Tarea 5

Pablo Gracia Galeana
Cesar Gonzalez Macedo
Miguel Ángel Fuentes Borboa
Roberto Antonio Yglesias Galeana

Pregunta 1

a)

```
Se tienen 12 datos para hacer muestras de tamaño 12 con reemplazo.
Así que el número de posibles muestras es $12^{12}$
```

b)

```
 Se hacen las 100 remuestras
```

```r
X <- c(4.94,5.06,4.53,5.07,4.99,5.16,4.38,4.43,4.93,4.72,4.92,4.96)
Bootstrap <- matrix(0, nrow = 100, ncol = 12)
  for(i in 1:100)
  {
    Bootstrap[i,] <- sample(X,replace = T)
  }
head(Bootstrap)
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12]
## [1,] 4.53 4.99 4.93 4.43 4.94 4.72 5.06 4.94 4.96  4.38  4.99  4.93
## [2,] 4.94 4.94 4.92 5.07 5.06 4.99 5.07 4.99 4.92  4.38  4.53  5.07
## [3,] 4.38 4.93 4.38 5.16 4.96 4.38 4.72 4.96 4.93  4.96  4.93  4.43
## [4,] 4.72 4.94 5.06 4.93 5.07 5.16 4.99 4.53 5.06  5.16  5.06  4.96
## [5,] 4.93 4.94 4.38 4.53 4.72 4.93 4.38 5.07 4.99  5.07  4.38  4.96
## [6,] 5.16 4.38 4.94 4.96 5.06 5.16 4.99 5.16 4.94  4.93  4.92  5.16
```

Media Original

```r
mean(X)
```

```
## [1] 4.840833
```

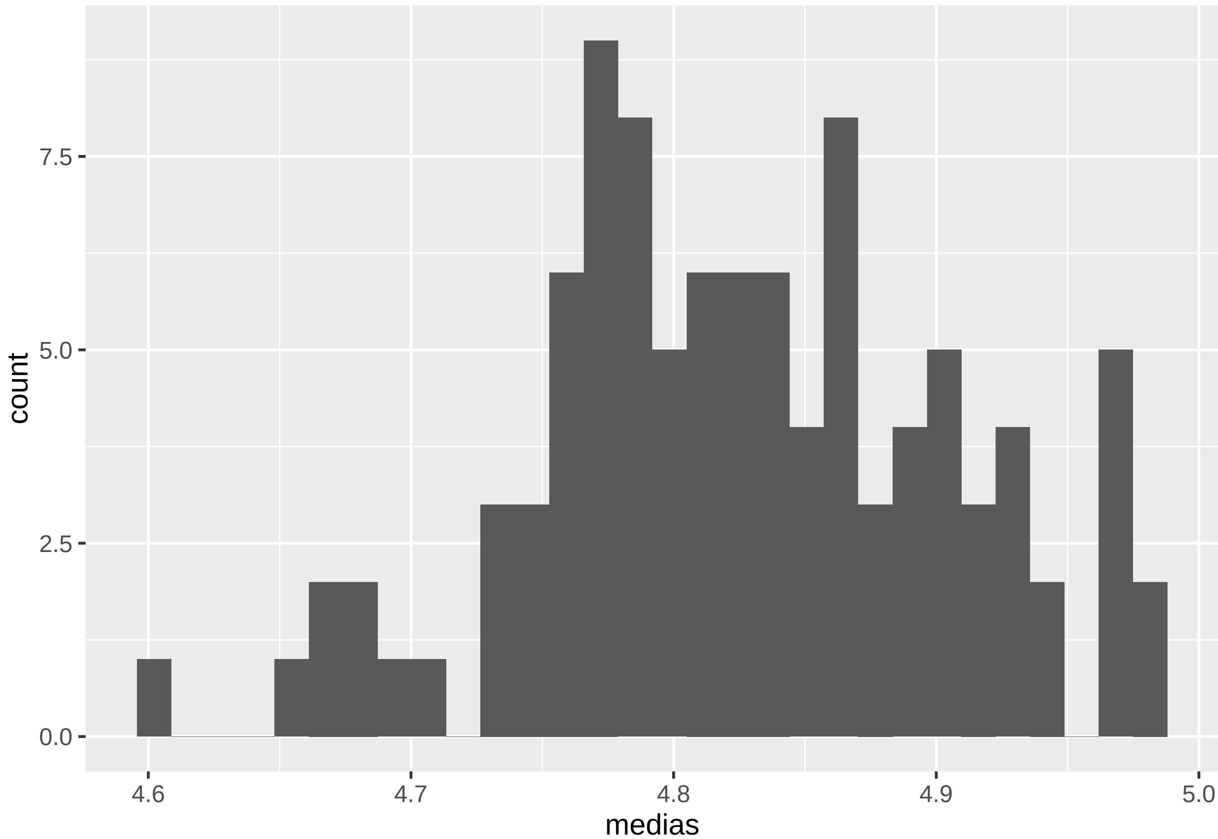
Media Bootstrap

```r
medias <- apply(Bootstrap, 1, mean)
mean(medias)
```

```
## [1] 4.825833
```

```r
qplot(medias)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



c)

Intervalo de confianza del 95% para la media utilizando una aproximación normal estandar.

```
se <- sd(medias)
c(mean(medias)-1.96*se,mean(medias)+1.96*se)
```

```
## [1] 4.670285 4.981382
```

Pregunta 2

a)

```
Accidentes <- c(23,16,21,24,34,30,28,24,26,18,23,23,36,37,49,50,51,56,46,41,54,30,40,31)
```

Media

```
mean(Accidentes)
```

```
## [1] 33.79167
```

Error Estandar

```
sd(Accidentes)
```

```
## [1] 12.06497
```

Mediana

```r
median(Accidentes)
```

```
## [1] 30.5
```

b)

Se hacen las muestras bootstraps

```r
BootAcc <- matrix(0, ncol = 24, nrow = 1000)
for(i in 1:1000){
  BootAcc[i,] <- sample(Accidentes, replace = T)
}
```

Estimación de la media, mediana y error estandar

```r
mediab <- round(mean(apply(BootAcc, 1, mean)), digits = 4)
medianab <- mean(apply(BootAcc, 1, median))
se <- round(mean(apply(BootAcc, 1, sd)), digits = 4)
```

	Media	Mediana	Error Estandar
Muestra Original	33.7917	30.5	12.065
Bootstrap	33.7656	31.515	11.7211

Mediana de las medianas

```r
median(apply(BootAcc, 1, median))
```

```
## [1] 30.5
```

c)

Las estimaciónes de la media, mediana y el error, se modificaron al aplicarles la media. La mediana de la mediana no se modificó.

Pregunta 3

Primero se corre el bootsrap para $\hat{\rho}$

```r
data("law")
n <- nrow(law)
corr <- numeric(1000)
indices <- matrix(0,nrow=1000, ncol=n)

for (k in 1:1000) {
i <- sample(1:n, size = n, replace = T)
LSAT <- law$LSAT[i]
GPA <- law$GPA[i]
corr[k] <- cor(LSAT,GPA)
indices[k, ] <- i
}
```

Estimación bootstrap de $\hat{\rho}$

```
mean(corr)
```

```
## [1] 0.7733149
```

Luego se usa Jackknife para estimar $se(\hat{\rho})$

```
se.corr <- NULL
for (i in 1:n) {
theta_i <-(1:1000)[apply(indices, MARGIN = 1, FUN = function(k) {!any(k==i)})]
se.corr[i] <- sd(corr[theta_i])
}
```

Valor estimado jackknife-after-bootstrap de el error estándar

```
sqrt((n-1)*mean((se.corr-mean(se.corr))^2))
```

```
## [1] 0.07640069
```

Pregunta 4

Si $X \sim exp(\lambda)$ entonces $f(x|\lambda) = \lambda e^{\lambda x}$ Para encontr el estimador de máxima verosimilitud hay que maximizar la función $l(\lambda) = \ln(L(\lambda|\underline{X}_{(n)}))$

$$L(\lambda|\underline{X}) = \prod_{i=i}^{n} \{\lambda e^{\lambda x_i}\} = \lambda^n e^{-\lambda \sum x_i}$$

por lo tanto

$$l(\lambda) = n \ln(\lambda) - \lambda \sum_{i=1}^{n} x_i$$

Por lo tanto el el estimador de máxima verosimilitud es

$$\hat{\theta} \text{ tal que } l'(\hat{\theta}) = 0$$

$$\frac{dl}{d\theta} = 0 \Leftrightarrow \frac{n}{\lambda} - \sum_{i=1}^{n} x_i = 0$$

$$\Leftrightarrow \hat{\theta} = \sum_{i=1}^{n} \frac{x_i}{n}$$

```
horas <- c(3, 5, 7, 18, 43, 85, 91, 98, 100, 130, 230, 487)
media <- mean(horas)
media
```

```
## [1] 108.0833
```

Ahora se hace el Bootstrap

```
lambdab <- matrix(0, nrow = 1000, ncol = 12)

for(i in 1:1000){
  lambdab[i,] <- sample(horas, replace = T)
}
```

Valor Estimado bootstrap

```r
mediaboot <- mean(apply(lambdab, 1, mean))
mediaboot
```

```
## [1] 108.1167
```

Estimación del sesgo

```r
mediaboot-media
```

```
## [1] 0.03341667
```

Error estandar

```r
sd(apply(lambdab, 1, mean))
```

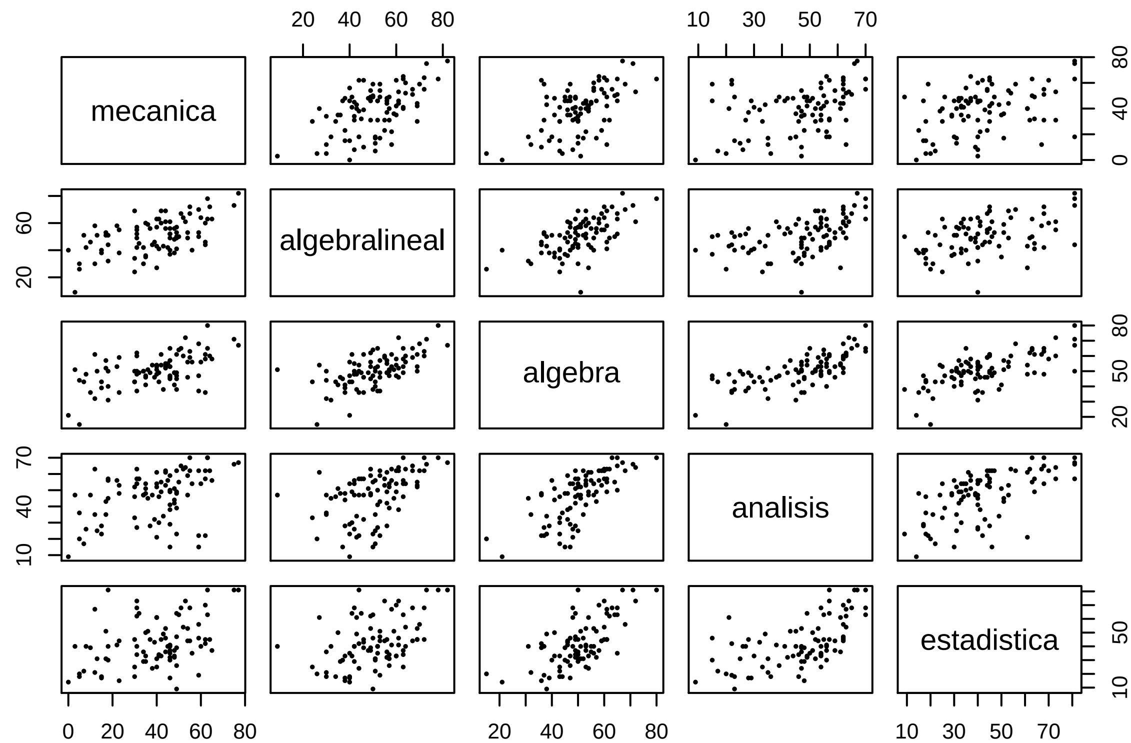
```
## [1] 37.32478
```

Pregunta 5

a)

```r
data("scor")

mecanica <- scor$mec
algebralineal <- scor$vec
algebra <- scor$alg
analisis <- scor$ana
estadistica <- scor$sta
pairs(cbind(mecanica,algebralineal,algebra,analisis,estadistica),pch = 16,cex = .5)
```



b)

```
n <- nrow(scor)
mecboot <- matrix(0, nrow=2000, ncol=n)
alglinboot <- matrix(0, nrow=2000, ncol=n)
algboot <- matrix(0, nrow=2000, ncol=n)
anaboot <- matrix(0, nrow=2000, ncol=n)
estboot <- matrix(0, nrow=2000, ncol=n)
rho.m.v <- NULL
rho.al.an <- NULL
rho.al.st <- NULL
rho.an.st <- NULL

for(i in 1:2000){
  mecboot[i,] <- sample(mecanica ,replace = T)
  alglinboot[i,] <- sample(algebralineal ,replace = T)
  algboot[i,] <- sample(algebra,replace = T)
  anaboot[i,] <- sample(analisis ,replace = T)
  estboot[i,] <- sample(estadistica ,replace = T)
  rho.m.v[i] <- cor(mecboot[i,],alglinboot[i,])
  rho.al.an[i] <- cor(algboot[i,],anaboot[i,])
  rho.al.st[i] <- cor(algboot[i,],estboot[i,])
  rho.an.st[i] <- cor(anaboot[i,],estboot[i,])
}
```

Error estandar de $\hat{\rho}_{mec,vec}$

```
sd(rho.m.v)
```

```
## [1] 0.1095484
```

Error estandar de $\hat{\rho}_{alg,ana}$

```
sd(rho.al.an)
```

```
## [1] 0.1083822
```

Error estandar de $\hat{\rho}_{alg,sta}$

```
sd(rho.al.st)
```

```
## [1] 0.1085988
```

Error estandar de $\hat{\rho}_{ana,sta}$

```
sd(rho.an.st)
```

```
## [1] 0.1034673
```

c)

```
theta <- NULL
eigenvalores <- NULL
for(i in 1:2000){
mat.var.cov <- var(cbind(mecboot[i,],
      alglinboot[i,],
      algboot[i,],
      anaboot[i,],
      estboot[i,]))
eigenvalores <- eigen(mat.var.cov)$values
theta[i] <- eigenvalores[1]/sum(eigenvalores)
}
head(theta)
```

```
## [1] 0.3420837 0.2931053 0.3070835 0.3260830 0.3378865 0.3138984
```

Estimador Bootstrap sesgo

```
eigenv <-  eigen(var(scor))$values
theta1 <- eigenv[1]/sum(eigenv)
mean(theta)-theta1
```

```
## [1] -0.300968
```

Estimador Bootstrap del error estandar

```
sd(theta)
```

```
## [1] 0.02368403
```

d)

Estimador Jackkife del sesgo

```
theta <- function(x){
eigenv <-  eigen(var(x))$values
return(eigenv[1]/sum(eigenv))
}
jackknife(scor, theta)$jack.bias
```

```
## [1] 0.12228
```

Estimador Jackkife del del error estandar

```
jackknife(scor, theta)$jack.se
```

```
## [1] 0.05342633
```

e)

```
theta <- function(x,i){
       mec <- x[i,1]
      alglin <- x[i,2]
      alg <- x[i,3]
      ana  <- x[i,4]
      est <- x[i,5]

eigenv <-  eigen(var(cbind(mec,alglin,alg,ana,est)))$values
return(eigenv[1]/sum(eigenv))
}
```

Intervalo Percentil

```
boot.ci(boot(data=scor, statistic=theta, R=2000), type=c("perc"))
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 2000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = boot(data = scor, statistic = theta, R = 2000),
##     type = c("perc"))
##
## Intervals :
## Level     Percentile
## 95%   ( 0.5226,  0.7130 )
## Calculations and Intervals on Original Scale
```

Intervalo BC_a

```
boot.ci(boot(data=scor, statistic=theta, R=2000), type=c("bca"))
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 2000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = boot(data = scor, statistic = theta, R = 2000),
##     type = c("bca"))
##
## Intervals :
## Level       BCa
## 95%   ( 0.5218,  0.7084 )
## Calculations and Intervals on Original Scale
```

Pregunta 6

Se hacen las pruebas sobre muestras de tamaño 10 de una $U \sim uniforme(0,1)$ y una $Z \sim N(0,1)$

```
pruebaSpearman <- function(x,y){
s1 <- sample(x,replace=T)
s2 <- sample(y,replace=T)
return(cor(s1,s2, method = "spearman"))
}
u <- runif(10)
z <- rnorm(10)
theta <- NULL
B <- 1000
for(i in 1:B) {
  theta[i] <- pruebaSpearman(u,z)
}
ASL <-sum(theta > cor(u,z, method = "spearman"))/B
ASL
```

```
## [1] 0.426
```

```
cor.test(runif(10),rnorm(10))
```

```
##
##  Pearson's product-moment correlation
##
## data:  runif(10) and rnorm(10)
## t = 0.53204, df = 8, p-value = 0.6092
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.5033523  0.7295701
## sample estimates:
##       cor
## 0.1848608
```

Para ambas pruebas pasa la prueba de independencia. Pero la prueba de cor.test, le da mayor factibilidad a que sean independientes que la de permutación.