Problem Set 5

October 20, 2020

[1]:
```
#if (!requireNamespace("BiocManager", quietly = TRUE))
    #install.packages("BiocManager")

#BiocManager::install("multtest")

# Ignore, this is in class example
#analytical.sd <- 0.005838 n1 <- 1997 n2 <- 906 n3 <- 904 n4 <- 32 n <-␣
↪sum(n1,n2,n3,n4) theta.mle <- 0.0357 prob1 <- 0.25*(2+theta.mle) prob2 <- 0.
↪25*(1-theta.mle) prob3 <- 0.25*(1-theta.mle) prob4 <- 0.25*theta.
↪mle probabilities <- c(prob1,prob2,prob3,prob4) num.sim <- 10000 sd.theta <-␣
↪rep(0,num.sim) theta <- matrix(data=NA, nrow=2,ncol=num.sim) simulated.
↪samples <- rmultinom(num.sim,n,probabilities) for (i in 1:num.sim) {   coef.
↪two <- sum(simulated.samples[1:4,i])   coef.zero <- -2*simulated.
↪samples[4,i]   coef.one <- -simulated.samples[1,i]+2*simulated.
↪samples[2,i]+2*simulated.samples[3,i]+simulated.samples[4,i]   theta[,i] <-␣
↪polyroot(c(coef.zero,coef.one,coef.two)) } sd.theta <- sd(Re(theta[1,]))
```

[20]:
```
library(multtest)
data(golub)
gene1.values = golub[1,]
gene1.values[1:5]
options(warn=-1)
```

1. -1.45769 2. -1.3942 3. -1.42779 4. -1.40715 5. -1.42668

1 Problem 1

[3]:
```
# 1a) since the distribution is normal, the MLE of mu is just its mean
mean(gene1.values)
```

-1.12901315789474

1.b) Since E(mu hat)=E[x]=mu, if n*mu is large, the distribution of x is approximatley normal; hence, that of mu hat is approximatley normal as well.
Because E(mu hat)=mu, we can say the estimate is “unbiased” and the sampling distribution is centered at mu.

1.c) Consistency is generalized by its variance. We cannot say if it is consistent only from mu.

1.d) No.

1.e)

$$\sigma^2 = \mu_2 - \mu_1^2 \mu_2 = E[x^2] = \mu^2 + \sigma^2 \hat{\hat{x}} \hat{\sigma^2} = \frac{1}{n} \sum_{i=1}^{n} (x_i - \bar{x})^2 \tag{1}$$

1.f) Bias in generalized from mu , I cannot tell simply from variance.

1.g)

$$Since\ \hat{\sigma} = \frac{1}{n} \sum_{i=1}^{n} (x_i - \bar{x})^2 \tag{2}$$

shows that the sampling distribution becomes more concentrated and consistent about mu as n increases.

1.h) No.

1.i) The distribution is normal and no.

1.j) By theorem A of section 4.2.1, if the estimate (theta hat) is unbiased, MSE(theta hat) = Var(theta hat)

[4]:
```
#1.k)
 a <- mean(gene1.values)
 s <- sd(gene1.values)
 n <- length(gene1.values)
 error <- qnorm(0.975)*s/sqrt(n)
 lower <- a-error
 upper <- a+error

print("Lower Bound:")
print(lower)

print("Upper Bound:")
print(upper)
```

```
[1] "Lower Bound:"
[1] -1.31591
[1] "Upper Bound:"
[1] -0.9421168
```

2 Problem 2

[5]:
```
yeast.counts = data.frame(cells=0:12,
concen.1 = c(213,128,37,18,3,1,0,0,0,0,0,0,0), concen.2 =
 ↪c(103,143,98,42,8,4,2,0,0,0,0,0,0), concen.3 =
 ↪c(75,103,121,54,30,13,2,1,0,1,0,0,0), concen.4 =
 ↪c(0,20,43,53,86,70,54,37,18,10,5,2,2))
yeast.counts
```

A data.frame: 13 × 5

cells <int>	concen.1 <dbl>	concen.2 <dbl>	concen.3 <dbl>	concen.4 <dbl>
0	213	103	75	0
1	128	143	103	20
2	37	98	121	43
3	18	42	54	53
4	3	8	30	86
5	1	4	13	70
6	0	2	2	54
7	0	0	1	37
8	0	0	0	18
9	0	0	1	10
10	0	0	0	5
11	0	0	0	2
12	0	0	0	2

[6]:

```
#2a
x <- yeast.counts$cells
y1 <- yeast.counts$concen.1
y2 <- yeast.counts$concen.2
y3 <- yeast.counts$concen.3
y4 <- yeast.counts$concen.4

# in a Poisson distribution the Maximum Likelihood estimator of the mean
↪parameter lambda is  the sample mean
mean1 <- sum(x*y1)/sum(y1)
mean2 <- sum(x*y2)/sum(y2)
mean3 <- sum(x*y3)/sum(y3)
mean4 <- sum(x*y4)/sum(y4)
print(mean1)
print(mean2)
print(mean3)
print(mean4)
```

```
[1] 0.6825
[1] 1.3225
[1] 1.8
[1] 4.68
```

[7]:

```
#2b
sd1 <- sd(y1)
sd2 <- sd(y2)
sd3 <- sd(y3)
sd4 <- sd(y4)

se1 <- sd1/sqrt(length(y1))
se2 <- sd2/sqrt(length(y2))
```

```
se3 <- sd3/sqrt(length(y3))
se4 <- sd2/sqrt(length(y4))

print(se1)
print(se2)
print(se3)
print(se4)
```

```
[1] 18.10728
[1] 13.91418
[1] 12.02516
[1] 13.91418
```

[8]:
```
#2c) this was the only question I couldnt get on time
```

[9]:
```
#install.packages("DescTools")

#2d)
library(DescTools)
# first data set CI 95 default

PoissonCI(x=sum(x*y1), n=sum(y1), method = c("exact","score", "wald", "byar"))
 ↪#1st data set
```

A matrix: 4 × 3 of type dbl

	est	lwr.ci	upr.ci
exact	0.6825	0.6039335	0.7684492
score	0.6825	0.6061997	0.7684039
wald	0.6825	0.6015402	0.7634598
byar	0.6825	0.6051067	0.7671272

[10]:
```
# second data set CI 95 default
PoissonCI(x=sum(x*y2), n=sum(y2), method = c("exact","score", "wald", "byar"))
 ↪#1st data set
```

A matrix: 4 × 3 of type dbl

	est	lwr.ci	upr.ci
exact	1.3225	1.212188	1.440153
score	1.3225	1.214502	1.440102
wald	1.3225	1.209802	1.435198
byar	1.3225	1.213383	1.438852

[11]:
```
# third data set CI
PoissonCI(x=sum(x*y3), n=sum(y3), method = c("exact","score", "wald", "byar"))
 ↪#1st data set
```

A matrix: 4 × 3 of type dbl

	est	lwr.ci	upr.ci
exact	1.8	1.670905	1.936421
score	1.8	1.673236	1.936368
wald	1.8	1.668522	1.931478
byar	1.8	1.672108	1.935127

[12]:
```r
# fourth data set CI
PoissonCI(x=sum(x*y4), n=sum(y4), method = c("exact","score", "wald", "byar"))
 #1st data set
```

A matrix: 4 × 3 of type dbl

	est	lwr.ci	upr.ci
exact	4.68	4.470375	4.896917
score	4.68	4.472745	4.896859
wald	4.68	4.467997	4.892003
byar	4.68	4.471596	4.895640

[13]:
```r
#2e) goes with 2.c)
```

3 Problem 3

This is a random varaible because our sample mean will change from sample to sample when we select from the population.

4 Problem 4

4.a)

$$IfE[x] = \mu = \frac{\alpha}{3}, \, thus \, , \, by \, random \, sampling \, , \, E\bar{x} = E[x] = \mu \therefore E[3\bar{x}] = 3\mu = \alpha \tag{3}$$

4.b)

$$E[x^2] - \mu^2 = (\frac{1}{3} - \frac{\alpha^2}{9}), thus \, the \, variance \, of \, [3\bar{x}] = 9Var[x]/n = \frac{(3 - \alpha^2)}{n} \tag{4}$$

4.c)

$$\hat{\alpha} \tag{5}$$

will be asymptotically normally distributed with mean of   and var of (3- ^2)/n.

$$var = (3 - 1^2)/20 = 2/20 = 1/10mean = 1 \tag{6}$$

[14]:
```r
#4.c) calculation
pnorm(.5, mean = 1, sd = sqrt(1/10), lower.tail = FALSE) #.5 to infinity
```

0.943076850996671

5 Problem 5

5.a)

$$\frac{\partial}{\partial\theta}I(\theta) = \frac{-2n_1 + n_2}{1 - \theta} + \frac{2n_3 + n_2}{\theta} 0 = \frac{-2n_1 + n_2}{1 - \theta} + \frac{2n_3 + n_2}{\theta} \theta_{MLE} = \frac{2n_3 + n_2}{2n_1 + 2n_2 + 2n_3} = \frac{2 * 112 + 68}{2 * 190} = .76842 \tag{7}$$

5.b)

$$Var(\theta_{MLE}) \xrightarrow{P} \frac{1}{nl(\theta_{MLE})} = \frac{1}{190(.76642)(1-.76642)} = \frac{1}{190(.17795)} = .03 \tag{8}$$

5.c)

$$I(\theta_{MLE}) = \frac{2n}{\theta_{MLE}(1-\theta_{MLE})} = \frac{2*190}{(.76642)(1-.76642)} = 2,135.42 \tag{9}$$

$$CI = (\theta_{MLE} - \frac{Z(\alpha/2)}{\sqrt{l(\theta_{MLE})}} \ , \ \theta_{MLE} + \frac{Z(\alpha/2)}{\sqrt{l(\theta_{MLE})}}) = (.76842 - \frac{2.576}{\sqrt{2,135.42}} \ , \ .76842 + \frac{2.576}{\sqrt{2,135.42}}) = (.71267 \ , \ .82416) \tag{10}$$

[15]:

```
#5d
sim.samples = rmultinom(10000, size=190, prob=c(0.0526, 0.35789, 0.58947)) #
↪prob = 10/190,68/190, 112/190
sim.samples[,1:10]

thetaMLE = function(n1, n2, n3) {
    return ((2*n3 + n2)/(2*(n1+n2+n3)))
}

theta.hats = apply(sim.samples, 2, function(x) {
    return (thetaMLE(x[1], x[2], x[3]))
 })
```

A matrix: 3 × 10 of type int

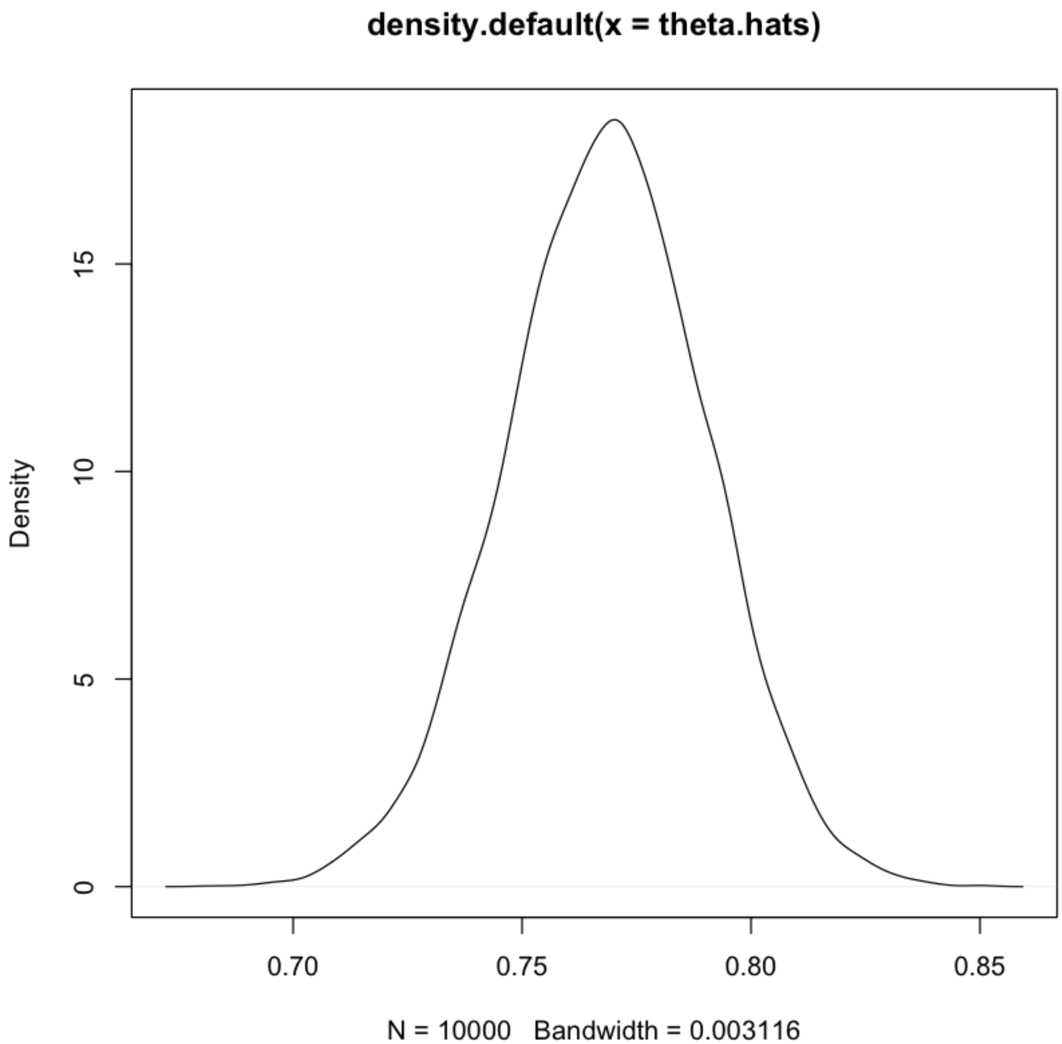
11	9	9	9	15	5	10	12	9	13
59	63	67	78	62	87	67	69	71	51
120	118	114	103	113	98	113	109	110	126

[16]:

```
#5d
theta.hats[1:5]
sd(theta.hats)
plot(density(theta.hats))
```

1. 0.786842105263158 2. 0.786842105263158 3. 0.776315789473684 4. 0.747368421052632 5. 0.757894736842105

0.0218459164262661



5d) The bootstrap sample is actually really close to the MLE done asymptotic. My asymptotic MLE was .76842 and my bootstrap MLE (.768472) is very close to that!

[17]:
```
mean(theta.hats)
```

0.768278157894737

[18]:
```
#5e
denom = ((mean(theta.hats)) * (1 - (mean(theta.hats)))) #denominator
newd = (denom*190) #multiply by n
var = 1/newd #finish equation
print(var)
```

```
[1] 0.02956385
```

5e) our variance rounds also to .03

[19]:
```
#5f
exactCI = function(values, alpha) {
 x.bar = mean(values)
         n = length(values)
 t.n_1 = -qt(alpha/2, df=n-1)
         S = sd(values)
 CI = c(x.bar - (S*t.n_1)/sqrt(n),
 x.bar + (S*t.n_1)/sqrt(n))
 return(CI)
}
 (exact.CI = exactCI(theta.hats, alpha=0.01))
```

1. 0.767715336941264 2. 0.76884097884821

5f) our 99% CI using bootstrap is tighter than our approximation in our part 5c. Both our CI's contain our null value.