Homework 6

April 22, 2025

David Lewis
lewis3d7@mail.uc.edu

```R
x <- scan("rabbitblood.txt")
```

```
Read 40 items
```

Question 1

```R
BootstrapMSEmean=function(x,reps){
    n=length(x);
    thetahat=mean(x);
    thetahatbootstrap=rep(0,reps);
    for(i in 1:reps){
    xbootstrap=sample(x,n,replace=TRUE);
    #bootstrap sample
    thetahatbootstrap[i]=mean(xbootstrap)
    #sample mean for bootstrap sample
    }
    MSE=sum((thetahatbootstrap-thetahat)^2)/reps;
    MSE;
}

margin_of_error <- function(MSE){
    2 * sqrt(MSE)
}

MSE <- BootstrapMSEmean(x, 10000)
print(sprintf("MSE=%f", MSE ))
print(sprintf("Margin of Error=%f", margin_of_error(MSE)))
```

```
[1] "MSE=83.363831"
[1] "Margin of Error=18.260759"
```

Question 2

```r
BootstrapBiasmean=function(x,reps){
    n=length(x);
    thetahat=mean(x);
    thetahatbootstrap=rep(0,reps);
    for(i in 1:reps){
        xbootstrap=sample(x,n,replace=TRUE);
        #bootstrap sample
        thetahatbootstrap[i]=mean(xbootstrap)
        #sample mean for bootstrap sample
    }
    Ehat=sum(thetahatbootstrap)/reps
    Bhat=Ehat - thetahat
    Bhat;
}

Bias <- BootstrapBiasmean(x, 10000)
print(sprintf("Bias=%f", Bias ))
```

```
[1] "Bias=0.008312"
```

Question 3

```r
BootstrapMSEmedian=function(x,reps){
    n=length(x);
    thetahat=median(x);
    thetahatbootstrap=rep(0,reps);
    for(i in 1:reps){
    xbootstrap=sample(x,n,replace=TRUE);
    #bootstrap sample
    thetahatbootstrap[i]=median(xbootstrap)
    #sample median for bootstrap sample
    }
    MSE=sum((thetahatbootstrap-thetahat)^2)/reps;
    MSE;
}

margin_of_error <- function(MSE){
    2 * sqrt(MSE)
}

MSE <- BootstrapMSEmedian(x, 10000)
print(sprintf("MSE=%f", MSE ))
print(sprintf("Margin of Error=%f", margin_of_error(MSE)))
```

```
[1] "MSE=82.213050"
[1] "Margin of Error=18.134282"
```

Question 4

```r
Bootstrapmeaninterval=function(x,reps,level){
    n=length(x)
    meanx=mean(x)
    sdx=sd(x)
    v=rep(0,reps)
    for(i in 1:reps){
      xbootstrap=sample(x,n,replace=TRUE)
      #bootstrap sample
      bootstrapmean=mean(xbootstrap)
      bootstrapsd=sd(xbootstrap)
      v[i]=(bootstrapmean-meanx)/(bootstrapsd/sqrt(n))
    }
    alpha=1-level
    lower=quantile(v,alpha/2)
    upper=quantile(v,1-alpha/2)
    left=meanx-upper*sdx/sqrt(n)
    right=meanx-lower*sdx/sqrt(n)
    c(left,right)
}

Bootstrapmeaninterval(x, 10000, 0.90)
```

```
95%        5%
110.8022 142.5475
```

Question 5

As this is population variance instead of sample variance, we need to change n-1 to n in the example from the slides.

```r
BootstrapVarianceinterval=function(x,reps,level){
    n=length(x)
    meanx=mean(x)
    sdx=sd(x)
    v=rep(0,reps)
    for(i in 1:reps){
      xbootstrap=sample(x,n,replace=TRUE)
      #bootstrap sample
      bootstrapmean=mean(xbootstrap)
      bootstrapsd=sd(xbootstrap)
      v[i]= ((n)*(bootstrapsd)^2)/(sdx)^2
    }
    alpha=1-level
    lower=quantile(v,alpha/2)
    upper=quantile(v,1-alpha/2)
    left=(n)*sdx^2/upper
    right=(n)*sdx^2/lower
    c(left,right)
}
BootstrapVarianceinterval(x, 10000, 0.95)
```

```
97.5%      2.5%
2216.561 6954.044
```

Question 6

```r
height <- read.table("height.txt")
x <- height$V1
y <- height$V2
```

```r
BootstrapCorrCI=function(x, y, reps, level){
    reps = 1000
    n=length(x)
    thetahat=cor(x,y)
    thetahatbootstrap=rep(0,reps)
    for(i in 1:reps){
      bootstrap_index=sample(1:n,n,replace=TRUE)
      xbootstrap = x[bootstrap_index]
      #bootstrap sample x
      ybootstrap = y[bootstrap_index]
      #bootstrap sample y
    if((var(xbootstrap)!=0)&(var(ybootstrap)!=0)){
      thetahatbootstrap[i]=cor(xbootstrap, ybootstrap)
      #sample corr for bootstrap sample
      }
    }
    alpha = 1-level
    lower = alpha/2
    upper = 1-alpha/2
    quantile(thetahatbootstrap, prob = c(lower, upper), na.rm = TRUE)
}

BootstrapCorrCI(x, y, 10000, 0.95)
```

```
2.5%     97.5%
0.3168959 0.7596239
```

Question 7

```r
confint <- function(x, y, level){
  fit <- lm(y~x)
  alpha = 1-level
  e = fit$residuals
  sx = sd(x)
  n=length(x)
  SE=sqrt(sum(e^2)/((n-2)*(n-1)*(sx^2)))
  left = fit$coef[2]-SE*qt(1-alpha/2, n-2)
  right = fit$coef[2]-SE*qt(alpha/2, n-2)
  names(left) <- sprintf("%.3f%%",1-alpha/2)
  names(right) <- sprintf("%.3f%%",alpha/2)
  c(left, right)
}

confint(x, y, 0.95)
```

```
0.975%    0.025%
0.3519083 1.1073067
```

Question 8

In method 1, X and Y are both assumed to be random. In this assumption, X and Y pairs are randomly resampled.

In method 2, X is assumed to not be random, but Y is considered to be random. $\hat{h}$ is calulated to obtain observed errors which help create a better estimate for Y as X is assumed to be related to Y.

Question 9

R

```r
fit <- lm(y~x)

sprintf("The estimate of B_0 = %f", fit$coef[1])
sprintf("The estimate of B_1 = %f", fit$coef[2])
```

Results

```
[1] "The estimate of B_0 = 17.131183"
[1] "The estimate of B_1 = 0.729607"
```

Question 10

This uses method 2 and assumes that X is fixed (not normal).

R

```r
SEbeta1=function(x,y){
    fit=lm(y~x)
    e=fit$residuals
    sx=sd(x)
    n=length(x)
    sqrt(sum(e^2)/((n-2)*(n-1)*(sx^2)))
}

bootstrapbeta1=function(x,y,reps,level = 0.95){
    fit=lm(y~x)
    e=fit$residuals
    sx=sd(x)
    tb=rep(0,reps)
    for(i in 1:reps){
      eb = sample(e,replace=TRUE)
      yb = fit$coef[1] + fit$coef[2]*x + eb
      fitb=lm(yb~x);
      tb[i]=(fitb$coef[2] - fit$coef[2])/SEbeta1(x,yb)
    }
    alpha=1-level
    left=fit$coef[2]-SEbeta1(x,y)*quantile(tb,1-alpha/2)
    right=fit$coef[2]-SEbeta1(x,y)*quantile(tb,alpha/2)
    names(left) <- sprintf("%.3f%%",1-alpha/2)
    names(right) <- sprintf("%.3f%%",alpha/2)
    c(left,right)
}

bootstrapbeta1(x, y, 10000, level=0.95)
```

```
0.975%    0.025%
0.3462644 1.1014058
```

Lisp

```lisp
(defun org-typst-code (code _contents info)
  (when-let* ((code-text (org-element-property :value code)))
    (org-typst--raw code-text code info t)))
(defun org-typst-fixed-width (fixed-width _contents info)
  (org-typst--raw (org-element-property :value fixed-width) fixed-width info))
```

```
org-typst-fixed-width
```