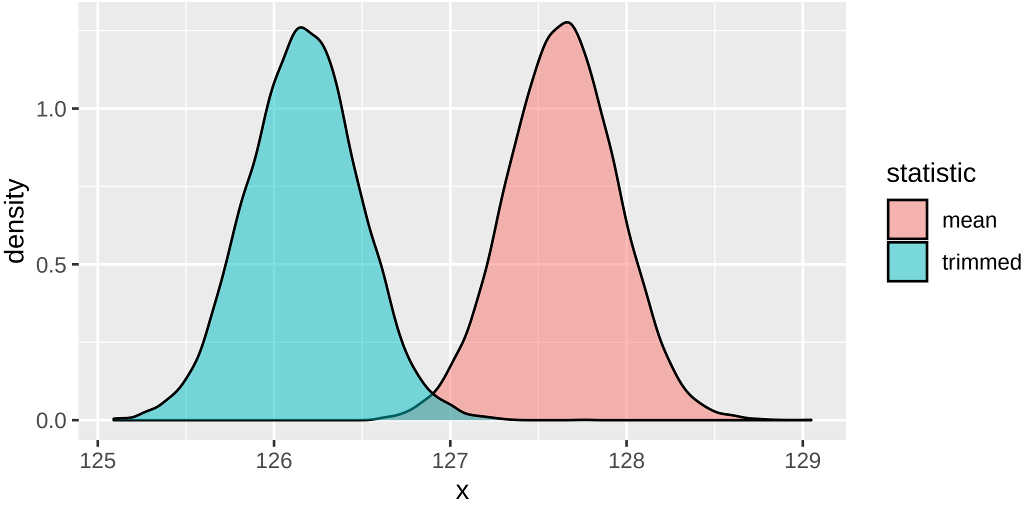
Trimmed Mean

```
> trimmed_mean <- function(x, p) {
+     xqs <- quantile(x, c(p/2, 1 - (p/2)))
+     keep <- x > xqs[1] & x < xqs[2]
+     mean(x[keep])
+ }
> (observed_mean <- mean(sys_mean))

[1] 127.6

> (observed_trim <- trimmed_mean(sys_mean, p = 0.2))

[1] 126.2
```

Bootstrap Sample

```
> B <- 10000
> n <- length(sys_mean)
> bootstrap_samples <- rerun(B,
+     sample(sys_mean, size = n, replace = TRUE))
> bootstrap_means <- map_dbl(bootstrap_samples, mean)
> bootstrap_trims <- map_dbl(bootstrap_samples, trimmed_mean, p = 0.2)
```

Comparing Sampling Distributions



Using the distribution

We are **estimating the sampling distribution** for these statistics. What information might we want to know about?

- Variance
- Quantiles
- Bias (would need to know true parameter) and MSE

We'll use the first two in order to **develop confidence intervals** and return to bias and MSE later.

Bootstrap Confidence Intervals

Our goal is to estimate some θ using $\hat{\theta} = T(X_1, X_2, \ldots, X_n)$ (we get this without the bootstrap).

We use the bootstrap to understand the **sampling distribution** of $\hat{\theta}$. We can use this to create **confidence intervals** in three basic ways:

- **Large sample approximation** intervals $\hat{\theta} \pm z_{\alpha/2}\hat{\sigma}_{\hat{\theta}}$
- Combining the **estimator on the original sample** with the **bootstrap distribution** in two different ways.

Large sample (Normal approximation)

Many statistics have the property that in **large samples**,

$$T \sim N(\theta, \sigma_T^2) \quad \text{(approx.)}$$

Using **test inversion** we could create a confidence interval as:

$$T \pm z_{\alpha/2}\sigma_T$$

Even if this is known, it may be difficult to compute or estimate σ_T. We substitute the **standard deviation of the bootstrap distribution**:

$$T \pm z_\alpha \sqrt{\frac{1}{B-1}\sum_{i=j}^{B}(T_j^* - \bar{T}^*)^2}$$

Bootstrap intervals vs. usual t-intervals

We derived the following interval:

$$T \pm z_\alpha \sqrt{\frac{1}{B-1} \sum_{i=j}^{B} (T_j^* - \bar{T}^*)^2}$$

Notice that this interval differs from the usual t based interval in three ways:

- We used a Normal quantile instead of a t-quantile. We could have used a t-quantile, but with large B it is basically equivalent.
- We used T instead of $\bar{T}^*$ as the center of the interval (observed estimator instead of mean of the bootstrap distribution).
- We are estimating the variance of T directly, so there is no $1/\sqrt{n}$ term (variance of sample mean is $\mathrm{Var}\left(\bar{X}\right) = (1/n)\mathrm{Var}\left(X\right)$).

Normal interval for systolic BP

The trimmed mean is a good example of a statistic that has an asymptotic Normal distribution (see Stigler (1973)) but a tricky variance.

```
> observed_trim + c(1, -1) * qnorm(0.025) * sd(bootstrap_trims)
```

```
[1] 125.6 126.9
```

Basic intervals

As before let $T = T(X_1, X_2, \ldots, X_n)$ (the **observed statistic**) and $T^* = T(X_1^*, X_2^*, \ldots, X_n^*)$ (the **bootstrap statistic**).

Suppose we want to create intervals of the form, $[T - a, T + b]$ so that

$$P(T - a < \theta < T + b) \geq 1 - \alpha$$

(i.e., $[T - a, T + b]$ is a valid CI).

Rewriting, we have the equivalent formulation:

$$P(-b \leq T - \theta \leq a) \geq 1 - \alpha$$

To pick $-b$ and a, we need the distribution of $T - \theta$.

Estimating $T - \theta$ using the bootstrap

The main goal of bootstrapping is **approximate the distribution of T using T^***. So that

$$P(-b \leq T - \theta \leq a) \approx P(-b \leq T^* - \theta \leq a)$$

We know the distribution of T^* (more properly, have an MC estimate of it), **what about θ**?

Provided T is a good estimator of θ (i.e., $T \xrightarrow{P} \theta$), then

$$P(-b \leq T^* - \theta \leq a) \approx P(-b \leq T^* - T \leq a)$$

This implies that we can pick $-b = T^*_{\alpha/2} - T$ and $a = T^*_{1-\alpha/2} - T$.

Putting it all together

We wanted CIs of the form:

$$P(\theta \in [T - a, T + b]) \geq 1 - \alpha$$

We found $a = T^*_{1-\alpha/2} - T$ and $b = T - T^*_{\alpha/2}$.

$$[T - a, T + b] = [2T - T^*_{1-\alpha/2}, 2T - T^*_{\alpha/2}]$$

We call these **basic bootstrap confidence intervals**.

Applying to previous examples

```
> ba_trim <- quantile(bootstrap_trims, c(0.975, 0.025))
> basic_trim <- 2 * observed_trim - ba_trim
> names(basic_trim) <- NULL # quantile adds some names, we don't need them
> basic_trim

[1] 125.7 126.9
```

Percentile Intervals

During the previous approach, we were looking for a and b such that

$$P(-b \leq T - \theta \leq a) \geq 1 - \alpha$$

and we had the approximation:

$$P(T^*_{\alpha/2} - T \leq T - \theta \leq T^*_{1-\alpha/2} - T) = P(T^*_{\alpha/2} \leq 2T - \theta \leq T^*_{1-\alpha/2})$$

Again, since $T \xrightarrow{P} \theta$, $2T \approx 2\theta$, so

$$P(T^*_{\alpha/2} \leq 2T - \theta \leq T^*_{1-\alpha/2}) \approx P(T^*_{\alpha/2} \leq 2\theta - \theta \leq T^*_{1-\alpha/2}) = P(T^*_{\alpha/2} \leq \theta \leq T^*_{1-\alpha/2})$$

which suggests intervals:

$$[T^*_{\alpha/2}, T^*_{1-\alpha/2}]$$

These **percentile intervals** also often form the basis for other refinements.

Trimmed mean example

```
> quantile(bootstrap_trims, c(0.025, 0.975))
 2.5% 97.5%
125.6 126.8
```

Looking across the three methods

We've see **three methods of creating confidence intervals**. Which one should you use?

- All are equally easy to compute.
- In practice, rarely much difference.
- If you are going to compute the full bootstrap, **why use the Normal approximation?**
- The basic and percentile intervals will always have **the same width**.
- All methods are **approximations** based on the assumptions that:
    - $T \approx \theta$ (n is large)
    - $\hat{F} \approx F$ (n is large)
    - $F^* \approx F$ (B is large)

Using R's `boot` package

Much of the code for running bootstrap estimates is repetitive. Let's use the `boot` package instead.

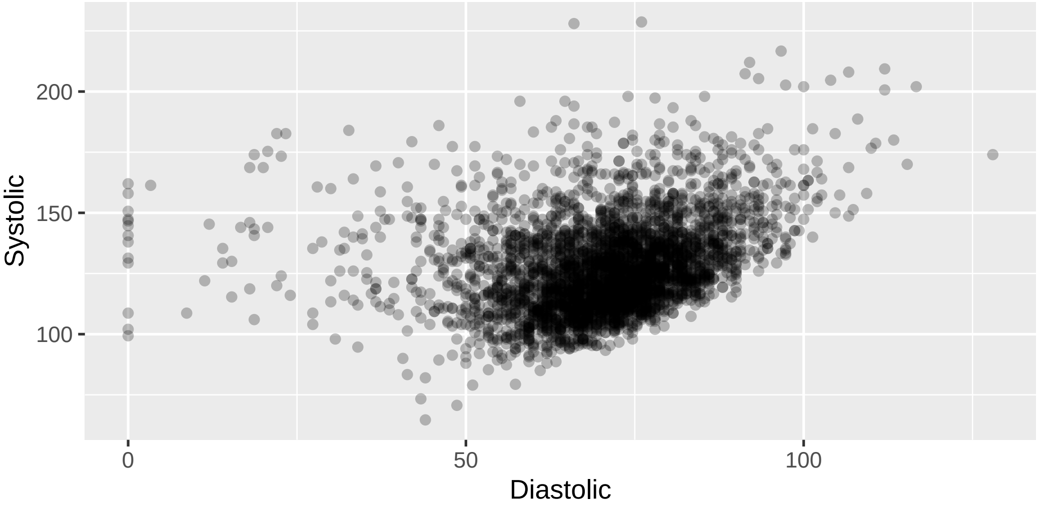
We need to set up our test statistic function to take a copy of the data and an index for the particular bootstrap sample.

```
> trimmed_mean_boot <- function(x, index, p = 0.1) {
+     trimmed_mean(x[index], p = p)
+ }
> library(boot)
> boot_tm <- boot(sys_mean, statistic = trimmed_mean_boot, p = 0.2, R = B)
```

```
> boot.ci(boot_tm, type = c("norm", "basic", "perc"))

BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 10000 bootstrap replicates

CALL :
boot.ci(boot.out = boot_tm, type = c("norm", "basic", "perc"))

Intervals :
Level      Normal              Basic              Percentile
95%   (125.7, 126.9 )   (125.7, 127.0 )   (125.5, 126.8 )
Calculations and Intervals on Original Scale
```

Estimating the correlation between systolic and diastolic BP

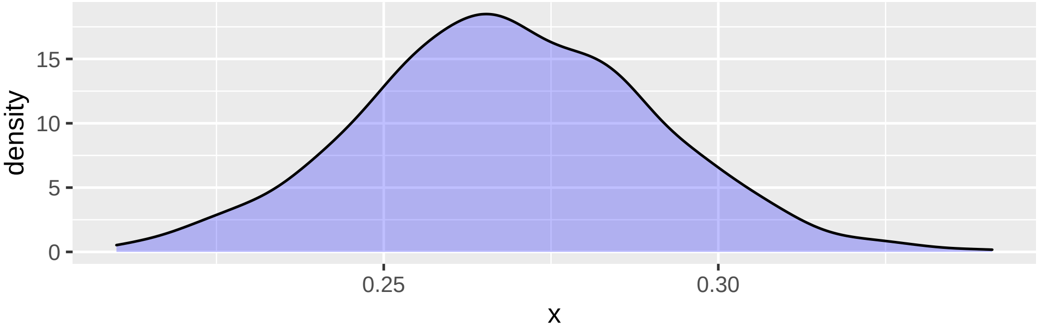
Research question: Are systolic and diastolic pressure linearly related?



Bootstrapping the Correlation Coefficient

```
> cor_boot_stat <- function(x, index) {
+     cor(x[index, 1], x[index, 2], use = "complete")
+ }
> library(boot)
> boot_cor <- boot(nhanes[, c("sys_mean", "dia_mean")],
+                  statistic = cor_boot_stat,
+                  R = 1000)
```

Correlation coefficient distribution



```
> boot.ci(boot_cor, type = c("norm", "basic", "perc"))

BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 1000 bootstrap replicates

CALL :
boot.ci(boot.out = boot_cor, type = c("norm", "basic", "perc"))

Intervals :
Level      Normal              Basic              Percentile
95%   ( 0.2269,  0.3117 )   ( 0.2266,  0.3118 )   ( 0.2262,  0.3114 )
Calculations and Intervals on Original Scale
```

Studentization

Suppose $X_1, \ldots, X_n \sim N(\mu, \sigma^2)$ (independent). Then

$$W = \frac{\bar{X} - \mu}{(S^2/n)^{1/2}}$$

has a Student's t-distribution with $n-1$ degrees of freedom.

More generally we say that a statistic is **studentized** if we subtract off a hypothesized location parameter and divide by an estimate of the standard deviation of the estimator.

Bootstrap-t (percentile) confidence intervals

We noted that **percentile** confidence intervals are frequently the basis of **improved confidence intervals**.

Define the "studentized" bootstrap replicate

$$W^* = \frac{T^* - T}{\hat{\sigma}^*}$$

Undo the studentization to get back to the T scale:

$$[T - \hat{\sigma} W^*_{1-\alpha/2}, T - \hat{\sigma} W^*_{\alpha/2}]$$

Variance Estimators

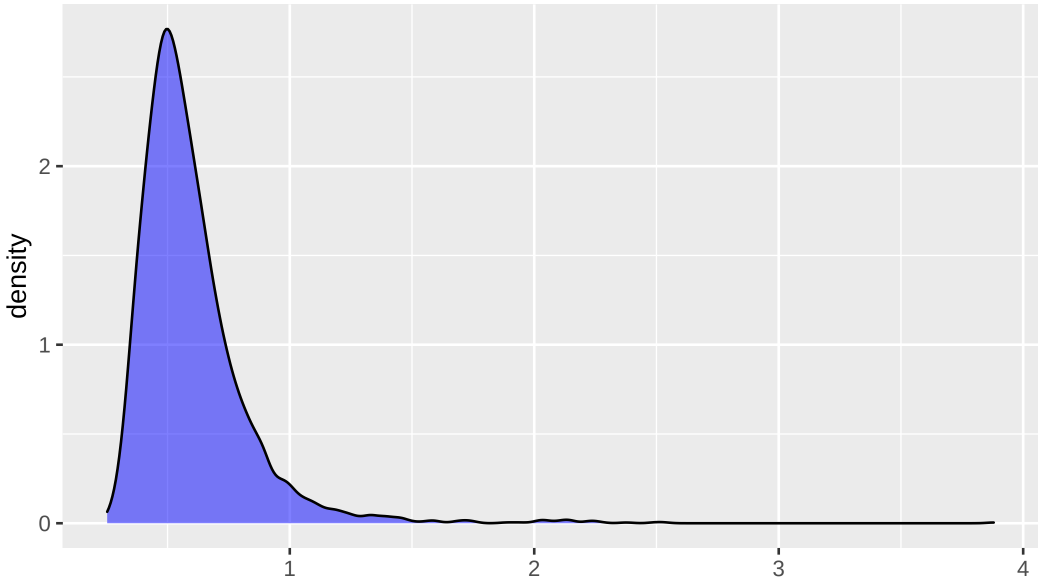
In the previous algorithm, we used **two different variance estimators** (for notational ease, I'm going to write these using standard deviations instead):

- $\hat{\sigma}^*$: estimates $\text{Var}(T^*)^{1/2}$ based on a particular bootstrap sample
- $\hat{\sigma}$: estimates $\text{Var}(T)^{1/2}$ based on the original sample

For either of these we could use

- Analytical estimator (e.g., $\text{Var}\left(\bar{X}\right) = (1/n)\text{Var}\left(X\right)$ and estimate $\text{Var}\left(X\right)$ using sample variance statistic S_x^2)
- Bootstrap estimate of variance ("nested bootstrap")
- The Jackknife (which we'll discuss a bit later)

Log Ratio of Systolic to Diastolic



Bootstrapping the mean

```
> library(boot)
> mean_boot <- function(x, index) { mean(x[index]) }
> boot_mean <- boot(log(sysdia_ratio), statistic = mean_boot, R = 1000)
```

```
> boot.ci(boot_mean, type = c("norm", "basic", "perc"))
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 1000 bootstrap replicates

CALL :
boot.ci(boot.out = boot_mean, type = c("norm", "basic", "perc"))

Intervals :
Level      Normal              Basic              Percentile
95%   ( 0.5931,  0.6086 )   ( 0.5929,  0.6085 )   ( 0.5935,  0.6090 )
Calculations and Intervals on Original Scale
```

Bootstrap-t: sample variance estimator

```
> B <- 1000
> lsdr <- log(sysdia_ratio)
> n <- length(lsdr)
> est_t <- mean(lsdr)
> est_var_t <- var(lsdr) / n
```

```
> boot_sv <- replicate(B, {
+     xstar <- sample(lsdr, replace = TRUE)
+     (mean(xstar) - est_t) / sqrt(var(xstar) / n)
+ })
> (boot_ci_svar <- est_t - sqrt(est_var_t) *
+     quantile(boot_sv, c(0.975, 0.025)))

    97.5%      2.5%
0.5932055 0.6088884
```

Bootstrap-t: Nested bootstrap

```
> boot_for_var <- replicate(100, {
+     xstar <- sample(lsdr, replace = TRUE)
+     mean(xstar)
+ })
> boot_var_est <- var(boot_for_var)
```

```
> boot_boot <- replicate(100, {
+     xstar <- sample(lsdr, replace = TRUE)
+     xstar_boot <- replicate(100, {
+         xstarstar <- sample(xstar, replace = TRUE)
+         mean(xstarstar)
+     })
+     (mean(xstar) - est_t) / sd(xstar_boot)
+ })
> (boot_ci_boot <- est_t - sqrt(boot_var_est) *
+      quantile(boot_boot, c(0.975, 0.025)))

    97.5%      2.5%
0.5942140 0.6081809
```

Using the `boot` package

If we return **two values**, the `boot` package will treat the first as T^* and the second as $\hat{\sigma}^2_*$.

```
> mean_var <- function(x, index) {
+   xstar <- x[index]
+   n <- length(xstar)
+   c(mean(xstar), var(xstar) / n)
+ }
> boot_both <- boot(lsdr, mean_var, R = 1000)
```

```
> boot.ci(boot_both, type = 'stud')

BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 1000 bootstrap replicates

CALL :
boot.ci(boot.out = boot_both, type = "stud")

Intervals :
Level    Studentized
95%   ( 0.5927,  0.6089 )
Calculations and Intervals on Original Scale
```

Comparing CIs

	Low	High	Rel. Width
Basic	0.592299625	0.608321349	1.000000000
Percentile	0.593614827	0.609636551	1.000000000
Studentized	0.592678187	0.608857458	1.009833294

Nested bootstrap of the median

With long tailed data (like the log-ratio were using), **the median** may be a better measure of **central tendency** than the mean.

We can't use sample variance estimate for the median, so we'll use **nested bootstrap**.

```
> median_idx <- function(data, idx) {
+   median(data[idx])
+ }
> median_nested <- function(data, idx) {
+     xstar <- data[idx]
+     meds <- boot(xstar, median_idx, R = 100)$t # just the T values
+     return(c(median(xstar), var(meds)))
+ }
```

Bootstrapping with Parallel Library

```
> library(parallel)
> cl <- makeCluster(detectCores())
> ## load the nested bootstrap components on the cluster
> ignore <- clusterEvalQ(cl, library(boot))
> clusterExport(cl, c("median_idx", "median_nested"))
> boot_median <- boot(lsdr, median_nested, R = 1000,
+   parallel = "snow", cl = cl, ncpus = detectCores())
> stopCluster(cl)
```

Confidence Intervals

```
> boot.ci(boot_median, type = "stud")

BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 1000 bootstrap replicates

CALL :
boot.ci(boot.out = boot_median, type = "stud")

Intervals :
Level    Studentized
95%   ( 0.5423,  0.5550 )
Calculations and Intervals on Original Scale
```

Assessing CI Coverage

Confidence Coefficient

For a **parameter** θ and the **random interval** $[A, B]$ we define the **confidence coefficient** c as

$$c = P(A \leq \theta, B \geq \theta)$$

If $c \geq 1 - \alpha$ then $[A, B]$ is a valid $(1 - \alpha) \times 100\%$ **Confidence Interval**.

It can be the case that a procedure we claim is a valid CI has $c < 1 - \alpha$. We need to evaluate our procedures to make sure this doesn't happen.

For

$$X \sim F, \mathrm{E}(X) = \mu, \mathrm{Var}(X) < \infty$$

the **central limit theorem** suggests that $\bar{X}$ is approximately

$$\bar{X} \sim N(\mu, \mathrm{Var}(X)/n)$$

This suggests that the usual t-intervals are **approximate confidence intervals**:

$$P(\bar{X} - t_{1-\alpha/2} s/\sqrt{n} \leq \mu \leq \bar{X} + t_{1-\alpha/2} s/\sqrt{n}) \approx 1 - \alpha$$

Quality of Approximation

As we saw in HW3, the **quality of this approximation depends on F**, the distribution of X.

We investigated the following when $\mu = 1/2$ and found

- $n = 20, X_i \sim$ Laplace$(1/2)$: good approximation
- $n = 20, X_i \sim$ Exp(2): poor approximation
- $n = 500, X_i \sim$ Exp(2): good approximation

Difficult problem: in general, is the Normal approximation valid?

Quick review of HW3 Results: Laplace

```
> k <- 10000
> laplace_intervals <- rerun(k, {
+   t.test(rlaplace(20, mean = 1/2), conf.level = 0.95)$conf.int
+ })
> a1 <- map_dbl(laplace_intervals, ~ .x[1] <= 1/2 & .x[2] >= 1/2)
> binom.test(sum(a1), k, conf.level = 0.99)$conf.int
```

```
[1] 0.9439078 0.9552592
attr(,"conf.level")
[1] 0.99
```

Quick review of HW3 Results: Exponential

```r
> exponential_intervals <- rerun(k, {
+   t.test(rexp(20, rate = 2), conf.level = 0.95)$conf.int
+ })
> a2 <- map_dbl(exponential_intervals, ~ .x[1] <= 1/2 & .x[2] >= 1/2)
> binom.test(sum(a2), k, conf.level = 0.99)$conf.int
```

```
[1] 0.9132797 0.9273164
attr(,"conf.level")
[1] 0.99
```

Evaluating Bootstrap Procedure

Goal: Do the bootstrap confidence interval procedures have proper confidence coefficients?

- Generate sample n from known distribution
- Perform bootstrap procedure B times
- Compute Normal theory, basic and percentile intervals
- Repeat k times

Note, requires about $n \times B \times k$ steps. Let's **parallelize!**

Setting up local cluster

```
> library(boot)
> library(parallel)
> cl <- makeCluster(detectCores() - 1)
> ## Example usage:
> mean_boot <- function(x, idx) { mean(x[idx]) }
> a <- boot(rlaplace(20, 1/2), mean_boot, R = 1000,
+           parallel = "snow",
+           cl = cl,
+           ncpus = 2) %>% # bug: must be greater than 1
+        boot.ci(type = c("norm", "basic", "perc"))
```

Formatted results

```
> print(a)

BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 1000 bootstrap replicates

CALL :
boot.ci(boot.out = ., type = c("norm", "basic", "perc"))

Intervals :
Level      Normal              Basic              Percentile
95%   ( 0.3568,  1.1857 )   ( 0.3439,  1.1986 )   ( 0.3674,  1.2221 )
Calculations and Intervals on Original Scale
```

What is the structure of `boot.ci` result?

```
> names(a)
[1] "R"        "t0"       "call"     "normal"  "basic"    "percent"
> a$normal ; a$basic ; a$percent
     conf
[1,] 0.95 0.3567675 1.185686
     conf
[1,] 0.95 975.98 25.03 0.3439484 1.198577
     conf
[1,] 0.95 25.03 975.98 0.367446 1.222074
```

Pulling out just the intervals

```
> getCIs <- function(boot_ci_result) {
+     with(boot_ci_result,
+          matrix(c(normal[2:3],
+                   basic[4:5],
+                   percent[4:5]), nrow = 2))
+ }
> getCIs(a)
```

```
          [,1]      [,2]     [,3]
[1,] 0.3567675 0.3439484 0.367446
[2,] 1.1856857 1.1985765 1.222074
```

Putting it together: Laplace

```
> k <- 1000 ; R <- 1000
> laplace_bootstrap_cis <- rerun(k, {
+    boot(rlaplace(20, 1/2), mean_boot, R = R,
+          parallel = "snow", cl = cl, ncpus = 2) %>%
+         boot.ci(type = c("norm", "basic", "perc")) %>%
+         getCIs
+ })
```

Putting it together: Exponential

```
> exp_bootstrap_cis <- rerun(k, {
+    boot(rexp(20, 2), mean_boot, R = R,
+         parallel = "snow", cl = cl, ncpus = 2) %>%
+        boot.ci(type = c("norm", "basic", "perc")) %>%
+        getCIs
+ })
```

Counting Covering CIs

```
> covers <- function(x) { x[1, ] <= 1/2 & x[2, ] >= 1/2 }
> laplace_bootstrap_covers <- map(laplace_bootstrap_cis, covers) %>%
+     simplify2array  %>% rowSums
> exp_bootstrap_covers <- map(exp_bootstrap_cis, covers) %>%
+     simplify2array %>% rowSums
```

Coverage Rates: Laplace

```
> coverage_ci <- function(x) { binom.test(x, k, conf.level = 0.99)$conf.int
> map(laplace_bootstrap_covers, coverage_ci)

[[1]]
[1] 0.9099948 0.9517603


[[2]]
[1] 0.9237124 0.9619124


[[3]]
[1] 0.8975841 0.9422873
```

Coverage Rates: Exponential

```
> map(exp_bootstrap_covers, coverage_ci)

[[1]]
[1] 0.8621267 0.9140733

[[2]]
[1] 0.8511989 0.9051019

[[3]]
[1] 0.8698147 0.9203143
```

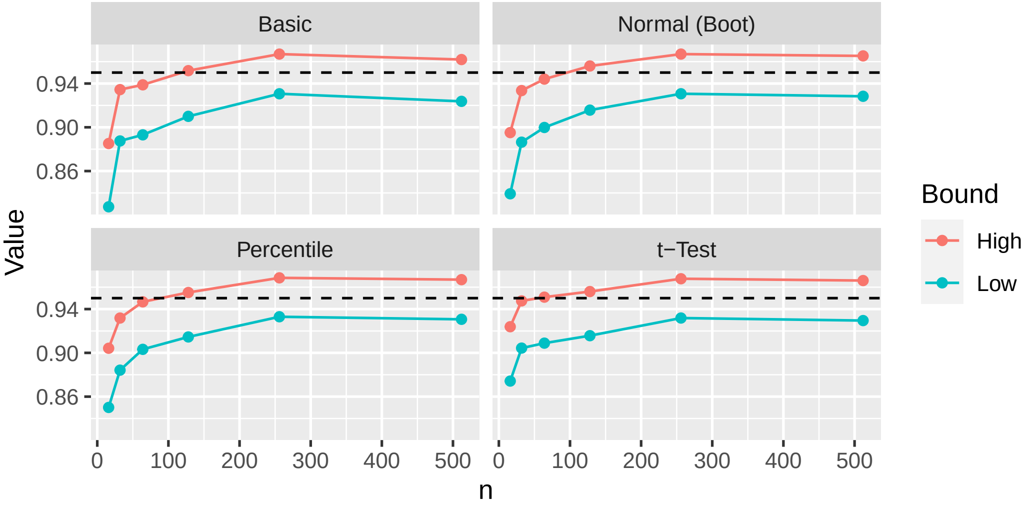
Why poor coverage rates?

Remember, we need $\hat{F}(x)$ **to be close to** $F(x)$ for all x.

We had a sample size of 20. Is this enough for a good approximation? (Apparently not.)

Let's repeat this procedure at **increasingly large sample sizes** (and compare to `t.test` along the way).

Plotting results



Interpreting Coverage Rates

In the previous examples, we saw that **all** of the methods had **confidence coefficients** below the targeted level.

For the t-test based intervals, we know this is because **the CLT Normal approximation only holds as** $n \to \infty$.

For the other methods, we only have that $\hat{F}(x) \xrightarrow{P} F$, another **law of large numbers** type result. The bootstrap requires the approximation $\hat{F}(x) \approx F$ to hold, which for small samples is tenuous.

Remember: the primary benefit of the bootstrap is **trading Monte Carlo methods for analytic methods** not magically making new data.

Summary

- Goal: **estimate** θ using an **estimator** $\hat{\theta} = T = T(X_1, X_2, \ldots, X_n)$.
- Need to know the **sampling distribution of** $\hat{\theta}$ to find **confidence intervals**.
- Main idea: If $X_i \overset{\text{iid}}{\sim} F$, **estimate** $\hat{F}$ in order to apply **inversion method**: $\hat{Q}(U) \sim F$ (approximately).
- The **bootstrap sample** of $X_1^*, X_2^*, \ldots, X_n^*$ is used to compute $T^* = T(X_1^*, \ldots, X_n*)$.
- Since $T^* \sim T$ (approx.) we can create confidence intervals:
  - **Large sample**: $T \pm z_{\alpha/2}\sigma_{T^*}$
  - **Basic**: $[2T - T^*_{1-\alpha/2}, 2T - T^*_{\alpha/2}]$
  - **Percentile**: $[T^*_{\alpha/2}, T^*_{1-\alpha/2}]$
  - **Studentized**: $[T - \sigma W^*_{1-\alpha/2}, T - \sigma W^*_{\alpha/2}]$, where $W^* = (T^* - T)/\hat{\sigma}^*$ (requires within bootstrap sample variance estimate)