4211 Homework 10

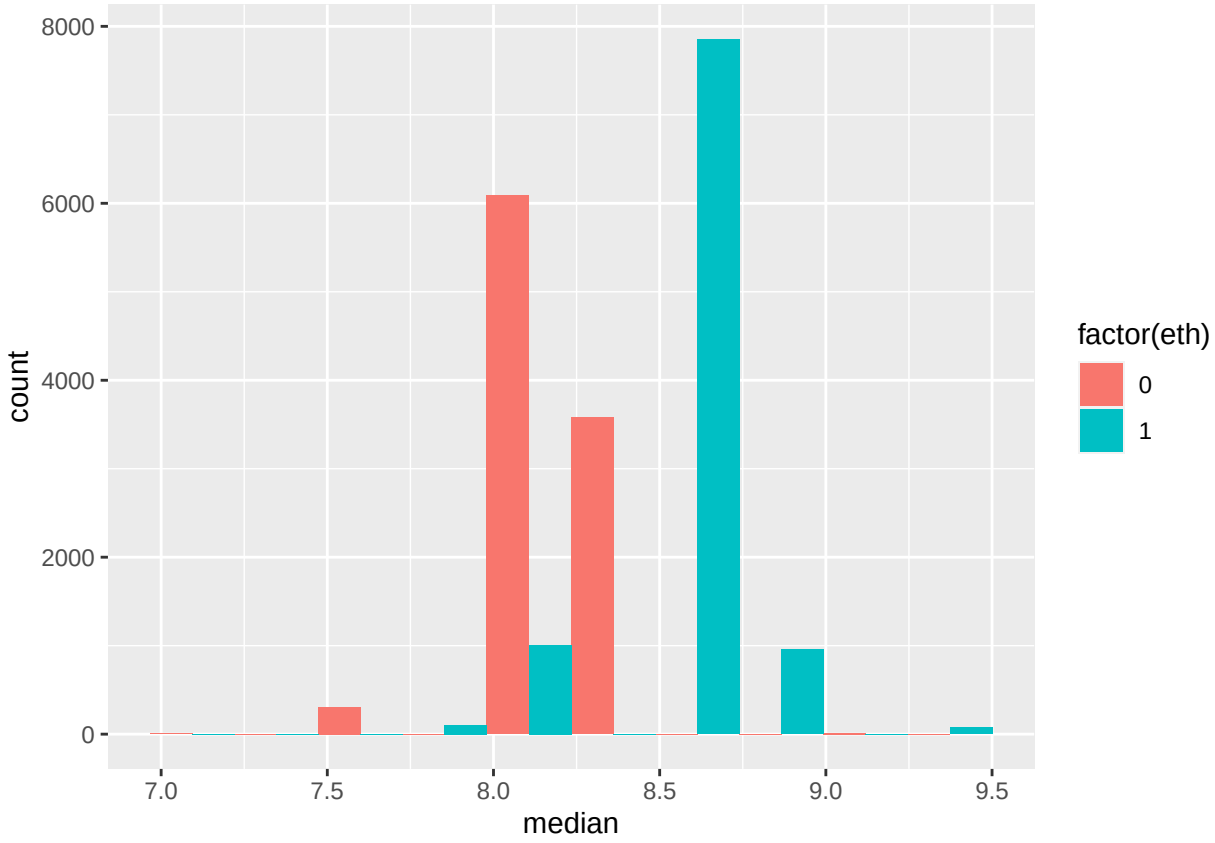
Matthew DeSantis

2023-04-28

1

(a)

```
x = data.frame(
  msce = c(8.27,8.20,8.25,8.14,9.00,8.10,7.20,8.32,7.70,8.50,9.48,8.65,8.16,8.83,7.76,8.63),
  eth = c(rep(0,9), rep(1,7))
)


midtest = function(x){
  xstar = sample(x, replace = TRUE)
  return(median(xstar))
}

set.seed(488103)
bootedc = replicate(10000, midtest(x[x$eth==0,]$msce))
bootedn = replicate(10000, midtest(x[x$eth==1,]$msce))

x2 = data.frame(
  median = c(bootedc, bootedn),
  eth = c(rep(0,10000),rep(1,10000))
)

ggplot(x2, aes(x = median, fill = factor(eth))) + geom_histogram(bins = 10, position = "dodge")
```



(b)

```
cauc = c(8.27,8.20,8.25,8.14,9.00,8.10,7.20,8.32,7.70)
nat = c(8.50,9.48,8.65,8.16,8.83,7.76,8.63)

bootmid = function(x, id){
  xstar = x[id]
  return(median(xstar))
}

boot.ci(boot(cauc, bootmid, 10000))
```

```
## Warning in boot.ci(boot(cauc, bootmid, 10000)): bootstrap variances needed for
## studentized intervals

## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 10000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = boot(cauc, bootmid, 10000))
##
## Intervals :
## Level      Normal              Basic
```

```
## 95%   ( 8.000,  8.434 )   ( 8.080,  8.700 )
##
## Level     Percentile            BCa
## 95%   ( 7.70,  8.32 )   ( 7.70,  8.27 )
## Calculations and Intervals on Original Scale
```

```
boot.ci(boot(nat, bootmid, 10000))
```

```
## Warning in boot.ci(boot(nat, bootmid, 10000)): bootstrap variances needed for
## studentized intervals
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 10000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = boot(nat, bootmid, 10000))
##
## Intervals :
## Level      Normal              Basic
## 95%   ( 8.288,  9.076 )   ( 8.430,  9.100 )
##
## Level     Percentile            BCa
## 95%   ( 8.16,  8.83 )   ( 7.76,  8.65 )
## Calculations and Intervals on Original Scale
```

2

(a)

```
salmon <- read.csv("C:/Users/Hyrul/Desktop/School/6 Semester/Stats for DS 2/Homework/HW5/salmon.dat", s
salmon$R = 1/salmon$recruits
salmon$S = 1/salmon$spawners
mod1 = lm(R~S, data = salmon)
beta0 = as.double(mod1$coefficients[1])
beta1 = as.double(mod1$coefficients[2])
(R = as.double((1 - mod1$coefficients[2])/mod1$coefficients[1]))
```

```
## [1] 150.0976
```

```
samtest = function(x, para = FALSE){

  ids = sample(1:nrow(x), replace = TRUE)

  if(para){
    ers = mod1$residuals[ids]
    x$Rstar = beta0 + beta1*x$S + ers
    bootmod = lm(Rstar~S, data = x)
  }

  else{
    xstar = x[ids,]
    bootmod = lm(R~S, data = xstar)
  }
  return(as.double((1 - bootmod$coefficients[2])/bootmod$coefficients[1]))
}

set.seed(488103)
nonparaboot = replicate(10000, samtest(salmon))
nonparaboot = na.omit(nonparaboot)
```

```
sd(nonparaboot)
```

```
## [1] 3.814935
```

```
quantile(nonparaboot, c(0.025, 0.975))
```

```
##     2.5%    97.5%
## 142.8096 157.7216
```

(b)

```
set.seed(488103)
paraboot = replicate(10000, samtest(salmon, para = TRUE))
paraboot = na.omit(paraboot)
```

```
sd(paraboot)
```

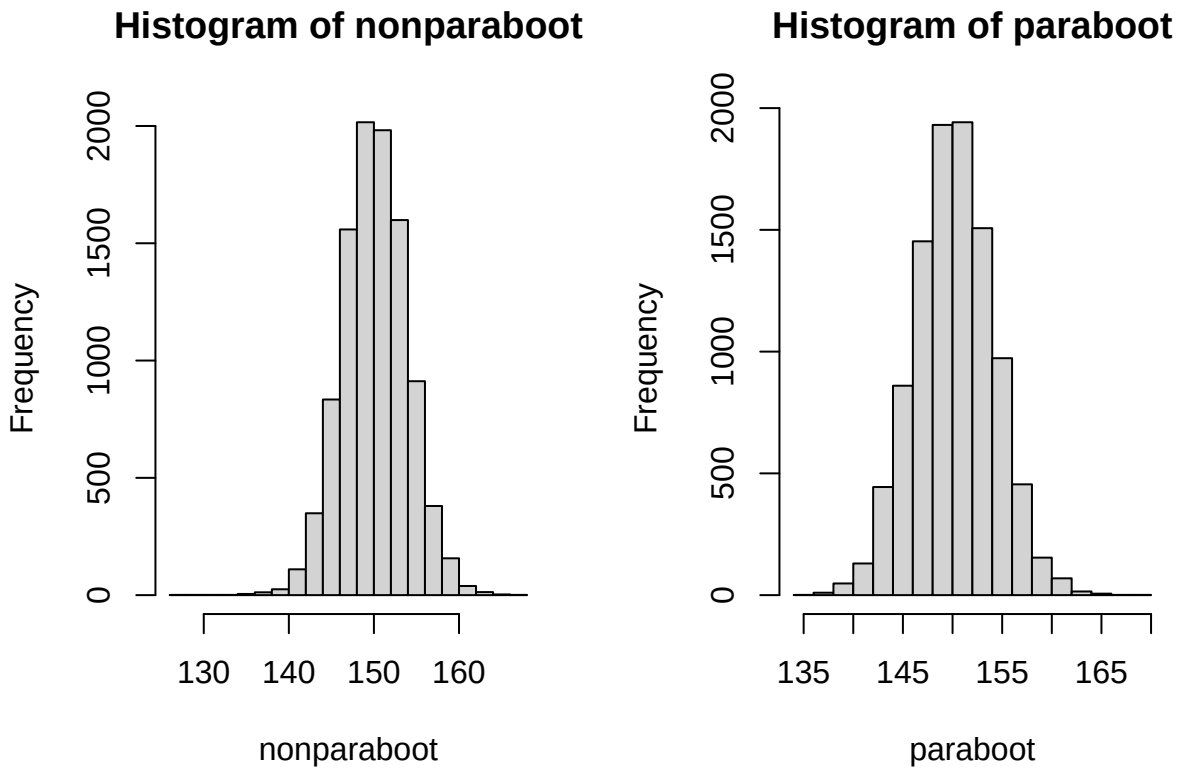
```
## [1] 4.012399
```

```
quantile(paraboot, c(0.025, 0.975))
```

```
##     2.5%    97.5%
## 142.4152 157.9812
```

(c)

```
par(mfrow=c(1,2))
hist(nonparaboot)
hist(paraboot)
```



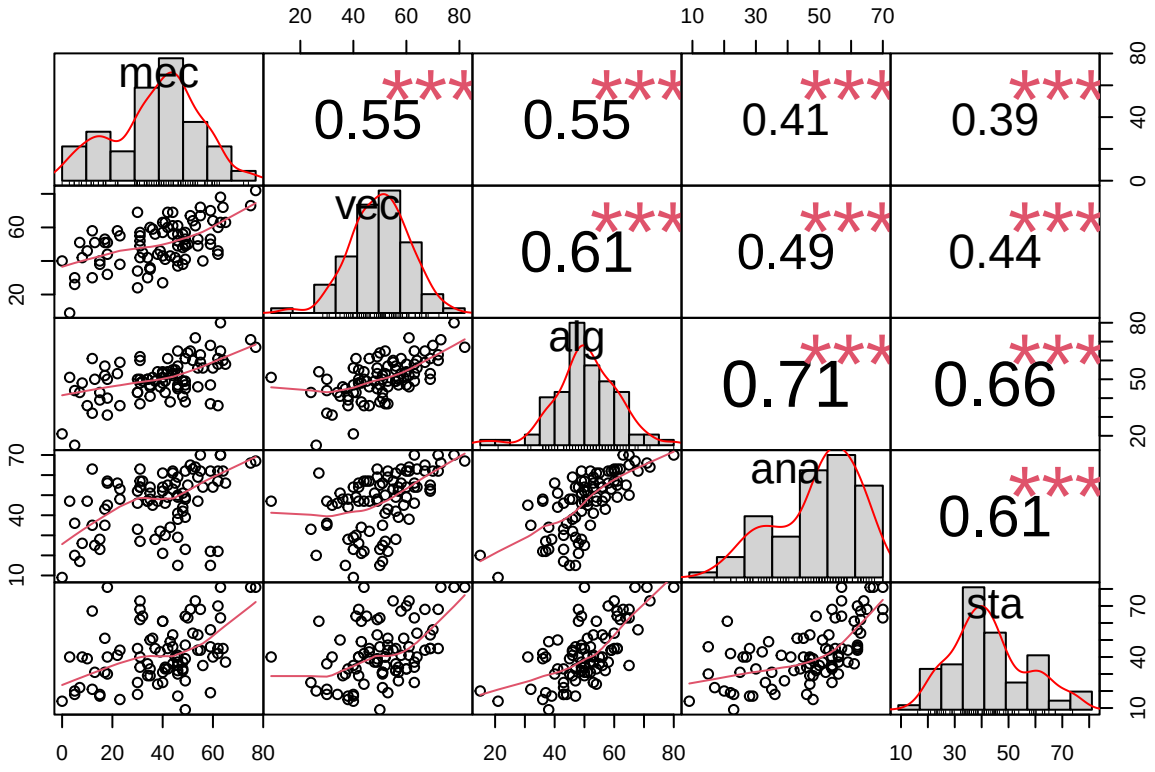
There doesn't appear to be much difference between the parametric and nonparametric approaches in this case.

3

(a)

```
data("scor")
PerformanceAnalytics::chart.Correlation(scor)
```

```
## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter

## Warning in par(usr): argument 1 does not name a graphical parameter
```



(b)

```
cortest = function(x){
  ids = sample(1:nrow(x), replace = TRUE)
  xstar = x[ids,]
  return(c(cor(xstar$mec,xstar$vec), cor(xstar$alg,xstar$ana), cor(xstar$alg,xstar$sta), cor(xstar$ana,x
}

set.seed(488103)
bootcor = replicate(10000, cortest(scor))
sd(bootcor[seq(1, 39997, 4)])
```

```
## [1] 0.07558732
```

```
sd(bootcor[seq(2, 39998, 4)])
```

```
## [1] 0.04829872
```

```
sd(bootcor[seq(3, 39999, 4)])
```

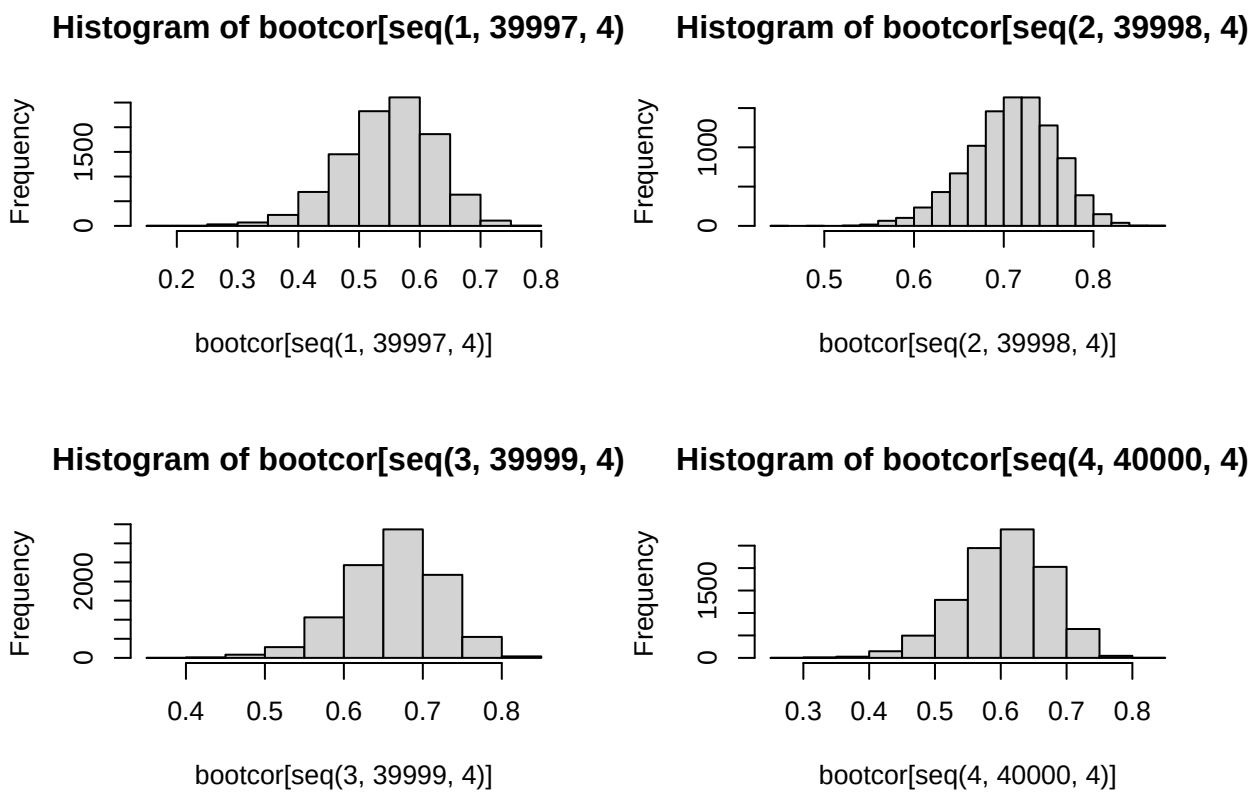
```
## [1] 0.05969669
```

```
sd(bootcor[seq(4, 40000, 4)])
```

```
## [1] 0.06745407
```

(c)

```
par(mfrow=c(2,2))
hist(bootcor[seq(1, 39997, 4)])
hist(bootcor[seq(2, 39998, 4)])
hist(bootcor[seq(3, 39999, 4)])
hist(bootcor[seq(4, 40000, 4)])
```




```
sd(bootcor[seq(1, 39997, 4)])
```

```
## [1] 0.07558732
```

```
sd(bootcor[seq(2, 39998, 4)])
```

```
## [1] 0.04829872
```

```r
sd(bootcor[seq(3, 39999, 4)])
```

```
## [1] 0.05969669
```

```r
sd(bootcor[seq(4, 40000, 4)])
```

```
## [1] 0.06745407
```

```r
quantile(bootcor[seq(1, 39997, 4)], c(0.025, 0.975))
```

```
##      2.5%     97.5%
## 0.3866612 0.6814260
```

```r
quantile(bootcor[seq(2, 39998, 4)], c(0.025, 0.975))
```

```
##      2.5%     97.5%
## 0.6066860 0.7957441
```

```r
quantile(bootcor[seq(3, 39999, 4)], c(0.025, 0.975))
```

```
##      2.5%     97.5%
## 0.5342337 0.7679185
```

```r
quantile(bootcor[seq(4, 40000, 4)], c(0.025, 0.975))
```

```
##      2.5%     97.5%
## 0.4607064 0.7239366
```

As seen by the CIs, all are significant at the confidence level alpha = 0.05

4

(a)

```
thestat = function(x, ids){
  xstar = x[ids,]
  vals = eigen(cov(xstar))$values
  return(vals[1]/(sum(vals)))
}

boot(scor, thestat, 10000)
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = scor, statistic = thestat, R = 10000)
##
##
## Bootstrap Statistics :
##     original      bias    std. error
## t1* 0.619115 0.001405965    0.047403
```

(b)

```
boot.ci(boot(scor, thestat, 10000))
```

```
## Warning in boot.ci(boot(scor, thestat, 10000)): bootstrap variances needed for
## studentized intervals

## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 10000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = boot(scor, thestat, 10000))
##
## Intervals :
## Level      Normal              Basic
## 95%   ( 0.5237,  0.7103 )   ( 0.5281,  0.7164 )
##
## Level     Percentile            BCa
## 95%   ( 0.5219,  0.7101 )   ( 0.5171,  0.7056 )
## Calculations and Intervals on Original Scale
```