Bootstrap Resampling

Nathaniel E. Helwig

Assistant Professor of Psychology and Statistics
University of Minnesota (Twin Cities)

Updated 04-Jan-2017

Copyright

Outline of Notes

1) Background Information
- Statistical inference
- Sampling distributions
- Need for resampling

2) Bootstrap Basics
- Overview
- Empirical distribution
- Plug-in principle

3) Bootstrap in Practice
- Bootstrap in R
- Bias and mean-squared error
- The Jackknife

4) Bootstrapping Regression
- Regression review
- Bootstrapping residuals
- Bootstrapping pairs

For a thorough treatment see:
Efron, B. & Tibshirani, R.J. (1993). *An Introduction to the Bootstrap*. Chapman & Hall/CRC.

Background Information

The Classic Statistical Paradigm

X is some random variable, e.g., age in years

$\mathcal{X} = \{x_1, x_2, x_3, \ldots\}$ is some population of interest, e.g.,

- Ages of all students at the University of Minnesota
- Ages of all people in the state of Minnesota

At the population level...

- $F(x) = P(X \leq x)$ for all $x \in \mathcal{X}$ is the population CDF
- $\theta = t(F)$ is population parameter, where t is some function of F

At the sample level...

- $\mathbf{x} = (x_1, \ldots, x_n)'$ is sample of data with $x_i \overset{\text{iid}}{\sim} F$ for $i \in \{1, \ldots, n\}$
- $\hat{\theta} = s(\mathbf{x})$ is sample statistic, where s is some function of $\mathbf{x}$

The Classic Statistical Paradigm (continued)

$\hat{\theta}$ is a random variable that depends on **x** (and thus F)

The sampling distribution of $\hat{\theta}$ refers to the CDF (or PDF) of $\hat{\theta}$.

If F is known (or assumed to be known), then the sampling distribution of $\hat{\theta}$ may have some known distribution.

- If $x_i \overset{\text{iid}}{\sim} \text{N}(\mu, \sigma^2)$, then $\bar{x} \sim \text{N}(\mu, \sigma^2/n)$ where $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$
- Note in the above example, $\theta \equiv \mu$ and $\hat{\theta} \equiv \bar{x}$

How can we make inferences about θ using $\hat{\theta}$ when F is unknown?

The Hypothetical Ideal

Assume that $\mathcal{X}$ is too large to measure all members of population.

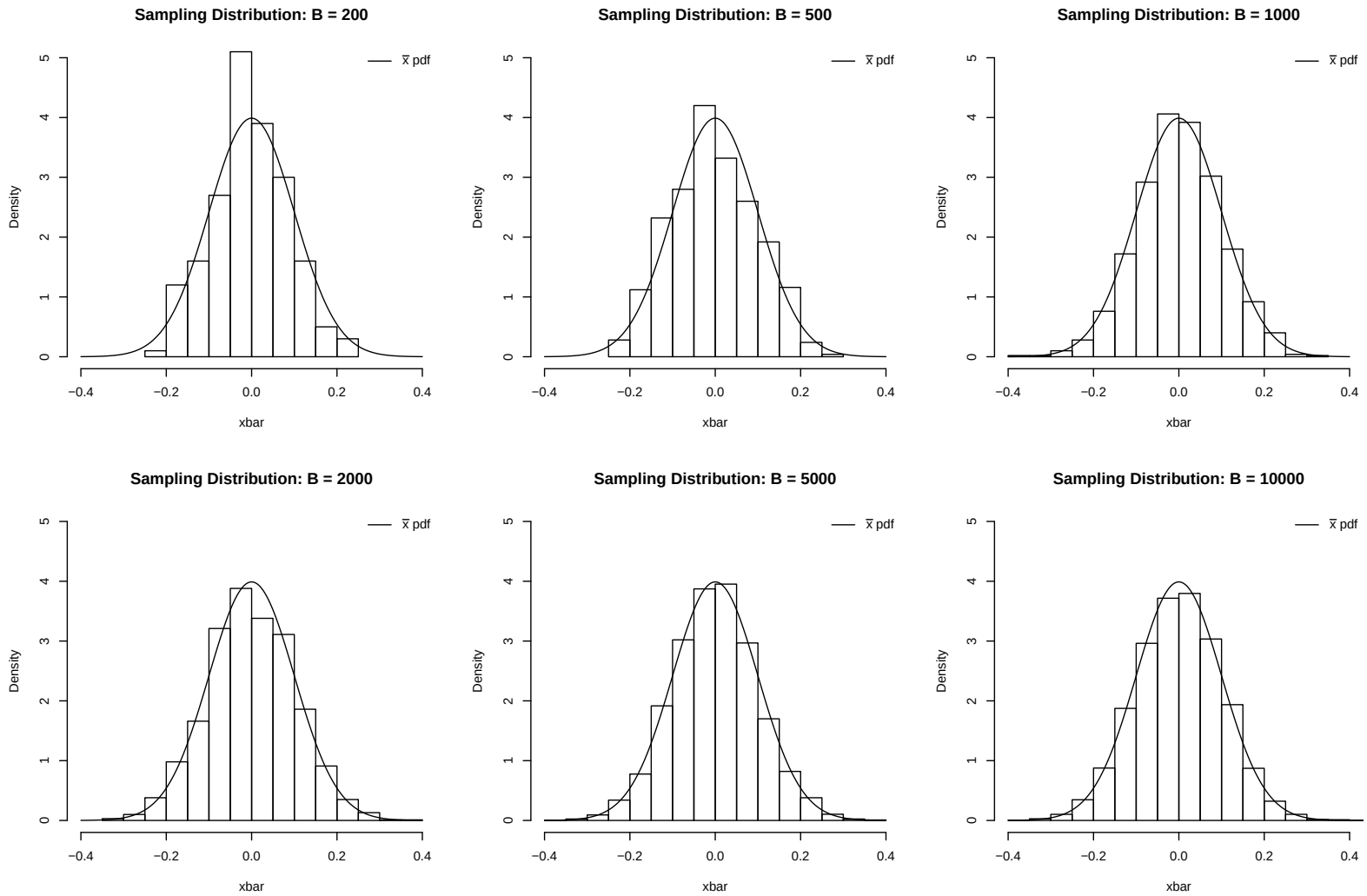
If we had a really LARGE research budget, we could collect B independent samples from the population $\mathcal{X}$

- $\mathbf{x}_j = (x_{1j}, \ldots, x_{nj})'$ is j-th sample with $x_{ij} \overset{\text{iid}}{\sim} F$
- $\hat{\theta}_j = s(\mathbf{x}_j)$ is statistic (parameter estimate) for j-th sample

Sampling distribution of $\hat{\theta}$ can be estimated via distribution of $\{\hat{\theta}_j\}_{j=1}^{B}$.

The Hypothetical Ideal: Example 1 (Normal Mean)

Sampling Distribution of $\bar{x}$ with $x_i \overset{\text{iid}}{\sim} \text{N}(0, 1)$ for $n = 100$:

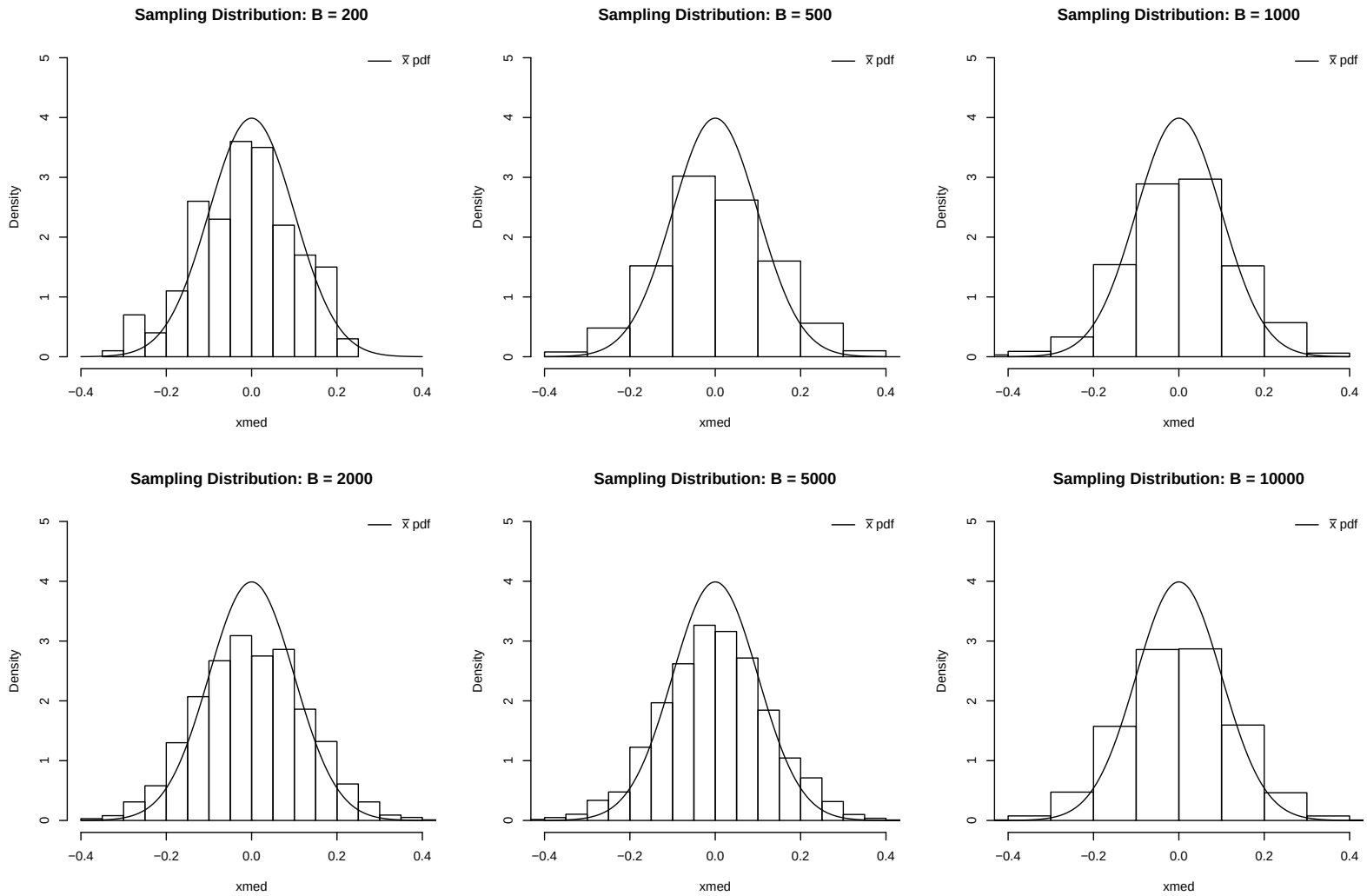


The Hypothetical Ideal: Example 1 R Code

```
# hypothetical ideal: example 1 (normal mean)
set.seed(1)
n = 100
B = c(200,500,1000,2000,5000,10000)
xseq = seq(-0.4,0.4,length=200)
quartz(width=12,height=8)
par(mfrow=c(2,3))
for(k in 1:6){
  X = replicate(B[k], rnorm(n))
  xbar = apply(X, 2, mean)
  hist(xbar, freq=F, xlim=c(-0.4,0.4), ylim=c(0,5),
       main=paste("Sampling Distribution: B =",B[k]))
  lines(xseq, dnorm(xseq, sd=1/sqrt(n)))
  legend("topright",expression(bar(x)*" pdf"),lty=1,bty="n")
}
```

The Hypothetical Ideal: Example 2 (Normal Median)

Sampling Distribution of median(x) with $x_i \overset{\text{iid}}{\sim} \text{N}(0, 1)$ for $n = 100$:



The Hypothetical Ideal: Example 2 R Code

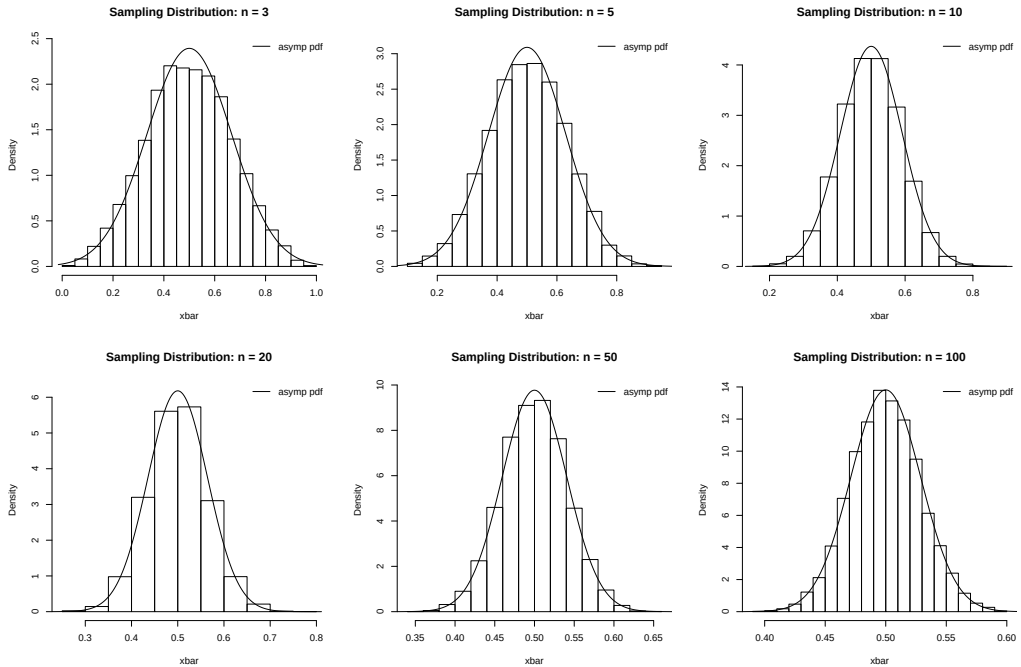
```
# hypothetical ideal: example 2 (normal median)
set.seed(1)
n = 100
B = c(200,500,1000,2000,5000,10000)
xseq = seq(-0.4,0.4,length=200)
quartz(width=12,height=8)
par(mfrow=c(2,3))
for(k in 1:6){
  X = replicate(B[k], rnorm(n))
  xmed = apply(X, 2, median)
  hist(xmed, freq=F, xlim=c(-0.4,0.4), ylim=c(0,5),
       main=paste("Sampling Distribution: B =",B[k]))
  lines(xseq, dnorm(xseq, sd=1/sqrt(n)))
  legend("topright",expression(bar(x)*" pdf"),lty=1,bty="n")
}
```

Back to the Real World

In most cases, we only have one sample of data. What do we do?

If n is large and we only care about $\bar{x}$, we can use the CLT.

Sampling Distribution of $\bar{x}$ with $x_i \overset{\text{iid}}{\sim} \text{U}[0, 1]$ for $B = 10000$:



The Need for a Nonparametric Resampling Method

For most statistics other than the sample mean, there is no theoretical argument to derive the sampling distribution.

To make inferences, we need to somehow obtain (or approximate) the sampling distribution of any generic statistic $\hat{\theta}$.

- Note that parametric statistics overcome this issue by assuming some particular distribution for the data
- Nonparametric bootstrap overcomes this problem by resampling observed data to approximate the sampling distribution of $\hat{\theta}$.

Bootstrap Basics

Problem of Interest

In statistics, we typically want to know the properties of our estimates, e.g., precision, accuracy, etc.

In parametric situation, we can often derive the distribution of our estimate given our assumptions about the data (or MLE principles).

In nonparametric situation, we can use the bootstrap to examine properties of our estimates in a variety of different situations.

Bootstrap Procedure

Suppose $\mathbf{x} = (x_1, \ldots, x_n)'$ with $x_i \overset{\text{iid}}{\sim} F(x)$ for $i \in \{1, \ldots, n\}$, and we want to make inferences about some statistic $\hat{\theta} = s(\mathbf{x})$.

Can use Monte Carlo Bootstrap:

1. Sample x_i^* with replacement from $\{x_1, \ldots, x_n\}$ for $i \in \{1, \ldots, n\}$
2. Calculate $\hat{\theta}^* = s(\mathbf{x}^*)$ for b-th sample where $\mathbf{x}^* = (x_1^*, \ldots, x_n^*)'$
3. Repeat 1–2 a total of B times to get bootstrap distribution of $\hat{\theta}$
4. Compare $\hat{\theta} = s(\mathbf{x})$ to bootstrap distribution

Estimated standard error of $\hat{\theta}$ is standard deviation of $\{\hat{\theta}_b^*\}_{b=1}^B$:

$$\hat{\sigma}_B = \sqrt{\frac{1}{B-1} \sum_{b=1}^B (\hat{\theta}_b^* - \bar{\theta}^*)^2}$$

where $\bar{\theta}^* = \frac{1}{B} \sum_{b=1}^B \hat{\theta}_b^*$ is the mean of the bootstrap distribution of $\hat{\theta}$.

Empirical Cumulative Distribution Functions

Suppose $\mathbf{x} = (x_1, \ldots, x_n)'$ with $x_i \overset{\text{iid}}{\sim} F(x)$ for $i \in \{1, \ldots, n\}$, and we want to estimate the cdf F.

The empirical cumulative distribution function (ecdf) $\hat{F}_n$ is defined as

$$\hat{F}_n(x) = \hat{P}(X \leq x) = \frac{1}{n} \sum_{i=1}^{n} I_{\{x_i \leq x\}}$$

where $I_{\{\cdot\}}$ denotes an indicator function.

The ecdf assigns probability $1/n$ to each value x_i, which implies that

$$\hat{P}_n(A) = \frac{1}{n} \sum_{i=1}^{n} I_{\{x_i \in A\}}$$

for any set A in the sample space of X.

Some Properties of ECDFs

For any fixed value x, we have that

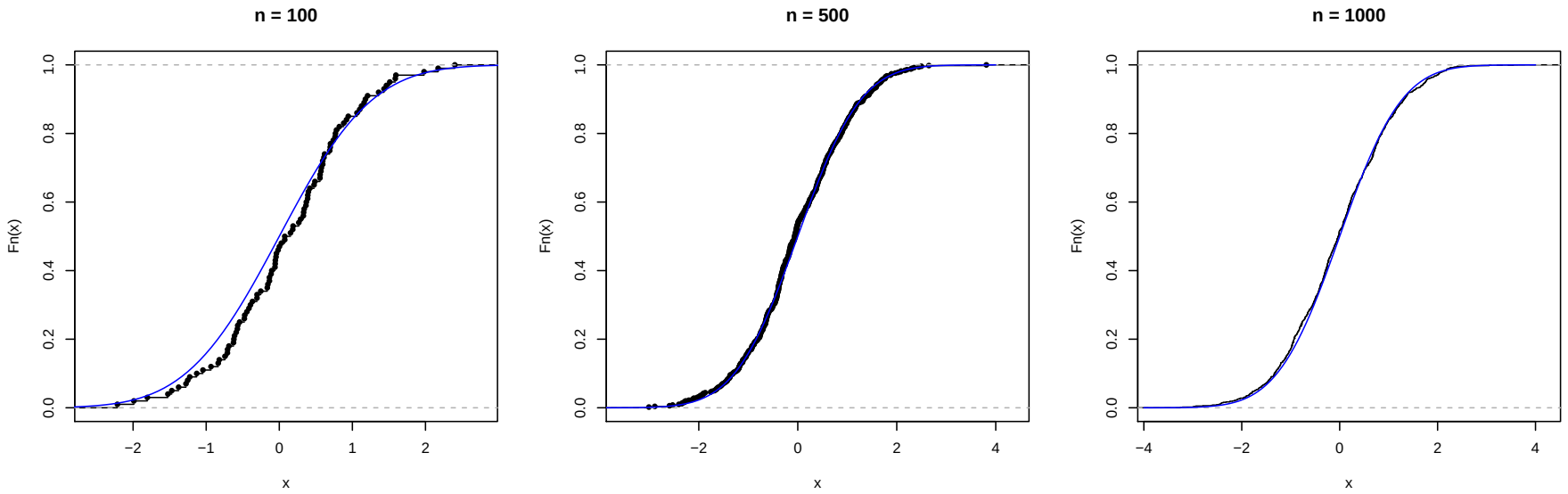
- $E[\hat{F}_n(x)] = F(x)$
- $V[\hat{F}_n(x)] = \frac{1}{n}F(x)[1 - F(x)]$

As $n \to \infty$ we have that

$$\sup_{x \in \mathbb{R}} |\hat{F}_n(x) - F(x)| \xrightarrow{as} 0$$

which is the Glivenko-Cantelli theorem.

ECDF Visualization for Normal Distribution




```
set.seed(1)
par(mfrow=c(1,3))
n = c(100,500,1000)
xseq = seq(-4,4,length=100)
for(j in 1:3){
  x = rnorm(n[j])
  plot(ecdf(x),main=paste("n = ",n[j]))
  lines(xseq,pnorm(xseq),col="blue")
}
```

ECDF Example

Table 3.1 *An Introduction to the Bootstrap* (Efron & Tibshirani, 1993).

School	LSAT (y)	GPA (z)	School	LSAT (y)	GPA (z)
1	576	3.39	9	651	3.36
2	635	3.30	10	605	3.13
3	558	2.81	11	653	3.12
4	578	3.03	12	575	2.74
5	666	3.44	13	545	2.76
6	580	3.07	14	572	2.88
7	555	3.00	15	594	2.96
8	661	3.43			

Defining $A = \{(y, z) : 0 < y < 600,\ 0 < z < 3.00\}$, we have

$$\hat{P}_{15}(A) = (1/15) \sum_{i=1}^{15} I_{\{(y_i, z_i) \in A\}} = 5/15$$

Plug-In Parameter Estimates

Suppose $\mathbf{x} = (x_1, \ldots, x_n)'$ with $x_i \overset{\text{iid}}{\sim} F(x)$ for $i \in \{1, \ldots, n\}$, and we want to estimate some parameter $\theta = t(F)$ that depends on the cdf F.

- Example: want to estimate expected value $E(X) = \int x f(x) \mathrm{d}x$

The plug-in estimate of $\theta = t(F)$ is given by

$$\hat{\theta} = t(\hat{F})$$

which is the statistic calculated using the ecdf in place of the cdf.

Plug-In Estimate of Mean

Suppose $\mathbf{x} = (x_1, \ldots, x_n)'$ with $x_i \overset{\text{iid}}{\sim} F(x)$ for $i \in \{1, \ldots, n\}$, and we want to estimate the expected value $\theta = E(X) = \int x f(x) \mathrm{d}x$.

The plug-in estimate of the expected value is the sample mean

$$\hat{\theta} = E_{\hat{F}}(x) = \sum_{i=1}^{n} x_i \hat{f}_i = \frac{1}{n} \sum_{i=1}^{n} x_i = \bar{x}$$

where $\hat{f}_i = \frac{1}{n}$ is sample probability from the ecdf.

Standard Error of Mean

Let $\mu_F = E_F(x)$ and $\sigma_F^2 = V_F(x) = E_F[(x - \mu_F)^2]$ denote the mean and variance of X, and denote this using the notation $X \sim (\mu_F, \sigma_F^2)$.

If $\mathbf{x} = (x_1, \ldots, x_n)'$ with $x_i \overset{\text{iid}}{\sim} F(x)$ for $i \in \{1, \ldots, n\}$, then the sample mean $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$ has mean and variance $\bar{x} \sim (\mu_F, \sigma_F^2/n)$.

The standard error of the mean $\bar{x}$ is the square root of the variance of $\bar{x}$

$$SE_F(\bar{x}) = \sigma_F/\sqrt{n}$$

Plug-In Estimate of Standard Error of Mean

Suppose $\mathbf{x} = (x_1, \ldots, x_n)'$ with $x_i \overset{\text{iid}}{\sim} F(x)$ for $i \in \{1, \ldots, n\}$, and we want to estimate the standard error of the mean $SE_F(\bar{x}) = \sigma_F/\sqrt{n} = \sqrt{E_F[(x - \mu_F)^2]/n}$.

The plug-in estimate of the standard deviation is given by

$$\hat{\sigma} = \sigma_{\hat{F}} = \left\{ \frac{1}{n} \sum_{i=1}^{n} (x_i - \bar{x})^2 \right\}^{1/2}$$

so the plug-in estimate of the standard error of the mean is

$$\hat{\sigma}/\sqrt{n} = \sigma_{\hat{F}}/\sqrt{n} = \left\{ \frac{1}{n^2} \sum_{i=1}^{n} (x_i - \bar{x})^2 \right\}^{1/2}$$

Bootstrap in Practice

Bootstrap Standard Error (revisited)

Suppose $\mathbf{x} = (x_1, \ldots, x_n)'$ with $x_i \overset{\text{iid}}{\sim} F(x)$ for $i \in \{1, \ldots, n\}$, and we want to make inferences about some statistic $\hat{\theta} = s(\mathbf{x})$.

Estimated standard error of $\hat{\theta}$ is standard deviation of $\{\hat{\theta}^*_b\}_{b=1}^B$:

$$\hat{\sigma}_B = \sqrt{\frac{1}{B-1} \sum_{b=1}^B (\hat{\theta}^*_b - \bar{\theta}^*)^2}$$

where $\bar{\theta}^* = \frac{1}{B} \sum_{b=1}^B \hat{\theta}^*_b$ is the mean of the bootstrap distribution of $\hat{\theta}$.

As the number of bootstrap samples goes to infinity, we have that

$$\lim_{B \to \infty} \hat{\sigma}_B = SE_{\hat{F}}(\hat{\theta})$$

where $SE_{\hat{F}}(\hat{\theta})$ is the plug-in estimate of $SE_F(\hat{\theta})$.

Illustration of Bootstrap Standard Error

Figure 6.1: *An Introduction to the Bootstrap* (Efron & Tibshirani, 1993).

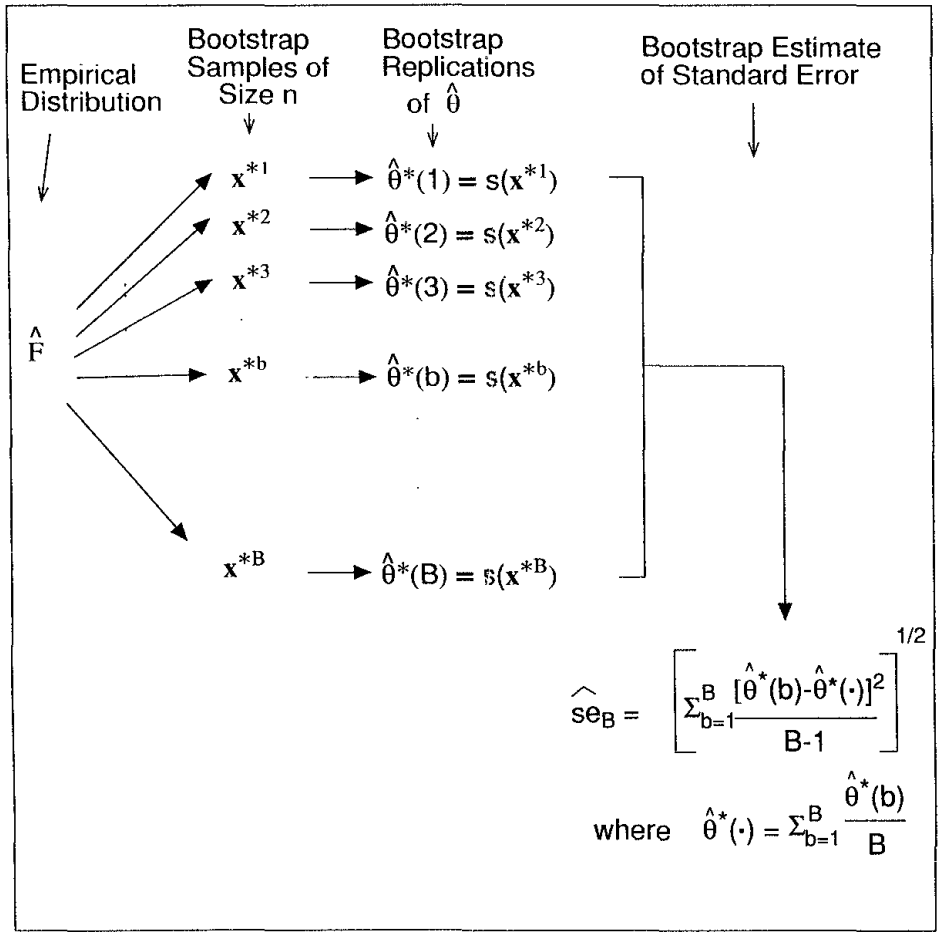
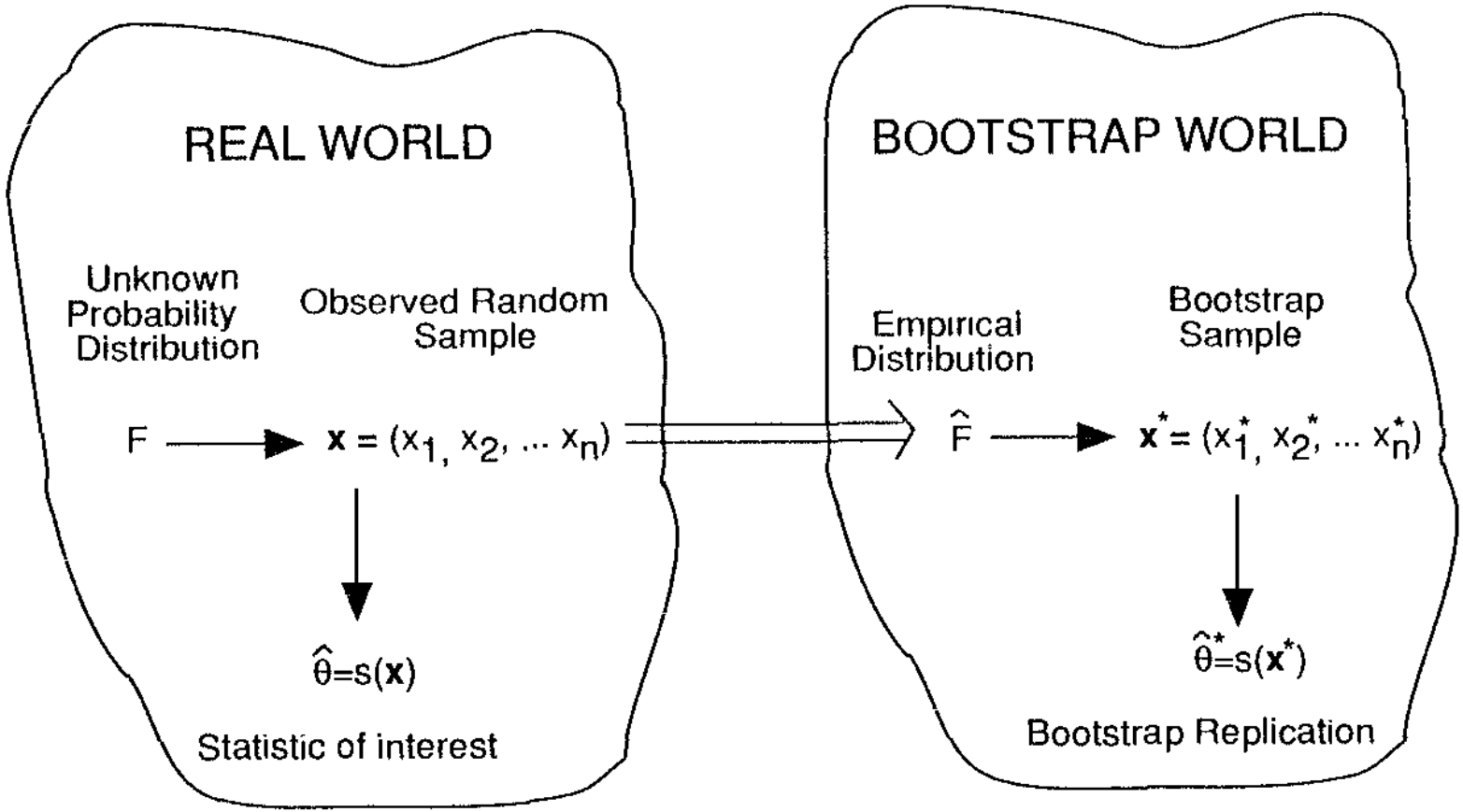


Illustration of Bootstrap Procedure

Figure 8.1: *An Introduction to the Bootstrap* (Efron & Tibshirani, 1993).



An R Function for Bootstrap Resampling

We can design our own bootstrap sampling function:

```
bootsamp <- function(x,nsamp=10000){
  x = as.matrix(x)
  nx = nrow(x)
  bsamp = replicate(nsamp,x[sample.int(nx,replace=TRUE),])
}
```

If `x` is a vector of length n, then `bootsamp` returns an $n \times B$ matrix, where B is the number of bootstrap samples (controlled via `nsamp`).

If `x` is a matrix of order $n \times p$, then `bootsamp` returns an $n \times p \times B$ array, where B is the number of bootstrap samples.

An R Function for Bootstrap Standard Error

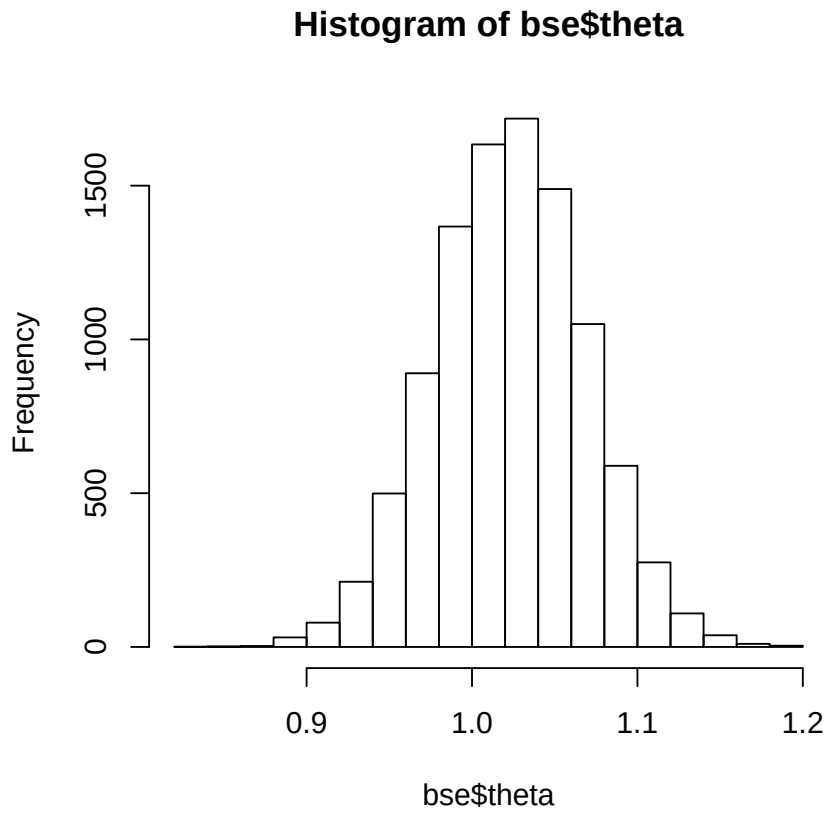
We can design our own bootstrap standard error function:

```
bootse <- function(bsamp,myfun,...){
  if(is.matrix(bsamp)){
    theta = apply(bsamp,2,myfun,...)
  } else {
    theta = apply(bsamp,3,myfun,...)
  }
  if(is.matrix(theta)){
    return(list(theta=theta,cov=cov(t(theta))))
  } else{
    return(list(theta=theta,se=sd(theta)))
  }
}
```

Returns a list where `theta` contains bootstrap statistic $\{\hat{\theta}^*_b\}_{b=1}^B$, and `se` contains the bootstrap standard error estimate (or `cov` contains bootstrap covariance matrix).

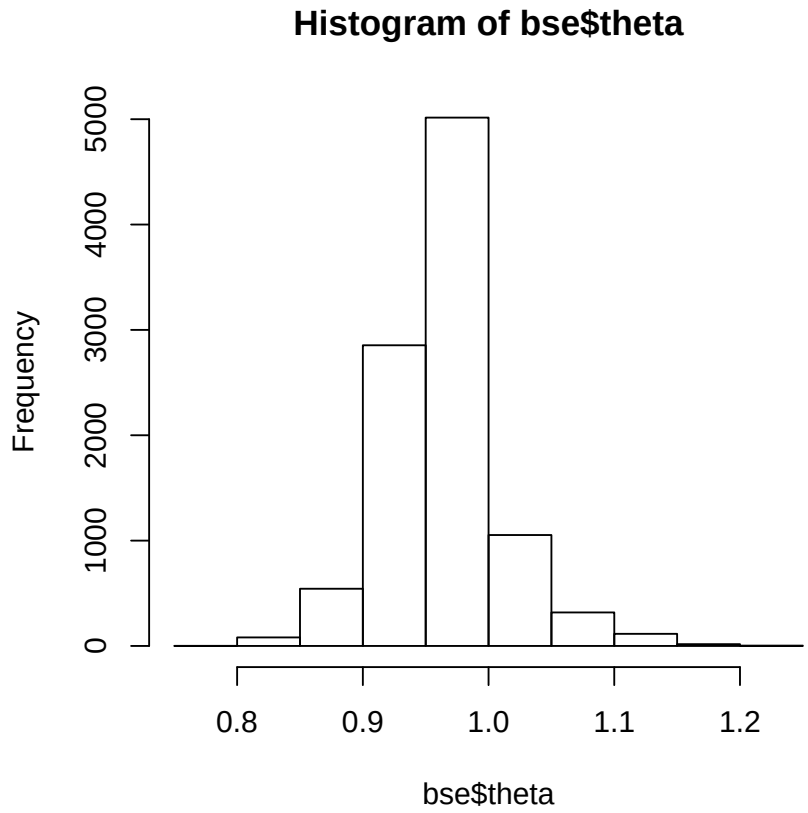
Example 1: Sample Mean

```
> set.seed(1)
> x = rnorm(500,mean=1)
> bsamp = bootsamp(x)
> bse = bootse(bsamp,mean)
> mean(x)
[1] 1.022644
> sd(x)/sqrt(500)
[1] 0.04525481
> bse$se
[1] 0.04530694
> hist(bse$theta)
```



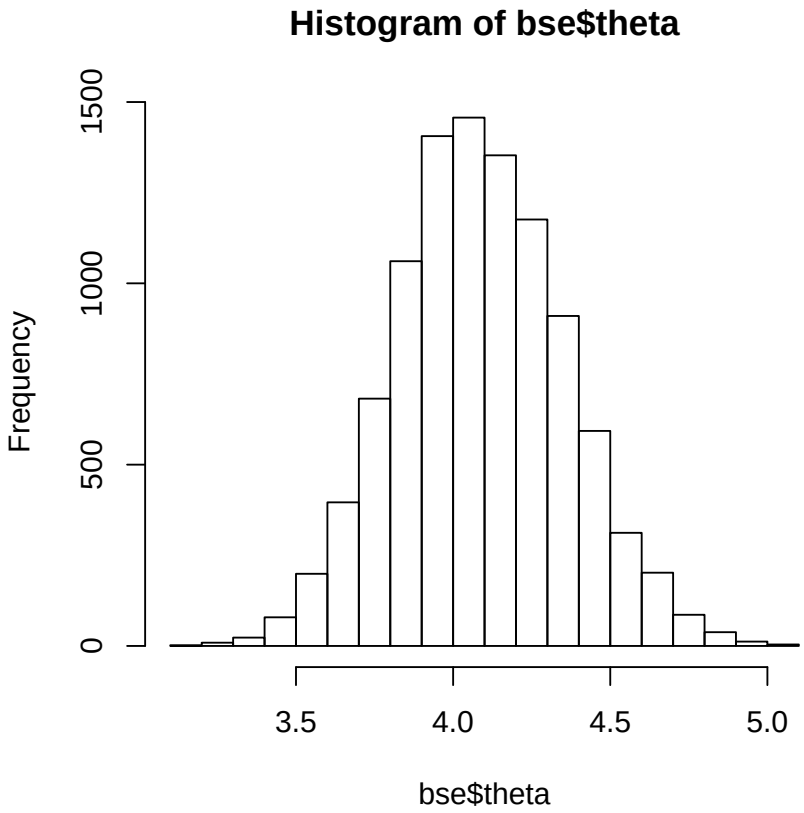
Example 2: Sample Median

```
> set.seed(1)
> x = rnorm(500,mean=1)
> bsamp = bootsamp(x)
> bse = bootse(bsamp,median)
> median(x)
[1] 0.9632217
> bse$se
[1] 0.04299574
> hist(bse$theta)
```



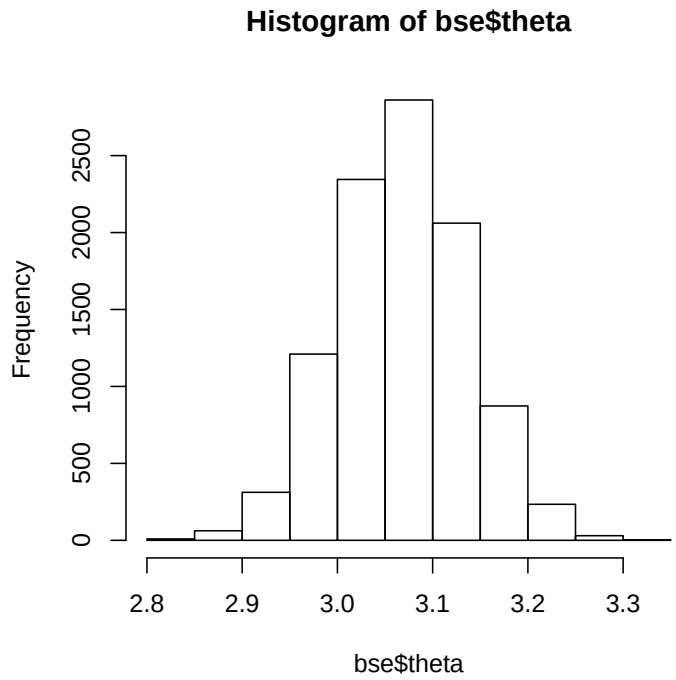
Example 3: Sample Variance

```
> set.seed(1)
> x = rnorm(500,sd=2)
> bsamp = bootsamp(x)
> bse = bootse(bsamp,var)
> var(x)
[1] 4.095996
> bse$se
[1] 0.2690615
> hist(bse$theta)
```



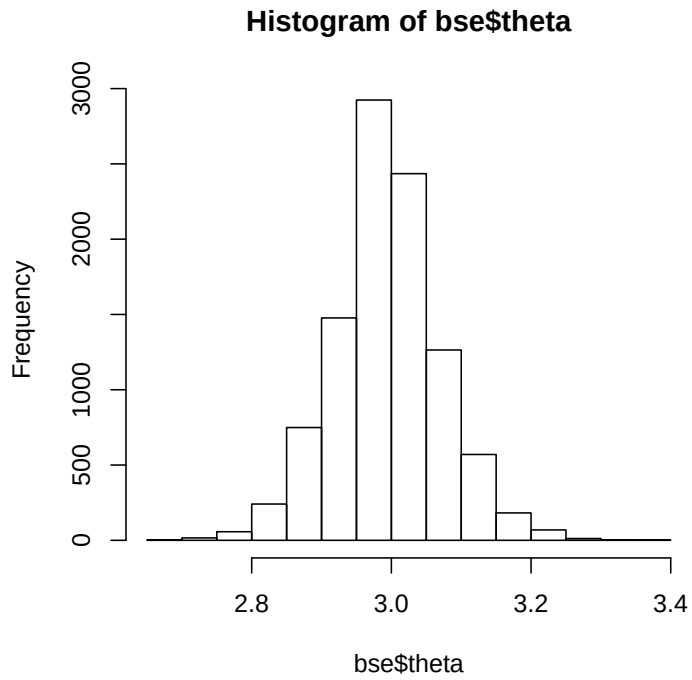
Example 4: Mean Difference

```
> set.seed(1)
> x = rnorm(500,mean=3)
> y = rnorm(500)
> z = cbind(x,y)
> bsamp = bootsamp(z)
> myfun = function(z) mean(z[,1]) - mean(z[,2])
> bse = bootse(bsamp,myfun)
> myfun(z)
[1] 3.068584
> sqrt( (var(z[,1])+var(z[,2]))/nrow(z) )
[1] 0.06545061
> bse$se
[1] 0.06765369
> hist(bse$theta)
```



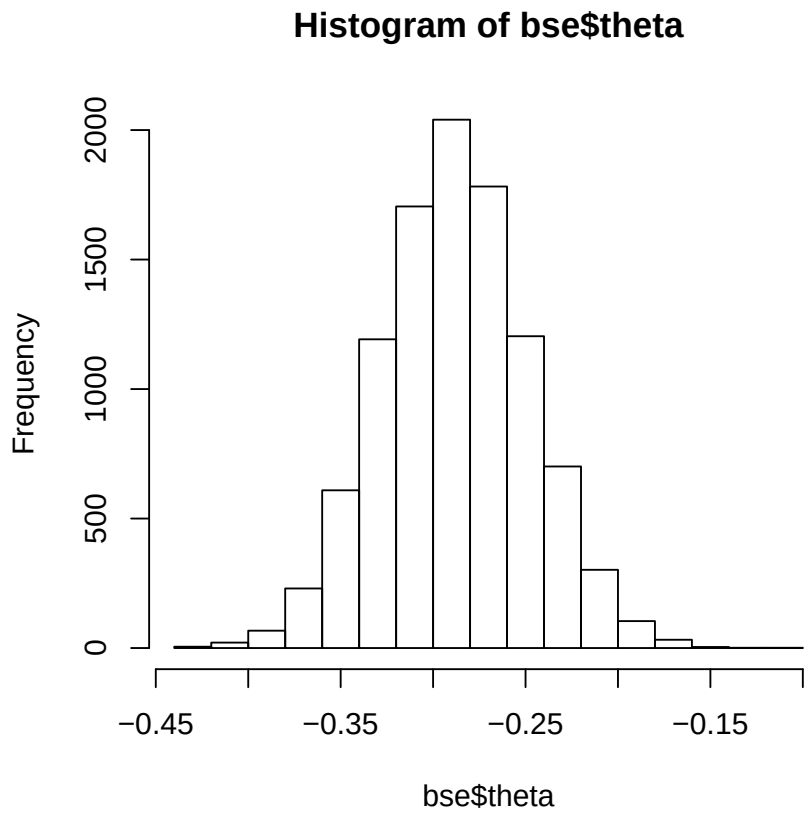
Example 5: Median Difference

```
> set.seed(1)
> x = rnorm(500,mean=3)
> y = rnorm(500)
> z = cbind(x,y)
> bsamp = bootsamp(z)
> myfun = function(z) median(z[,1]) - median(z[,2])
> bse = bootse(bsamp,myfun)
> myfun(z)
[1] 2.984479
> bse$se
[1] 0.07699423
> hist(bse$theta)
```



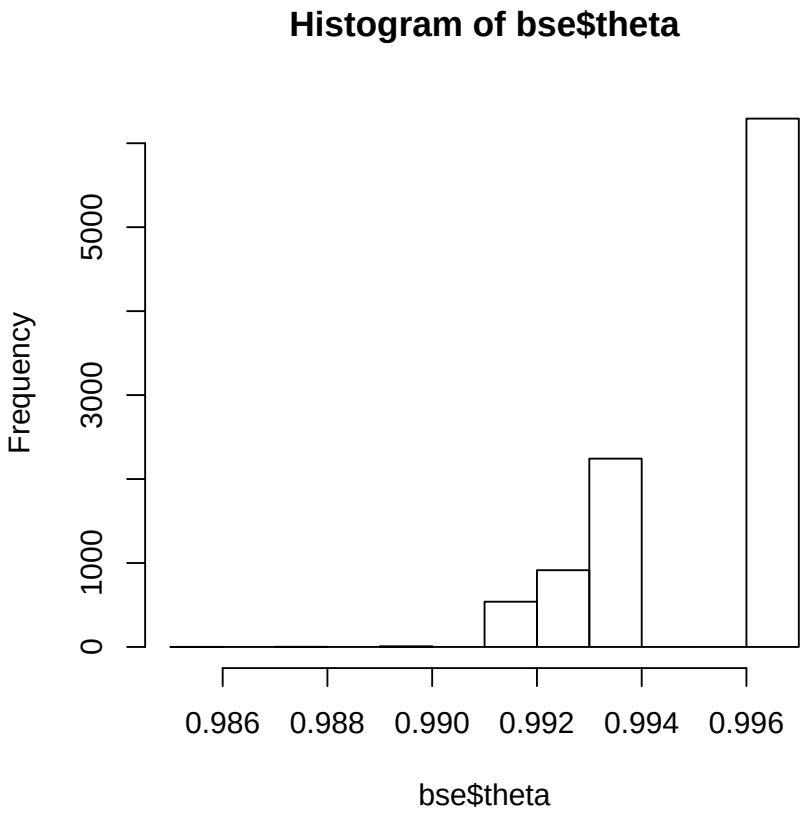
Example 6: Correlation Coefficient

```
> set.seed(1)
> x=rnorm(500)
> y=rnorm(500)
> Amat=matrix(c(1,-0.25,-0.25,1),2,2)
> Aeig=eigen(Amat,symmetric=TRUE)
> evec=Aeig$vec
> evalsqrt=diag(Aeig$val^0.5)
> Asqrt=evec%*%evalsqrt%*%t(evec)
> z=cbind(x,y)%*%Asqrt
> bsamp=bootsamp(z)
> myfun=function(z) cor(z[,1],z[,2])
> bse=bootse(bsamp,myfun)
> myfun(z)
[1] -0.2884766
> (1-myfun(z)^2)/sqrt(nrow(z)-3)
[1] 0.04112326
> bse$se
[1] 0.03959024
> hist(bse$theta)
```



Example 7: Uniform$[0, \theta]$ = bootstrap failure

```
> set.seed(1)
> x = runif(500)
> bsamp = bootsamp(x)
> myfun = function(x) max(x)
> bse = bootse(bsamp,myfun)
> myfun(x)
[1] 0.9960774
> bse$se
[1] 0.001472801
> hist(bse$theta)
```



$\hat{F}$ is not a good estimate of F in extreme tails.

Measuring the Quality of an Estimator

We have focused on standard error to measure the precision of $\hat{\theta}$.

- Small standard error is good, but other qualities are important too!

Consider the toy example:

- Suppose $\{x_i\}_{i=1}^n \overset{\text{iid}}{\sim} (\mu, \sigma^2)$ and want to estimate mean of X
- Define $\hat{\mu} = 10 + \bar{x}$ to be our estimate of μ
- Standard error of $\hat{\mu}$ is $\sigma/\sqrt{n}$ and $\lim_{n\to\infty} \sigma/\sqrt{n} = 0$
- But $\hat{\mu} = 10 + \bar{x}$ is clearly not an ideal estimate of μ

Bias of an Estimator

Suppose $\mathbf{x} = (x_1, \ldots, x_n)'$ with $x_i \overset{\text{iid}}{\sim} F(x)$ for $i \in \{1, \ldots, n\}$, and we want to make inferences about some statistic $\hat{\theta} = s(\mathbf{x})$.

The bias of an estimate $\hat{\theta} = s(\mathbf{x})$ of $\theta = t(F)$ is defined as

$$\text{Bias}_F = E_F[s(\mathbf{x})] - t(F)$$

where the expectation is taken with respect to F.

- Bias is difference between expectation of estimate and parameter
- In the example on the previous slide, we have that $\text{Bias}_F = E_F(\hat{\mu}) - \mu = E_F(10 + \bar{x}) - \mu = 10 + E_F(\bar{x}) - \mu = 10$

Bootstrap Estimate of Bias

The bootstrap estimate of bias substitutes $\hat{F}$ for F in bias definition

$$\text{Bias}_{\hat{F}} = E_{\hat{F}}[s(\mathbf{x})] - t(\hat{F})$$

where the expectation is taken with respect to the ecdf $\hat{F}$.

- Note that $t(\hat{F})$ is the plug-in estimate of θ
- $t(\hat{F})$ is not necessarily equal to $\hat{\theta} = s(\mathbf{x})$

Given B bootstrap samples, we can estimate bias using

$$\widehat{\text{Bias}}_B = \bar{\theta}^* - t(\hat{F})$$

where $\bar{\theta}^* = \frac{1}{B}\sum_{b=1}^{B} \hat{\theta}_b^*$ is the mean of the bootstrap distribution of $\hat{\theta}$

An R Function for Bootstrap Bias

We can design our own bootstrap bias estimation function:

```
bootbias <- function(bse,theta,...){
  if(is.matrix(bse$theta)){
    return(apply(bse$theta,1,mean) - theta)
  } else{
    return(mean(bse$theta) - theta)
  }
}
```

The first input `bse` is the object output from `bootse`, and the second input `theta` is the plug-in estimate of theta used for bias calculation.

Sample Mean is Unbiased

```
> set.seed(1)
> x = rnorm(500,mean=1)
> bsamp = bootsamp(x)
> bse = bootse(bsamp,mean)
> mybias = bootbias(bse,mean(x))
> mybias
[1] 0.0003689287
> mean(x)
[1] 1.022644
> sd(x)/sqrt(500)
[1] 0.04525481
> bse$se
[1] 0.04530694
```

Toy Example

```
> set.seed(1)
> x = rnorm(500,mean=1)
> bsamp = bootsamp(x)
> bse = bootse(bsamp,function(x) mean(x)+10)
> mybias = bootbias(bse,mean(x))
> mybias
[1] 10.00037
> mean(x)
[1] 1.022644
> sd(x)/sqrt(500)
[1] 0.04525481
> bse$se
[1] 0.04530694
```

Mean Squared Error (MSE)

The mean-squared error (MSE) of an estimate $\hat{\theta} = s(\mathbf{x})$ of $\theta = t(F)$ is

$$\begin{aligned} \text{MSE}_F &= E_F\{[s(\mathbf{x}) - t(F)]^2\} \\ &= V_F(\hat{\theta}) + [\text{Bias}_F(\hat{\theta})]^2 \end{aligned}$$

where the expectation is taken with respect to F.

- MSE is expected squared difference between $\hat{\theta}$ and θ
- In the toy example on the previous slide, we have that

$$\begin{aligned} \text{MSE}_F &= E_F\{(\hat{\mu} - \mu)^2\} \\ &= E_F\{(10 + \bar{x} - \mu)^2\} \\ &= E_F(10^2) + 2E_F[10(\bar{x} - \mu)] + E_F[(\bar{x} - \mu)^2] \\ &= 100 + \sigma^2/n \end{aligned}$$

Toy Example (revisited)

```
> set.seed(1)
> x = rnorm(500,mean=1)
> bsamp = bootsamp(x)
> bse = bootse(bsamp,function(x) mean(x)+10)
> mybias = bootbias(bse,mean(x))
> c(bse$se,mybias)
[1]  0.04530694 10.00036893
> c(bse$se,mybias)^2
[1] 2.052718e-03 1.000074e+02
> mse = (bse$se^2) + (mybias^2)
> mse
[1] 100.0094
> 100 + 1/length(x)
[1] 100.002
```

Balance between Accuracy and Precision

MSE quantifies both the accuracy (bias) and precision (variance).

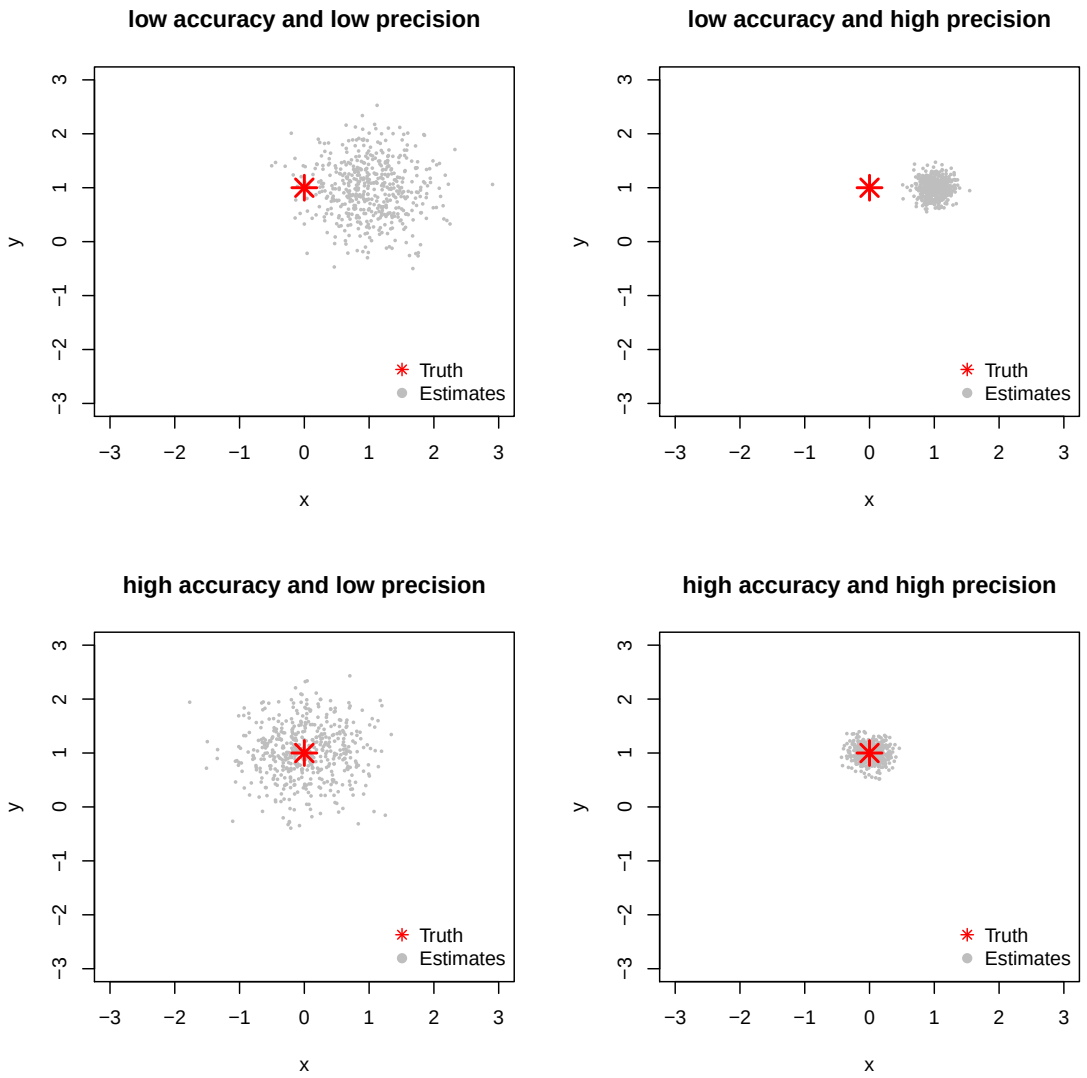
Ideal estimators are accurate (small bias) and precise (small variance).

Having some bias can be an ok (or even good) thing, despite the negative connotations of the word “biased”.

For example:

- Q: Would you rather have an estimator that is biased by 1 unit with a standard error of 1 unit? Or one that is unbiased but has a standard error of 1.5 units?
- A: The first estimator is better with respect to MSE

Accuracy and Precision Visualization



Jackknife Sample

Before the bootstrap, the jackknife was used to estimate bias and SE.

The i-th jackknife sample of $\mathbf{x} = (x_1, \ldots, x_n)$ is defined as

$$\mathbf{x}_{(i)} = (x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n)$$

for $i \in \{1, \ldots, n\}$. Note that...

- $\mathbf{x}_{(i)}$ is original data vector without the i-th observation
- $\mathbf{x}_{(i)}$ is a vector of length $(n-1)$

Jackknife Replication

The i-th jackknife replication $\hat{\theta}_{(i)}$ of the statistic $\hat{\theta} = s(\mathbf{x})$ is

$$\hat{\theta}_{(i)} = s(\mathbf{x}_{(i)})$$

which is the statistic calculated using the i-th jackknife sample.

For plug-in statistics $\hat{\theta} = t(\hat{F})$, we have that

$$\hat{\theta}_{(i)} = t(\hat{F}_{(i)})$$

where $\hat{F}_{(i)}$ is the empirical distribution of $\mathbf{x}_{(i)}$.

Jackknife Estimate of Standard Error

The jackknife estimate of standard error is defined as

$$\hat{\sigma}_{\text{jack}} = \sqrt{\frac{n-1}{n}\sum_{i=1}^{n}(\hat{\theta}_{(i)} - \hat{\theta}_{(\cdot)})^2}$$

where $\hat{\theta}_{(\cdot)} = \frac{1}{n}\sum_{i=1}^{n}\hat{\theta}_{(i)}$ is the mean of the jackknife estimates of θ.

Note that the $\frac{n-1}{n}$ factor is derived considering the special case $\hat{\theta} = \bar{x}$

$$\hat{\sigma}_{\text{jack}} = \sqrt{\frac{1}{(n-1)n}\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

which is an unbiased estimator of the standard error of $\bar{x}$.

Jackknife Estimate of Bias

The jackknife estimate of bias is defined as

$$\widehat{\text{Bias}}_{\text{jack}} = (n-1)(\hat{\theta}_{(\cdot)} - \hat{\theta})$$

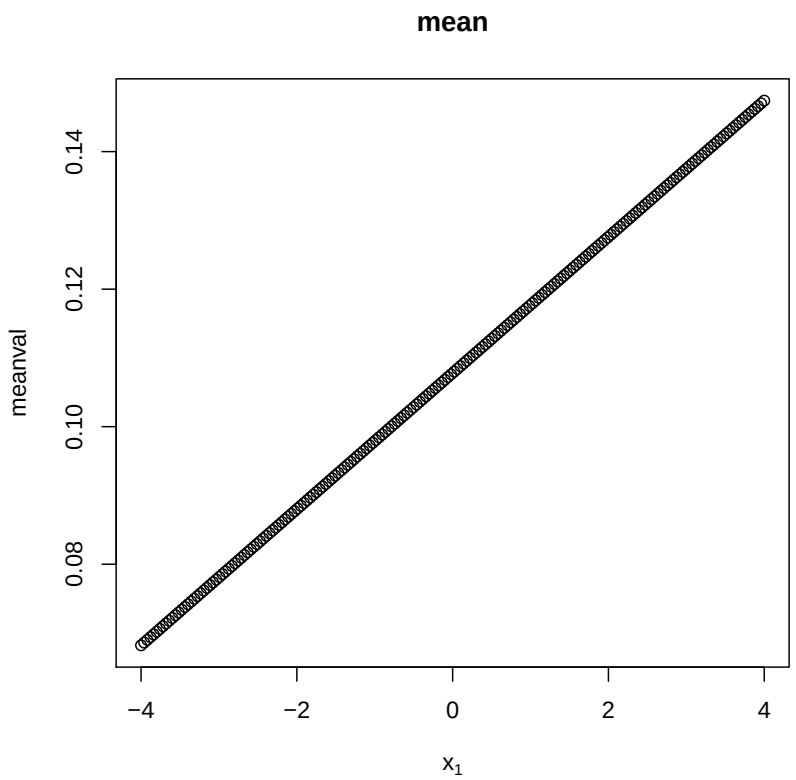
where $\hat{\theta}_{(\cdot)} = \frac{1}{n}\sum_{i=1}^{n} \hat{\theta}_{(i)}$ is the mean of the jackknife estimates of θ.

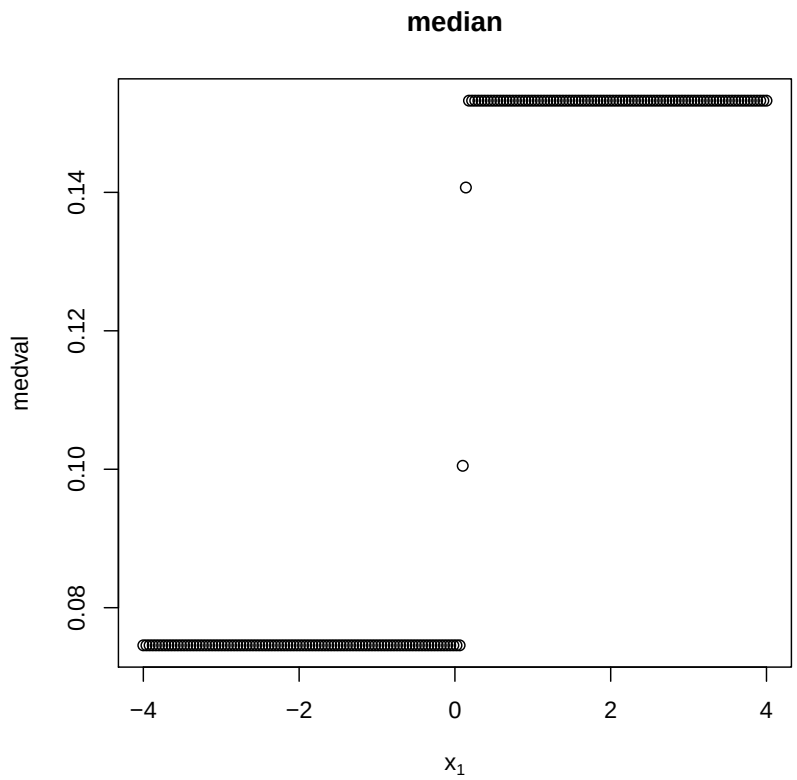
This approach only works for plug-in statistics $\hat{\theta} = t(\hat{F})$.

- Only works if $t(\hat{F})$ is smooth (e.g., mean or ratio)
- Doesn't work if $t(\hat{F})$ is unsmooth (e.g., median)
- Gives bias estimate using only n recomputations (typically $n \ll B$)

Smooth versus Unsmooth $t(\hat{F})$

Suppose we have a sample of data $(x_1, \ldots, x_n)$, and consider the mean and median as a function of x_1:





Smooth versus Unsmooth $t(\hat{F})$ R Code

```
# mean is smooth
meanfun <- function(x,z) mean(c(x,z))
set.seed(1)
z = rnorm(100)
x = seq(-4,4,length=200)
meanval = rep(0,200)
for(j in 1:200) meanval[j] = meanfun(x[j],z)
quartz(width=6,height=6)
plot(x,meanval,xlab=expression(x[1]),main="mean")

# median is unsmooth
medfun <- function(x,z) median(c(x,z))
set.seed(1)
z = rnorm(100)
x = seq(-4,4,length=200)
medval = rep(0,200)
for(j in 1:200) medval[j] = medfun(x[j],z)
quartz(width=6,height=6)
plot(x,medval,xlab=expression(x[1]),main="median")
```

Some R Functions for Jackknife Resampling

We can design our own jackknife functions:

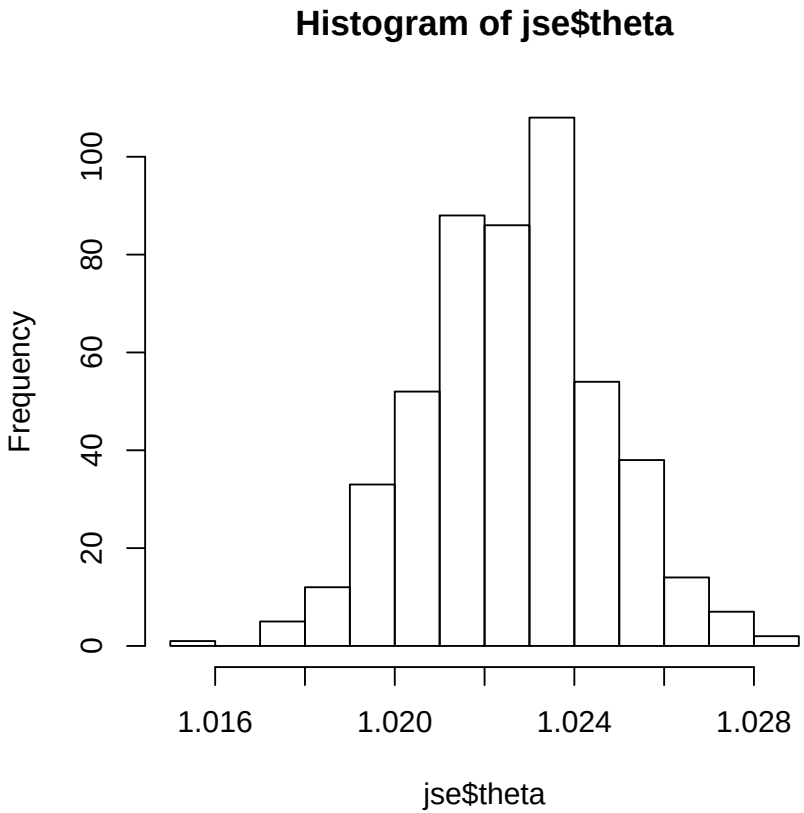
```
jacksamp <- function(x){
  nx = length(x)
  jsamp = matrix(0,nx-1,nx)
  for(j in 1:nx) jsamp[,j] = x[-j]
  jsamp
}

jackse <- function(jsamp,myfun,...){
  nx = ncol(jsamp)
  theta = apply(jsamp,2,myfun,...)
  se = sqrt( ((nx-1)/nx) * sum( (theta-mean(theta))^2 ) )
  list(theta=theta,se=se)
}
```

These functions work similar to the `bootsamp` and `bootse` functions if `x` is a vector and the statistic produced by `myfun` is unidimensional.

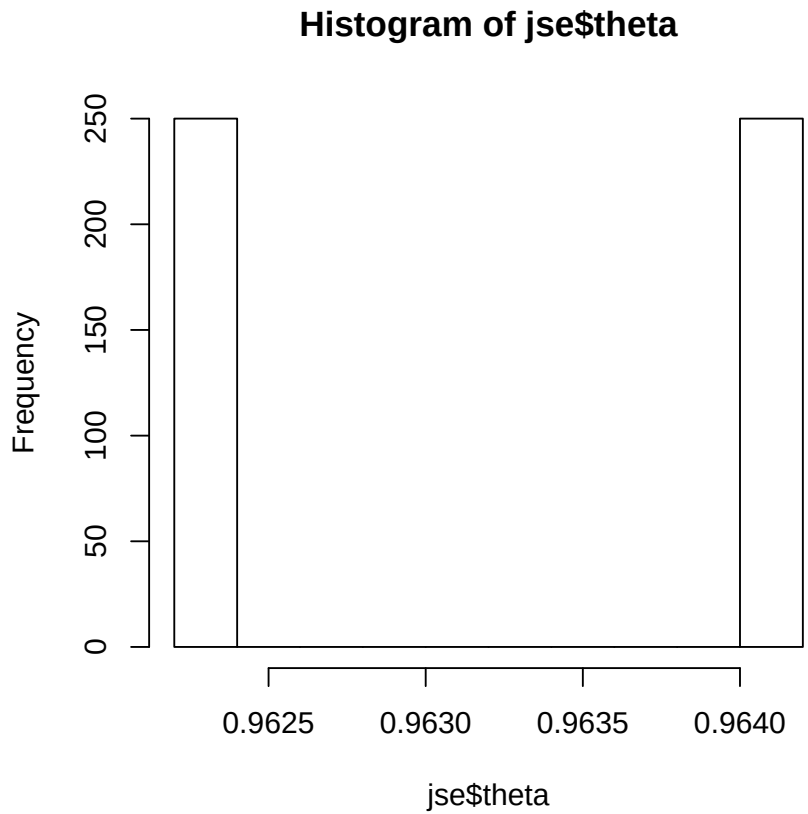
Example 1: Sample Mean (revisited)

```
> set.seed(1)
> x = rnorm(500,mean=1)
> jsamp = jacksamp(x)
> jse = jackse(jsamp,mean)
> mean(x)
[1] 1.022644
> sd(x)/sqrt(500)
[1] 0.04525481
> jse$se
[1] 0.04525481
> hist(jse$theta)
```



Example 2: Sample Median (revisited)

```
> set.seed(1)
> x = rnorm(500,mean=1)
> jsamp = jacksamp(x)
> jse = jackse(jsamp,median)
> median(x)
[1] 0.9632217
> jse$se
[1] 0.01911879
> hist(jse$theta)
```



Bootstrapping Regression

Simple Linear Regression Model: Scalar Form

The simple linear regression model has the form

$$y_i = b_0 + b_1 x_i + e_i$$

for $i \in \{1, \ldots, n\}$ where

- $y_i \in \mathbb{R}$ is the real-valued response for the i-th observation
- $b_0 \in \mathbb{R}$ is the regression intercept
- $b_1 \in \mathbb{R}$ is the regression slope
- $x_i \in \mathbb{R}$ is the predictor for the i-th observation
- $e_i \overset{\text{iid}}{\sim} (0, \sigma^2)$ is zero-mean measurement error

Implies that $(y_i | x_i) \overset{\text{ind}}{\sim} (b_0 + b_1 x_i, \sigma^2)$

Simple Linear Regression Model: Matrix Form

The simple linear regression model has the form

$$\mathbf{y} = \mathbf{X}\mathbf{b} + \mathbf{e}$$

where

- $\mathbf{y} = (y_1, \ldots, y_n)' \in \mathbb{R}^n$ is the $n \times 1$ response vector
- $\mathbf{X} = [\mathbf{1}_n, \mathbf{x}] \in \mathbb{R}^{n \times 2}$ is the $n \times 2$ design matrix
  - $\mathbf{1}_n$ is an $n \times 1$ vector of ones
  - $\mathbf{x} = (x_1, \ldots, x_n)' \in \mathbb{R}^n$ is the $n \times 1$ predictor vector
- $\mathbf{b} = (b_0, b_1)' \in \mathbb{R}^2$ is the 2×1 vector of regression coefficients
- $\mathbf{e} = (e_1, \ldots, e_n)' \sim (\mathbf{0}_n, \sigma^2 \mathbf{I}_n)$ is the $n \times 1$ error vector

Implies that $(\mathbf{y}|\mathbf{x}) \sim (\mathbf{X}\mathbf{b}, \sigma^2 \mathbf{I}_n)$

Ordinary Least Squares: Scalar Form

The ordinary least squares (OLS) problem is

$$\min_{b_0, b_1 \in \mathbb{R}} \sum_{i=1}^{n} (y_i - b_0 - b_1 x_i)^2$$

and the OLS solution has the form

$$\hat{b}_0 = \bar{y} - \hat{b}_1 \bar{x}$$

$$\hat{b}_1 = \frac{\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n} (x_i - \bar{x})^2}$$

where $\bar{x} = (1/n) \sum_{i=1}^{n} x_i$ and $\bar{y} = (1/n) \sum_{i=1}^{n} y_i$

Ordinary Least Squares: Matrix Form

The ordinary least squares (OLS) problem is

$$\min_{\mathbf{b}\in\mathbb{R}^2} \|\mathbf{y} - \mathbf{X}\mathbf{b}\|^2$$

where $\|\cdot\|$ denotes the Frobenius norm; the OLS solution has the form

$$\hat{\mathbf{b}} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y}$$

where

$$(\mathbf{X}'\mathbf{X})^{-1} = \frac{1}{n\sum_{i=1}^n (x_i - \bar{x})^2} \begin{pmatrix} \sum_{i=1}^n x_i^2 & -\sum_{i=1}^n x_i \\ -\sum_{i=1}^n x_i & n \end{pmatrix}$$

$$\mathbf{X}'\mathbf{y} = \begin{pmatrix} \sum_{i=1}^n y_i \\ \sum_{i=1}^n x_i y_i \end{pmatrix}$$

OLS Coefficients are Random Variables

Note that $\hat{\mathbf{b}}$ is a linear function of $\mathbf{y}$, so we can derive the following.

The expectation of $\hat{\mathbf{b}}$ is given by

$$\begin{aligned} \mathrm{E}(\hat{\mathbf{b}}) &= \mathrm{E}[(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y}] \\ &= \mathrm{E}[(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'(\mathbf{X}\mathbf{b}+\mathbf{e})] \\ &= \mathrm{E}[\mathbf{b}] + (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathrm{E}[\mathbf{e}] \\ &= \mathbf{b} \end{aligned}$$

and the covariance matrix is given by

$$\begin{aligned} \mathrm{V}(\hat{\mathbf{b}}) &= \mathrm{V}[(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y}] \\ &= (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathrm{V}[\mathbf{y}]\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1} \\ &= (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'(\sigma^2\mathbf{I}_n)\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1} \\ &= \sigma^2(\mathbf{X}'\mathbf{X})^{-1} \end{aligned}$$

Fitted Values are Random Variables

Similarly $\hat{\mathbf{y}} = \mathbf{X}\hat{\mathbf{b}}$ is a linear function of $\mathbf{y}$, so we can derive...

The expectation of $\hat{\mathbf{y}}$ is given by

$$\begin{aligned}
\mathrm{E}(\hat{\mathbf{y}}) &= \mathrm{E}[\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y}] \\
&= \mathrm{E}[\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'(\mathbf{X}\mathbf{b} + \mathbf{e})] \\
&= \mathrm{E}[\mathbf{X}\mathbf{b}] + \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathrm{E}[\mathbf{e}] \\
&= \mathbf{X}\mathbf{b}
\end{aligned}$$

and the covariance matrix is given by

$$\begin{aligned}
\mathrm{V}(\hat{\mathbf{y}}) &= \mathrm{V}[\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y}] \\
&= \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathrm{V}[\mathbf{y}]\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}' \\
&= \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'(\sigma^2\mathbf{I}_n)\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}' \\
&= \sigma^2\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'
\end{aligned}$$

Need for the Bootstrap

If the residuals are Gaussian $e_i \overset{\text{iid}}{\sim} \text{N}(0, \sigma^2)$, then we have that

- $\hat{\mathbf{b}} \sim \text{N}(\mathbf{b}, \sigma^2(\mathbf{X}'\mathbf{X})^{-1})$.
- $\hat{\mathbf{y}} \sim \text{N}(\mathbf{X}\mathbf{b}, \sigma^2\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}')$

so it is possible to make probabilistic statements about $\hat{\mathbf{b}}$ and $\hat{\mathbf{y}}$.

If $e_i \overset{\text{iid}}{\sim} F$ for some arbitrary distribution F with $E_F(e_i) = 0$, we can use the bootstrap to make inferences about $\hat{\mathbf{b}}$ and $\hat{\mathbf{y}}$.

- Use bootstrap with ecdf $\hat{F}$ as distribution of e_i

Bootstrapping Regression Residuals

Can use following bootstrap procedure:

1. Fit regression model to obtain $\hat{\mathbf{y}}$ and $\hat{\mathbf{e}} = \mathbf{y} - \hat{\mathbf{y}}$
2. Sample e_i^* with replacement from $\{\hat{e}_1, \ldots, \hat{e}_n\}$ for $i \in \{1, \ldots, n\}$
3. Define $y_i^* = \hat{y}_i + e_i^*$ and $\hat{\mathbf{b}}^* = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y}^*$
4. Repeat 2–3 a total of B times to get bootstrap distribution of $\hat{\mathbf{b}}$

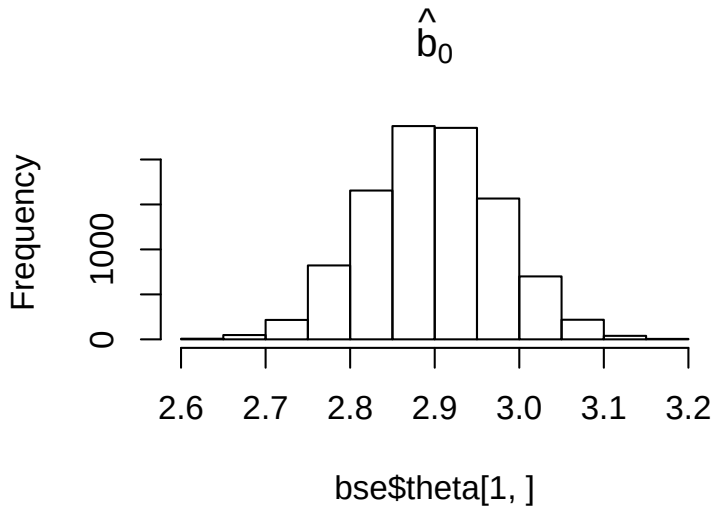
Don't need Monte Carlo simulation to get bootstrap standard errors:

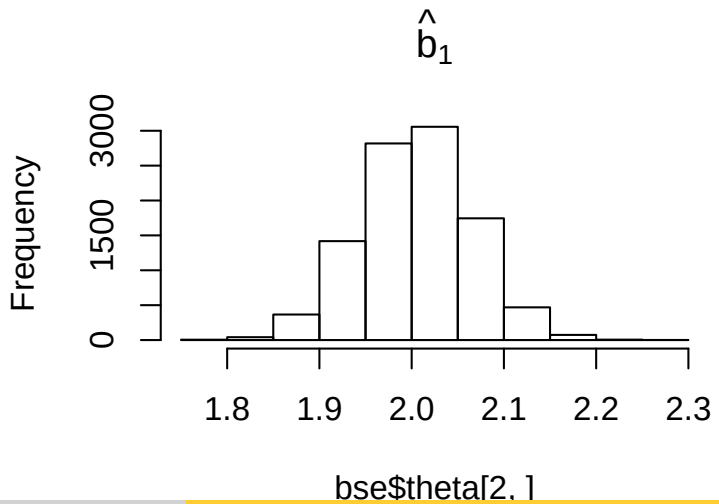
$$\begin{aligned} \mathrm{V}(\hat{\mathbf{b}}^*) &= (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathrm{V}(\mathbf{y}^*)\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1} \\ &= \hat{\sigma}_F^2(\mathbf{X}'\mathbf{X})^{-1} \end{aligned}$$

given that $\mathrm{V}(\mathbf{y}^*) = \hat{\sigma}_F^2\mathbf{I}_n$ where $\hat{\sigma}_F^2 = \sum_{i=1}^n e_i^2/n$

Bootstrapping Regression Residuals in R

```
> set.seed(1)
> n = 500
> x = rexp(n)
> e = runif(n,min=-2,max=2)
> y = 3 + 2*x + e
> linmod = lm(y~x)
> linmod$coef
(Intercept)           x
   2.898325    2.004220
> yhat = linmod$fitted.values
> bsamp = bootsamp(linmod$residuals)
> bsamp = matrix(yhat,n,ncol(bsamp)) + bsamp
> myfun = function(y,x) lm(y~x)$coef
> bse = bootse(bsamp,myfun,x=x)
> bse$cov
             (Intercept)             x
(Intercept)  0.006105788 -0.003438571
x           -0.003438571  0.003605852
> sigsq = mean(linmod$residuals^2)
> solve(crossprod(cbind(1,x))) * sigsq
                         x
   0.006112136 -0.003412774
x -0.003412774  0.003573180
> par(mfcol=c(2,1))
> hist(bse$theta[1,],main=expression(hat(b)[0]))
> hist(bse$theta[2,],main=expression(hat(b)[1]))
```





Bootstrapping Pairs Instead of Residuals

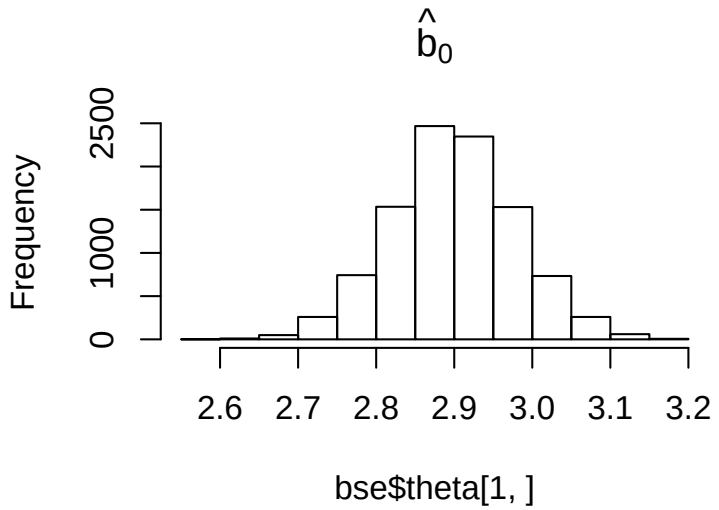
We could also use the following bootstrap procedure:

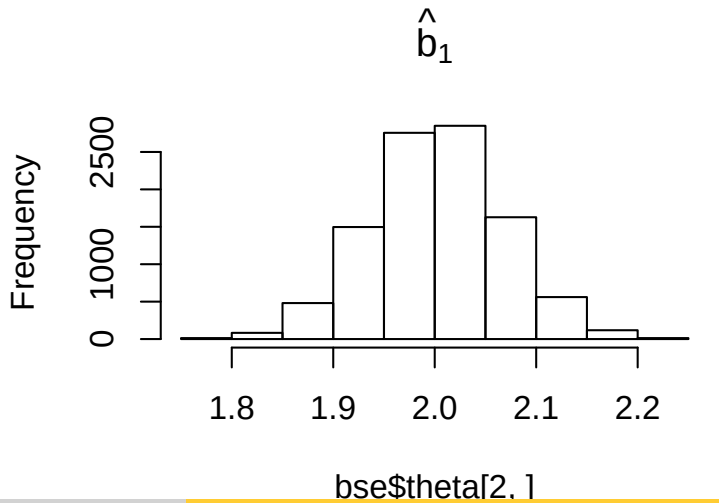
1. Fit regression model to obtain $\hat{\mathbf{y}}$ and $\hat{\mathbf{e}} = \mathbf{y} - \hat{\mathbf{y}}$
2. Sample $\mathbf{z}_i^* = (x_i^*, y_i^*)$ with replacement from $\{(x_1, y_1), \ldots, (x_n, y_n)\}$ for $i \in \{1, \ldots, n\}$
3. Define $\mathbf{x}^* = (x_1^*, \ldots, x_n^*)$, $\mathbf{X}_* = [\mathbf{1}_n, \mathbf{x}^*]$, $\mathbf{y}^* = (y_1^*, \ldots, y_n^*)$, , and $\hat{\mathbf{b}}^* = (\mathbf{X}_*'\mathbf{X}_*)^{-1}\mathbf{X}_*'\mathbf{y}^*$
4. Repeat 2–3 a total of B times to get bootstrap distribution of $\hat{\mathbf{b}}$

Bootstrapping pairs only assumes (x_i, y_i) are iid from some F.

Bootstrapping Regression Pairs in R

```
> set.seed(1)
> n = 500
> x = rexp(n)
> e = runif(n,min=-2,max=2)
> y = 3 + 2*x + e
> linmod = lm(y~x)
> linmod$coef
(Intercept)           x
   2.898325    2.004220
> z = cbind(y,x)
> bsamp = bootsamp(z)
> myfun = function(z) lm(z[,1]~z[,2])$coef
> bse = bootse(bsamp,myfun)
> bse$cov
             (Intercept)       z[, 2]
(Intercept)  0.006376993 -0.003913989
z[, 2]      -0.003913989  0.004308720
> sigsq = mean(linmod$residuals^2)
> solve(crossprod(cbind(1,x))) * sigsq
                          x
   0.006112136 -0.003412774
x -0.003412774  0.003573180
> par(mfcol=c(2,1))
> hist(bse$theta[1,],main=expression(hat(b)[0]))
> hist(bse$theta[2,],main=expression(hat(b)[1]))
```





Bootstrapping Regression: Pairs or Residuals?

Bootstrapping pairs requires fewer assumptions about the data

- Bootstrapping pairs only assumes (x_i, y_i) are iid from some F
- Bootstrapping residuals assumes $(y_i|x_i) \overset{\text{iid}}{\sim} (b_0 + b_1 x_i, \sigma^2)$

Bootstrapping pairs can be dangerous when working with categorical predictors and/or continuous predictors with skewed distributions

Bootstrapping residuals is preferable when regression model is reasonably specified (because **X** remains unchanged).