TP 1 - Muestreo

Gomez Vargas Andrea, Iummato Luciana, Pesce Andrea Gisele

2024-11-10

Contenido

Paquetes de trabajo

```
library(tidyverse)
library(survey)
library(readxl)
library(gt)
library(sampling)
```

Ejercicio I

```
datos <- "
Alumno X Y
a 6 14.0
b 9 20.0
c 5 12.0
d 4 10.0
e 2 5.0
f 7 12.0
g 10 24.0
h 4 5.0
i 12 21.0
j 5 9.0
k 8 18.0
l 12 20.0
m 5 8.0
n 9 15.0
o 2 2.5
p 6 11.0
q 11 20.0
r 8 15.0
"

# Convertimos los datos en un data.frame
ejercicio_1 <- read.table(text = datos, header = TRUE)


N= nrow(ejercicio_1)
n=9

#parámetros
parametro_mediaX= sum(ejercicio_1$X)/N
parametro_mediaY=sum(ejercicio_1$Y)/N
parametro_razonX_Y= sum(ejercicio_1$X)/sum(ejercicio_1$Y)

# Generar todas las combinaciones posibles
muestras_posibles <- combn(N, n)

# Ver la cantidad de combinaciones posibles
cantidad_muestras <- ncol(muestras_posibles)
cantidad_muestras
```

```
## [1] 48620
```

```
#RESOLUCIÓN PUNTOS 2 Y 3 CON UNA MUESTRA SELECCIONADA
# Seleccionamos ahora una muestra aleatoria simple con R de tamaño n
s_mas <- sample(N,n, replace=FALSE)

muestra <- ejercicio_1[s_mas,]

#estimadores para esa muestra
```

```r
estimador_mediaX= sum(muestra$X)/N
estimador_mediaY=sum(muestra$Y)/N
estimador_razonX_Y= estimador_mediaX/estimador_mediaY

#varianza de estimadores
s_cuadradoX=var(muestra$X)
s_cuadradoY=var(muestra$Y)

# varianza para la estimacion de la muestra:
# En el MAS
# Var(y_media) = (1-n/N)*S^2/n
Var_X_media = (1-n/N)*s_cuadradoX/n
Var_Y_media =(1-n/N)*s_cuadradoY/n
#no se como se hace la fórmula para una razón a partir de la muestra seleccionada

#coeficiente de variacion para la estimacion de la muestra:
CVMAS_X <- 100*sqrt(Var_X_media)/estimador_mediaX

CVMAS_X
```

```
## [1] 26.3679
```

```r
CVMAS_Y <- 100*sqrt(Var_Y_media)/estimador_mediaY

CVMAS_Y
```

```
## [1] 27.43669
```

```r
#no lo hice para la razón porque no se hacer la varianza para la razón de esa muestra en particular
#es una fórmula muy larga que creo que no vimos porque tiene covariación

#CALCULO CON SURVEY PARA ESA MUESTRA EN PARTICULAR
muestra$R <- muestra$X/muestra$Y
muestra$pondera <- N/n

muestra$fpc <- N

library(survey)
diseno <- svydesign(id= ~1,weights=~pondera, data=muestra, fpc=~fpc)
diseno
```

```
## Independent Sampling design
## svydesign(id = ~1, weights = ~pondera, data = muestra, fpc = ~fpc)
```

```r
# Calcular las medias
media_X <- svymean(~X, diseno)
media_Y <- svymean(~Y, diseno)

# Calcular la razón
razon <- as.numeric(media_X / media_Y)

# Mostrar la razón estimada
razon
```

```
## [1] 0.5388601
```

```
#RESOLUCIÓN DE PUNTOS 2 Y 3 CON EL CALCULO DE TODAS LAS MUESTRAS POSIBLES (ESTO SI LO VIMOS EN CLASE)
# Crear un vector para almacenar los valores de estimadores para cada muestra

medias_X <- numeric(ncol(muestras_posibles))
medias_Y <- numeric(ncol(muestras_posibles))
razon <- numeric(ncol(muestras_posibles))

# Calcular estimadores para cada muestra
for (i in 1:ncol(muestras_posibles)) {
  indices <- muestras_posibles[, i]
  medias_X[i] <- mean(ejercicio_1$X[indices])
  medias_Y[i] <- mean(ejercicio_1$Y[indices])
  razon[i] <- medias_X[i] / medias_Y[i]
}

# Crear el data frame con las tres columnas
muestras_posibles_estimadores <- data.frame(medias_X, medias_Y, razon)

#esperanza del estimador
esperanza_mediaX=mean(muestras_posibles_estimadores$medias_X) #insesgado
esperanza_mediaY=mean(muestras_posibles_estimadores$medias_Y) #insesgado
esperanza_razon=mean(muestras_posibles_estimadores$razon) #aproximado

esperanza_mediaX
```

```
## [1] 6.944444
```

```
esperanza_mediaY
```

```
## [1] 13.41667
```

```
esperanza_razon
```

```
## [1] 0.5182023
```

```
# Calcular la varianza de los estimadores
varianza_medias_X <- var(muestras_posibles_estimadores$medias_X)
varianza_medias_Y <- var(muestras_posibles_estimadores$medias_Y)
varianza_razon <- var(muestras_posibles_estimadores$razon)

varianza_medias_X
```

```
## [1] 0.5455813
```

```
varianza_medias_Y
```

```
## [1] 2.147511
```

```r
varianza_razon
```

```
## [1] 0.0004556671
```

```r
# Calcular la varianza de los estimadores
CVestimador_X <- 100*sqrt(varianza_medias_X)/esperanza_mediaX
CVestimador_y <- 100*sqrt(varianza_medias_Y)/esperanza_mediaY
CVestimador_r <- 100*sqrt(varianza_razon)/esperanza_razon

CVestimador_X
```

```
## [1] 10.63634
```

```r
CVestimador_y
```

```
## [1] 10.92253
```

```r
CVestimador_r
```

```
## [1] 4.11931
```

```r
#SELECCIÓN DE 10000 MUESTRAS

# Creo una funcion que seleccione una muestra de tamanio x
# y estime total de poblacion
estimo_diezmil <- function() {
  muestras <- ejercicio_1[sample(nrow(ejercicio_1), 9, replace = FALSE), ]  # Tamaño de muestra de 9
  estim_X <- mean(muestras$X)  # Calcular la media de X
  estim_Y <- mean(muestras$Y)  # Calcular la media de Y
  a <- c(estim_X, estim_Y)  # Crear un vector con las estimaciones
  return(a)  # Devolver el vector
}

# Crear una lista para almacenar las estimaciones
lista_estim <- lapply(1:10000, function(x) estimo_diezmil())

# Convertir la lista en un data frame
df_estimdiezmil <- data.frame(matrix(unlist(lista_estim),
                                     nrow = length(lista_estim), byrow = TRUE))

# Asignar nombres a las columnas
colnames(df_estimdiezmil) <- c("media_X", "media_Y")
df_estimdiezmil$razon <- df_estimdiezmil$media_X/df_estimdiezmil$media_Y


# Calcular la varianza de los estimadores
var_medias_X <- var(df_estimdiezmil$media_X)
var_medias_Y <- var(df_estimdiezmil$media_Y)
var_razon <- var(df_estimdiezmil$razon)

var_medias_X
```

```
## [1] 0.5387685
```

```
var_medias_Y
```

```
## [1] 2.114545
```

```
var_razon
```

```
## [1] 0.000453615
```

```
# Calcular la varianza de los estimadores
CV_X <- 100*sqrt(var_medias_X)/mean(df_estimdiezmil$media_X)
CV_y <- 100*sqrt(var_medias_Y)/mean(df_estimdiezmil$media_Y)
CV_r <- 100*sqrt(var_razon)/mean(df_estimdiezmil$razon)

CV_X
```

```
## [1] 10.57157
```

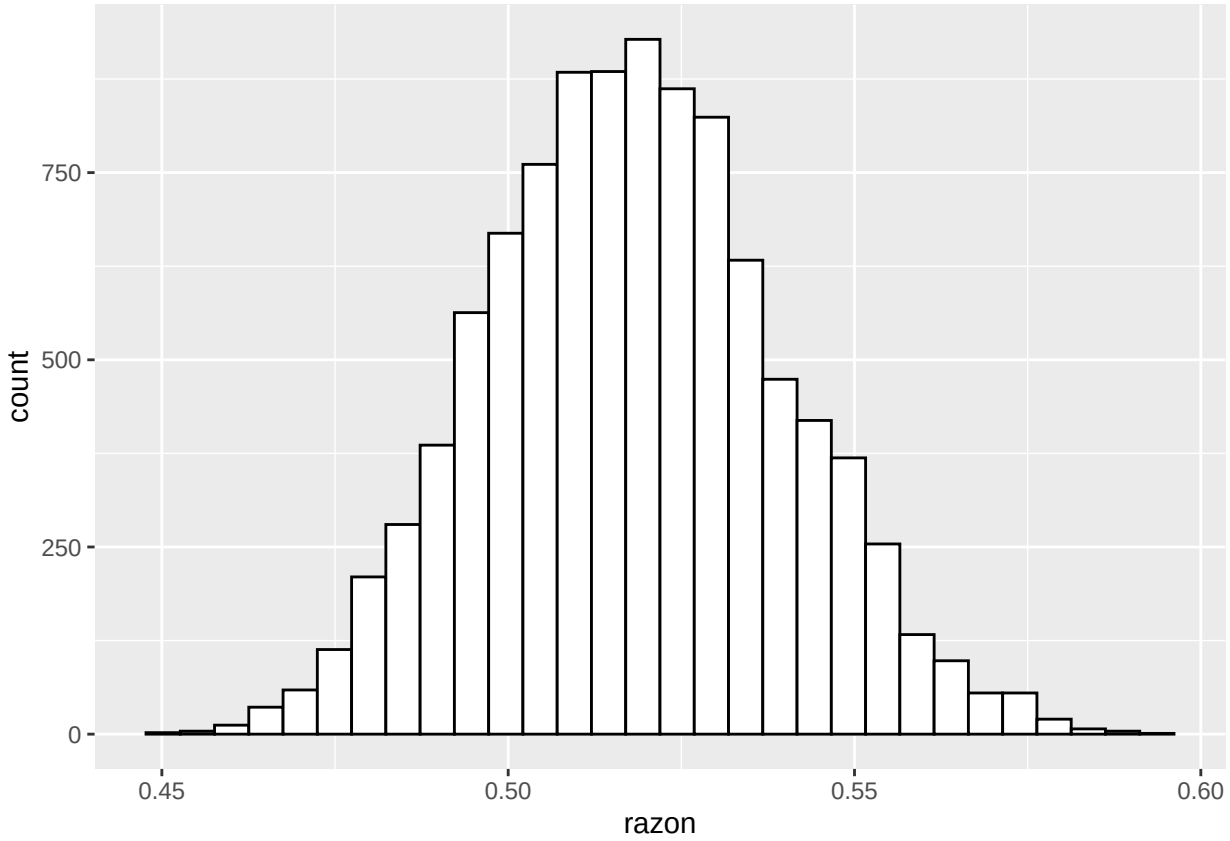
```
CV_y
```

```
## [1] 10.83862
```

```
CV_r
```

```
## [1] 4.110793
```

```
# Grafico la distribucion de las estimaciones
p <-ggplot(df_estimdiezmil, aes(x=razon)) +
  geom_histogram(color="black", fill="white")
p
```



Ejercicio II

```
df_tabla <- read_excel("tabla_muestras_posibles.xlsx")

#TODAS LAS MUESTRAS POSIBLES
df_muestras <- data.frame(matrix(unlist(combn(df_tabla$Y,10, simplify = FALSE)),
                                 ncol=10, byrow=TRUE))

#1 y 2. MEDIA, MEDIANA, MEDIA TRUNCADA DE TODAS LAS MUESTRAS POSIBLES

df_muestras[, 1:10] <- t(apply(df_muestras[, 1:10], 1, sort))

media <- apply(df_muestras[,1:10],1,mean)
mediana <- apply(df_muestras[,1:10],1,median)
media_t <- apply(df_muestras[,2:9],1,mean)

df_muestras$Media <- media
df_muestras$Mediana <- mediana
df_muestras$MediaTrunc <- media_t


#verificación de que la media es un estimador insesgado de la media poblacional
```

```r
media_poblacional=mean(df_tabla$Y)
mediana_poblacional=median(df_tabla$Y)
mediaTrunc_poblacional=mean(df_tabla$Y, trim = 0.1)

media_poblacional
```

```
## [1] 42.435
```

```r
mediana_poblacional
```

```
## [1] 28.37879
```

```r
mediaTrunc_poblacional
```

```
## [1] 39.34896
```

```r
media_muestras=mean(df_muestras$Media)
mediana_muestras=mean(df_muestras$Mediana)
mediaTrunc_muestras=mean(df_muestras$MediaTrunc)

media_muestras
```

```
## [1] 42.435
```

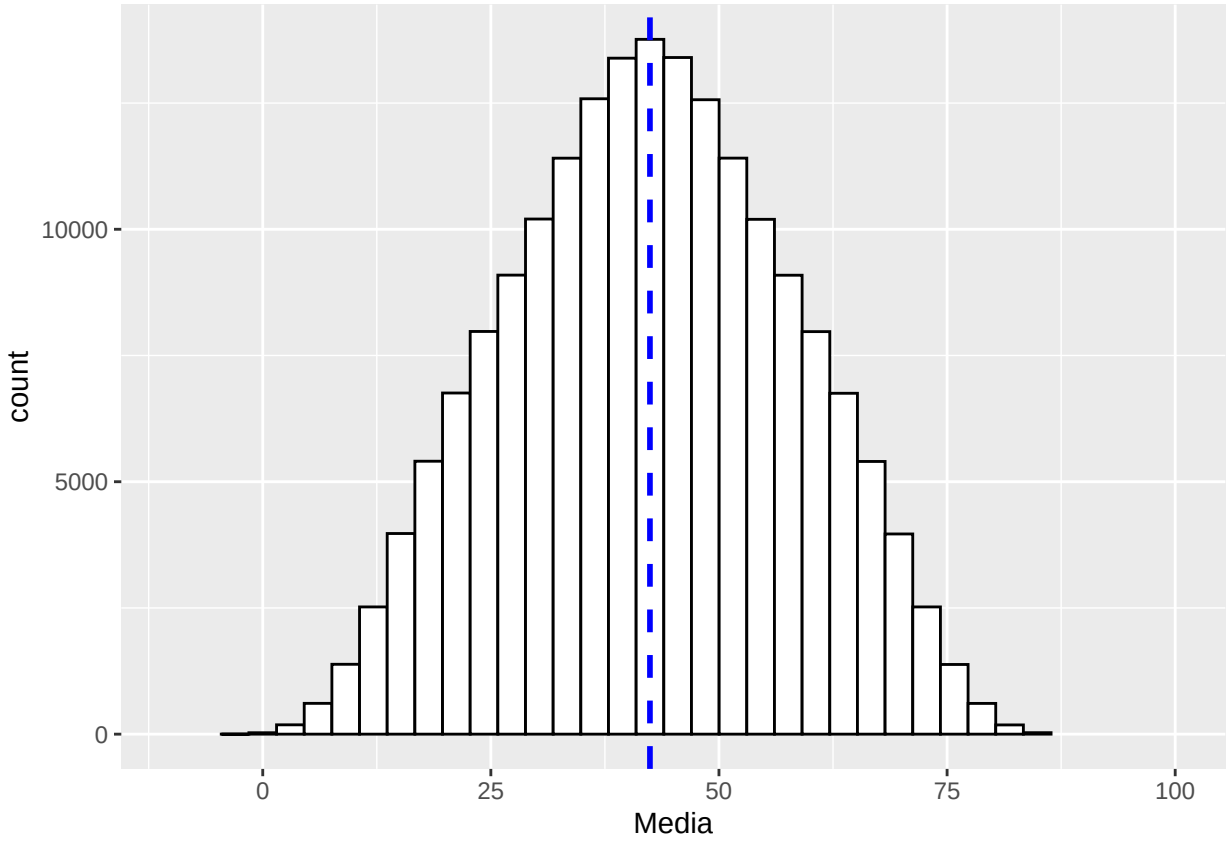
```r
mediana_muestras
```

```
## [1] 30.81752
```

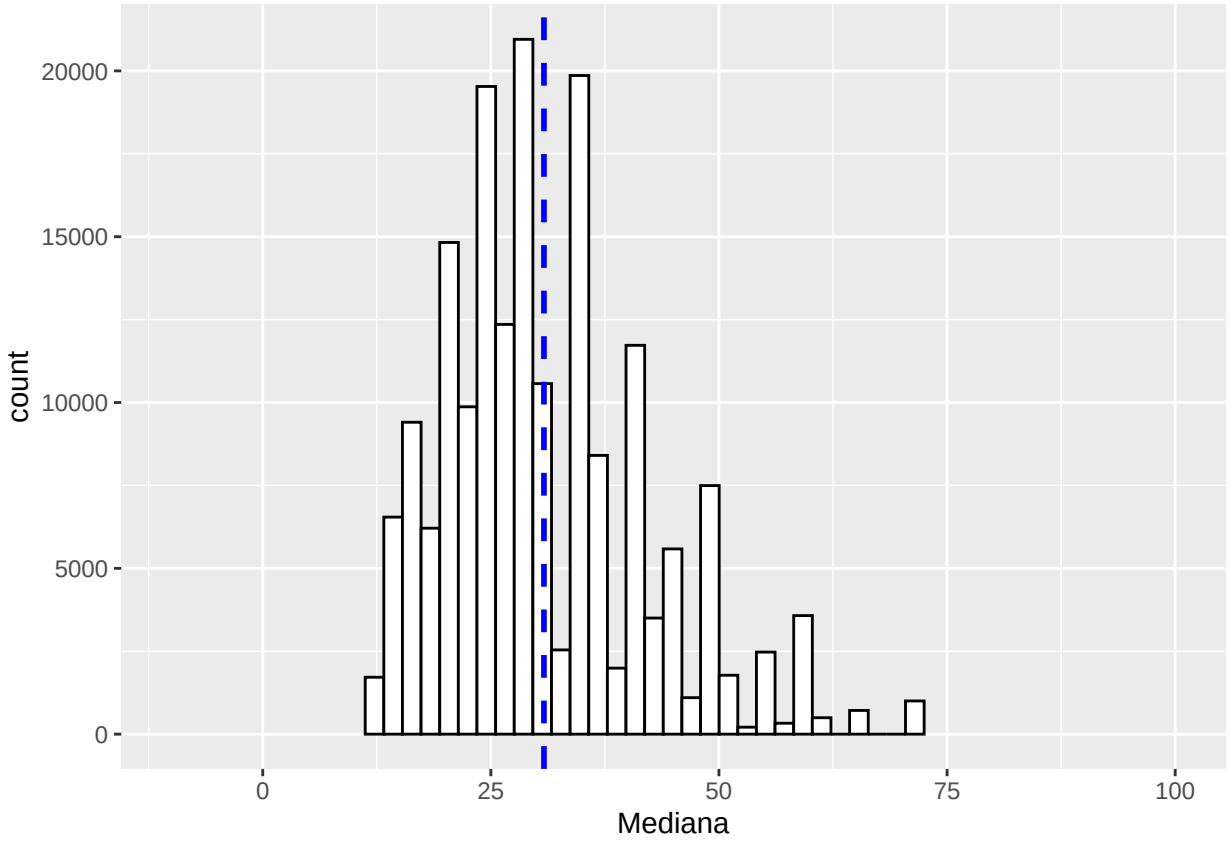
```r
mediaTrunc_muestras
```

```
## [1] 40.29746
```

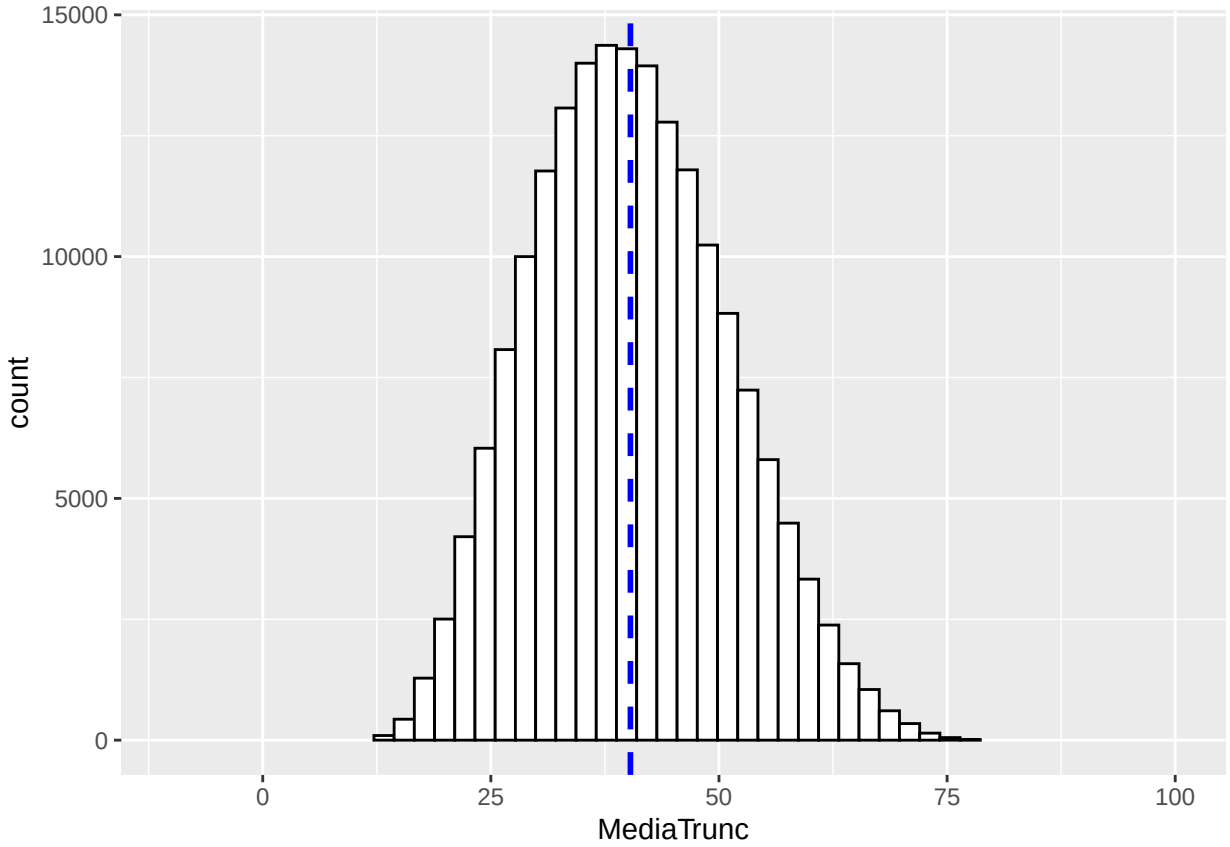
```r
#GRÁFICO DE LAS 3 ESTIMACIONES

#media
p <- ggplot(df_muestras, aes(x=Media)) +
  geom_histogram(bins=30, color="black", fill="white") +
  coord_cartesian(xlim = c(-10, 100))

p <- p+ geom_vline(aes(xintercept=mean(Media)),
                   color="blue", linetype="dashed", linewidth=1)
p
```




```
#mediana
p_median <- ggplot(df_muestras, aes(x=Mediana)) +
  geom_histogram(bins=30, color="black", fill="white") +
  coord_cartesian(xlim = c(-10, 100))

p_median <- p_median+ geom_vline(aes(xintercept=mean(Mediana)),
                  color="blue", linetype="dashed", linewidth=1)
p_median
```




```
#media truncada
p_mediaT <- ggplot(df_muestras, aes(x=MediaTrunc)) +
  geom_histogram(bins=30, color="black", fill="white") +
  coord_cartesian(xlim = c(-10, 100))

p_mediaT <- p_mediaT+ geom_vline(aes(xintercept=mean(MediaTrunc)),
                  color="blue", linetype="dashed", linewidth=1)
p_mediaT
```




```r
#cv y EMC de los 3 estimadores

cv_media <- sd(df_muestras$Media) / mean(df_muestras$Media)*100
cv_mediana <- sd(df_muestras$Mediana) / mean(df_muestras$Mediana)*100
cv_mediaT <- sd(df_muestras$MediaTrunc) / mean(df_muestras$MediaTrunc)*100

var_media<-var(df_muestras$Media)
var_mediana<-var(df_muestras$Mediana)
var_mediaT<-var(df_muestras$MediaTrunc)

sesgo_media<-media_poblacional - media_muestras
sesgo_mediana<-mediana_poblacional- mediana_muestras
sesgo_mediaT<-mediaTrunc_poblacional - mediaTrunc_muestras

emc_media <- var_media + sesgo_media^2
emc_mediana <- var_mediana + sesgo_mediana^2
emc_mediaT <- var_mediaT + sesgo_mediaT^2

resultados <- data.frame(
  Estimador = c("Media", "Mediana", "Media Truncada"),
  CV = c(cv_media, cv_mediana, cv_mediaT),
  EMC = c(emc_media, emc_mediana, emc_mediaT)
)

gt(resultados)
```

Estimador	CV	EMC
Media	36.24570	236.5710
Mediana	36.73640	134.1180
Media Truncada	26.55862	115.4421

```
#parece que el mejor estimador es la media truncada porque tiene menos varianza
```

Ejercicio III

```
# Lectura de las tablas
radios_sexo <- read_excel("cen2010_radios_sexo.xlsx")
radios_tipo <- read_excel("cen2010_radios_tipo.xlsx")

# Juntamos los archivos para unificarlos en un unico dataframe
radios_2010 = merge(radios_sexo, radios_tipo, by = "Codigo")

# Generamos nuevas variables en el dataframe que utilizaremos mas adelante

# Poblacion total en cada radio
radios_2010$Pob_radio <- radios_2010$Varon + radios_2010$Mujer

# Cantidad de viviendas en cada radio
radios_2010$Viv_radio <-  radios_2010$Casa +
  radios_2010$Rancho +
  radios_2010$Casilla +
  radios_2010$Departamento +
  radios_2010$Inquilinato +
  radios_2010$Hotel_pension

# Extraemos el codigo de provincia
radios_2010$prov <- floor(radios_2010$Codigo/10000000)


# Creamos etiqueta para las provincias
radios_2010$Provincia <- "X"
radios_2010 <- radios_2010 %>%
  mutate(Provincia = case_when(prov == 2 ~ 'CABA', prov == 6 ~ 'BsAs',
                               prov == 10 ~ 'Catamarca', prov == 14 ~ 'Cordoba',
                               prov == 18 ~ 'Corrientes', prov == 22 ~ 'Chaco',
                               prov == 26 ~ 'Chubut', prov == 30 ~ 'Entre Rios',
                               prov == 34 ~ 'Formosa', prov == 38 ~ 'Jujuy',
                               prov == 42 ~ 'La Pampa', prov == 46 ~ 'La Rioja',
                               prov == 50 ~ 'Mendoza', prov == 54 ~ 'Misiones',
                               prov == 58 ~ 'Neuquen', prov == 62 ~ 'Rio Negro',
                               prov == 66 ~ 'Salta', prov == 70 ~ 'San Juan',
                               prov == 74 ~ 'San Luis', prov == 78 ~ 'Santa Cruz',
                               prov == 82 ~ 'Santa Fe', prov == 86 ~ 'Santiago',
                               prov == 90 ~ 'Tucuman', prov == 94 ~ 'TdFuego'))
```

```r
# Eliminamos los radios sin viviendas
radios_2010 <- radios_2010[radios_2010$Viv_radio>0,]

N=nrow(radios_2010)
n=240

#parámetros

poblacion<- sum(radios_2010$Pob_radio)
hogares_casa<-sum(radios_2010$Casa)
hogares_rancho<-sum(radios_2010$Rancho)+sum(radios_2010$Casilla)
prop_rancho<-(sum(radios_2010$Rancho)+sum(radios_2010$Casilla))/sum(radios_2010$Viv_radio)

#CV
# Calculamos la varianza para la estimacion del total:
# En el MAS
# Var(y_media) = (1-n/N)*S^2/n
# Var(N*y_media) = N^2*(1-n/N)*S^2/n
P=prop_rancho
Q=1-prop_rancho
P+Q
```

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

```r
S2_pob <- var(radios_2010$Pob_radio)
S2_casa <- var(radios_2010$Casa)
S2_rancho <- var(radios_2010$Rancho + radios_2010$Casilla)
S2_prop <- P*Q

VarMAS_MediaPob <- (1-n/N)*S2_pob/n
VarMAS_MediaCasa <- (1-n/N)*S2_casa/n
VarMAS_MediaRancho <- (1-n/N)*S2_rancho/n


VarMAS_pob <- N^2*VarMAS_MediaPob   #Recordar: Var(k*X)= k^2*Var(X)
VarMAS_casa <- N^2*VarMAS_MediaCasa
VarMAS_rancho <- N^2*VarMAS_MediaRancho


# Calculamos el coeficiente de variacion
CVMAS_Pob <- 100*sqrt(VarMAS_pob)/poblacion
CVMAS_Casa <- 100*sqrt(VarMAS_casa)/hogares_casa
CVMAS_Rancho <- 100*sqrt(VarMAS_rancho)/hogares_rancho

ds_prop<-sqrt(P*Q)
CVMAS_Prop <- 100 *(ds_prop/prop_rancho) #revisar

CVMAS_Pob
```

```
## [1] 4.162589
```

```
CVMAS_Casa
```

```
## [1] 4.127404
```

```
CVMAS_Rancho
```

```
## [1] 14.6853
```

```
CVMAS_Prop #revisar
```

```
## [1] 537.2547
```

```
#el cv de la estimación de rancho o casilla es grande porque el N del universo es menor que en el caso
#el cv de la proporción de rancho o casilla es grande porque el valor de P es muy pequeño

#n=240*43

#para que el cv de la estimación de la población sea aproximadamente 2% n debe ser 1000
#para que el cv de la estimación de hogares tipo rancho o casilla sea aproximadamente 2% n debe ser 100

# Seleccionamos ahora una muestra aleatoria simple con R
s_mas <- sample(N,n, replace=FALSE)

muestra_radios <- radios_2010[s_mas,]

#estimaciones muestrales N*y_media

muestra_poblac<-N*mean(muestra_radios$Pob_radio)
muestra_casa<-N*mean(muestra_radios$Casa)
muestra_rancho<-N*mean(muestra_radios$Rancho+muestra_radios$Casilla)
muestra_prop<-(sum(muestra_radios$Rancho)+sum(muestra_radios$Casilla))/sum(muestra_radios$Viv_radio)

# Agregamos al data frame el factor de expansion
# (recordar que seleccione una muestra aleatoria simple de radios)
muestra_radios$pondera <- N/n


# Cantidad total de unidades en el marco de muestreo
# lo necesitare luego para survey
muestra_radios$fpc <- N

#junto rancho y casilla
muestra_radios$rancho_casilla <- muestra_radios$Rancho + muestra_radios$Casilla

# El objeto 'diseno' contiene toda la informacion que sera empleada
# para realizar las estimaciones.


diseno <- svydesign(id= ~1,weights=~pondera, data=muestra_radios, fpc=~fpc)
diseno
```

```
## Independent Sampling design
## svydesign(id = ~1, weights = ~pondera, data = muestra_radios,
##     fpc = ~fpc)
```

```
# (como es una muestra aleatoria simple ponemos fpc)

# Por ejemplo, si queremos extraer los pesos de un diseno podemos utilizar
pesos <- weights(diseno)


# total población
EstTotalPob <- survey :: svytotal(~Pob_radio, diseno, deff=TRUE, cv=TRUE, ci=TRUE)
EstTotalPob
```

```
##              total       SE DEff
## Pob_radio 38134961  1701675    1
```

```
# Puedo ahora extraer diferentes valores:
survey :: cv(EstTotalPob)      # -> coeficiente de variacion
```

```
##            Pob_radio
## Pob_radio 0.04462245
```

```
deff(EstTotalPob)    # -> efecto de diseno
```

```
## Pob_radio
##         1
```

```
SE(EstTotalPob)      # -> desvio estandar
```

```
##           Pob_radio
## Pob_radio   1701675
```

```
confint(EstTotalPob) # -> intervalor de confianza (por defecto 95%)
```

```
##              2.5 %   97.5 %
## Pob_radio 34799739 41470183
```

```
cv(EstTotalPob)
```

```
##            Pob_radio
## Pob_radio 0.04462245
```

```
# O pasar los resultados a un data frame
df_EstTotalPob <- as.data.frame(EstTotalPob)

# Quiero cambiar el nombre de las columnas
colnames(df_EstTotalPob) <- c("Estimacion", "SE", "deff")
```

```r
# Calculemos ahora el intervalo de confianza con un 90% de confianza
# (suponemos que el estimador es aprox normal)
df_EstTotalPob$Li <- df_EstTotalPob$Estimacion-1.64*df_EstTotalPob$SE
df_EstTotalPob$Ls <- df_EstTotalPob$Estimacion+1.64*df_EstTotalPob$SE


# Ahora calculo el CV del estimador
df_EstTotalPob$CV <-  100*df_EstTotalPob$SE/df_EstTotalPob$Estimacion


# total casas
EstTotalcasa <- survey :: svytotal(~Casa, diseno, deff=TRUE, cv=TRUE, ci=TRUE)
EstTotalcasa
```

```
##         total       SE DEff
## Casa 10126612   445668    1
```

```r
# Puedo ahora extraer diferentes valores:
survey :: cv(EstTotalcasa)      # -> coeficiente de variacion
```

```
##            Casa
## Casa 0.04400956
```

```r
deff(EstTotalcasa)    # -> efecto de diseno
```

```
## Casa
##    1
```

```r
SE(EstTotalcasa)      # -> desvio estandar
```

```
##          Casa
## Casa 445667.8
```

```r
confint(EstTotalcasa) # -> intervalor de confianza (por defecto 95%)
```

```
##        2.5 %   97.5 %
## Casa 9253120 11000105
```

```r
cv(EstTotalcasa)
```

```
##            Casa
## Casa 0.04400956
```

```r
# O pasar los resultados a un data frame
df_EstTotalcasa <- as.data.frame(EstTotalcasa)

# Quiero cambiar el nombre de las columnas
colnames(df_EstTotalcasa) <- c("Estimacion", "SE", "deff")
```

```r
# Calculemos ahora el intervalo de confianza con un 90% de confianza
# (suponemos que el estimador es aprox normal)
df_EstTotalcasa$Li <- df_EstTotalcasa$Estimacion-1.64*df_EstTotalPob$SE
df_EstTotalcasa$Ls <- df_EstTotalcasa$Estimacion+1.64*df_EstTotalPob$SE


# Ahora calculo el CV del estimador
df_EstTotalcasa$CV <-  100*df_EstTotalcasa$SE/df_EstTotalPob$Estimacion



# total rancho y casilla
EstTotalrancho <- survey :: svytotal(~rancho_casilla, diseno, deff=TRUE, cv=TRUE, ci=TRUE)
EstTotalrancho
```

```
##                  total     SE DEff
## rancho_casilla 506243  66488    1
```

```r
# Puedo ahora extraer diferentes valores:
survey :: cv(EstTotalrancho)      # -> coeficiente de variacion
```

```
##                 rancho_casilla
## rancho_casilla      0.1313365
```

```r
deff(EstTotalrancho)    # -> efecto de diseno
```

```
## rancho_casilla
##              1
```

```r
SE(EstTotalrancho)      # -> desvio estandar
```

```
##                 rancho_casilla
## rancho_casilla       66488.23
```

```r
confint(EstTotalrancho) # -> intervalor de confianza (por defecto 95%)
```

```
##                    2.5 %   97.5 %
## rancho_casilla 375928.8 636557.9
```

```r
cv(EstTotalrancho)
```

```
##                 rancho_casilla
## rancho_casilla      0.1313365
```

```r
# O pasar los resultados a un data frame
df_EstTotalrancho <- as.data.frame(EstTotalrancho)

# Quiero cambiar el nombre de las columnas
colnames(df_EstTotalrancho) <- c("Estimacion", "SE", "deff")
```

```r
# Calculemos ahora el intervalo de confianza con un 90% de confianza
# (suponemos que el estimador es aprox normal)
df_EstTotalrancho$Li <- df_EstTotalrancho$Estimacion-1.64*df_EstTotalPob$SE
df_EstTotalrancho$Ls <- df_EstTotalrancho$Estimacion+1.64*df_EstTotalPob$SE


# Ahora calculo el CV del estimador
df_EstTotalrancho$CV <-  100*df_EstTotalrancho$SE/df_EstTotalrancho$Estimacion


# proporcion
Estproporcion <- survey::svyratio(~rancho_casilla,~Viv_radio,  design = diseno, deff = TRUE, cv = TRUE,

Estproporcion
```

```
## Ratio estimator: svyratio.survey.design2(~rancho_casilla, ~Viv_radio, design = diseno,
##     deff = TRUE, cv = TRUE, ci = TRUE)
## Ratios=
##                 Viv_radio
## rancho_casilla 0.03763484
## SEs=
##                  Viv_radio
## rancho_casilla 0.004963442
```

```r
# Puedo ahora extraer diferentes valores:
estimador <- coef(Estproporcion)  # Estimador de la proporción
error_estandar <- SE(Estproporcion)  # Error estándar
cv_ <- cv(Estproporcion)  # Coeficiente de variación
intervalo_confianza <- confint(Estproporcion)  # Intervalo de confianza

#IC 90%

IC90Li <- estimador-1.64*error_estandar
IC90Ls <- estimador+1.64*error_estandar


# Ahora calculo el CV del estimador
CVe <-  100*error_estandar/estimador
CVe
```

```
## rancho_casilla/Viv_radio
##                 13.18842
```

Ejercicio IV

Estimación, encuestando en su totalidad una Muestra Sistemática de n=240 radios censales para Total de población, Total de hogares que habitan en viviendas tipo Casa y Total de hogares que habitan en viviendas rancho/ casilla

Estimación para Población

```
#Creo una nueva tabla para este ejercicio con la base que ya armamos en el Ejercicio 3.

radios_2010E4 <- radios_2010

N <- nrow(radios_2010E4)

# Tamano de la muestra
n <- 240

# Intervalo de selección
I <- floor(N/n)
print(I)
```

```
## [1] 218
```

```
# Parametro poblacional
ParametroPobE4 <- sum(radios_2010E4$Pob_radio)
print(ParametroPobE4)
```

```
## [1] 40115211
```

```
# Ordenamiento del marco de muestreo
radios_2010E4 <- radios_2010E4[order(radios_2010E4$Codigo),]
radios_2010E4$aleatorio <- runif(nrow(radios_2010E4),0,1)
radios_2010E4 <- radios_2010E4[order(radios_2010E4$Viv_radio),]
radios_2010E4 <- radios_2010E4[order(radios_2010E4$Pob_radio),]
radios_2010E4 <- radios_2010E4[order(radios_2010E4$aleatorio),]
```

Definiremos el arranque aleatorio, generando un número aleatorio entre 1 y el intervalo de selección I, que en nuestro caso es 218.

```
#aa <- sample(1:I, 1)
aa = 75
print(aa)
```

```
## [1] 75
```

El resultado de aa es 75. Le pondremos un # a la función dado que cada vez que se ejecuta cambiará el valor de aa

```
#Seleccionaremos una muestra sistemática de los datos en radios_2010E4 utilizando el arranque aleatorio


s = radios_2010E4[ seq(aa,N,I), ]
```

Calcularemos una estimación del total poblacional y su error relativo para evaluar la precisión de la muestra sistemática.

```r
#Primero realizaremos la estimación del total poblacional usando el método de Horvitz-Thompson, un esti

estim <- I*sum(s$Pob_radio)
print(estim)
```

```
## [1] 40602936
```

```r
error_rel <- 100*(ParametroPobE4 - estim) /ParametroPobE4
print(error_rel)
```

```
## [1] -1.215811
```

La estimación de la población usando una muestra de 240 radios censales es muy cercana al parámetro poblacional ParametroPobE4 (40115211), lo cual sugiere que la muestra debería ser buena.El error relativo es muy bajo, por lo que la muestra puede proveer una estimación precisa.

Estimación para viviendas tipo Casa

```r
#Parámetro poblacional para viviendas tipo Casa:
ParametroCasaE4 <- sum(radios_2010E4$Casa)
print(ParametroCasaE4)
```

```
## [1] 10620866
```

```r
#Estimación para la muestra:
estimCasaE4 <- I * sum(s$Casa)
print(estimCasaE4)
```

```
## [1] 10555996
```

```r
#Error relativo
error_relCasaE4 <- 100 * (ParametroCasaE4 - estimCasaE4) / ParametroCasaE4
print(error_relCasaE4)
```

```
## [1] 0.6107788
```

El valor de la estimación calculada a partir de la muestra se aproxima bastante al valor real de la población, por lo cual el muestreo es representativo y el tamaño de la muestra se puede interpretar como adecuado. El error relativo es muy bajo, por lo que la estimación es bastante cercana al parámetro poblacional real.

Estimación para viviendas tipo Rancho o Casilla

```r
# Crear la columna Rancho_Casilla
radios_2010E4$Rancho_Casilla <- radios_2010E4$Rancho + radios_2010E4$Casilla

# Seleccionar la muestra sistemática
```

```r
s = radios_2010E4[seq(aa, N, I), ]

# Parámetro poblacional
ParametroRanchoCasillaE4 <- sum(radios_2010E4$Rancho_Casilla)
print(ParametroRanchoCasillaE4)
```

```
## [1] 461725
```

```r
# Estimación para la muestra
estimRanchoCasillaE4 <- I * sum(s$Rancho_Casilla)
print(estimRanchoCasillaE4)
```

```
## [1] 384988
```

```r
# Error relativo
error_relRanchoCasillaE4 <- 100 * (ParametroRanchoCasillaE4 - estimRanchoCasillaE4) / ParametroRanchoCas
print(error_relRanchoCasillaE4)
```

```
## [1] 16.61963
```

La estimación de 426408 es cercano al parametro 461725, lo cual indica que el muestreo sistemático es preciso para estimar el total de viviendas tipo rancho o casilla.El error relativo es muy bajo siendo un buen indicador de precisión en el muestreo

Compararemos dos estrategias utilizando como estimador la media muestral

1. Muestreo sistemático, ordenando la tabla por Provincia-Total de viviendas del radio

Primero indicamos aquí nuevamente los parámetros que obtuvimos en el punto anterior:

```r
print(ParametroPobE4)
```

```
## [1] 40115211
```

```r
print(ParametroCasaE4)
```

```
## [1] 10620866
```

```r
print(ParametroRanchoCasillaE4)
```

```
## [1] 461725
```

Comenzamos con la estrategia 1 que implica Orden por “Provincia-Total de viviendas del radio”

```r
# Ordenar por Provincia y Total de viviendas del radio
Estrategia1E4 <- radios_2010E4[order(radios_2010E4$Provincia, radios_2010E4$Viv_radio), ]

# Tamaño de la muestra y cálculo del intervalo
n <- 240
N <- nrow(Estrategia1E4)
I <- floor(N / n)
print(I)
```

```
## [1] 218
```

```
# Arranque aleatorio
# aa <- sample(1:I, 1)
# La función arroja un aa de 191. Le colocamos # dado que sino arrojará un aa diferente en cada ejecuci
aa = 191
print(aa)
```

```
## [1] 191
```

```
# Selección sistemática
muestraE1 <- Estrategia1E4[seq(aa, N, by = I), ]

# Estimaciones para la muestra en Estrategia 1
estimPobE1 <- I * mean(muestraE1$Pob_radio)
estimCasaE1 <- I * mean(muestraE1$Casa)
estimRancho_CasillaE1 <- I * mean(muestraE1$Rancho + muestraE1$Casilla)

print(estimPobE1)
```

```
## [1] 172396.2
```

```
print(estimCasaE1)
```

```
## [1] 44291.24
```

```
print(estimRancho_CasillaE1)
```

```
## [1] 1963.817
```

Ahora revisamos el error relativo para Estrategia 1

```
error_relPobE1 <- 100 * abs(ParametroPobE4 - estimPobE1) / ParametroPobE4
error_relCasaE1 <- 100 * abs(ParametroCasaE4 - estimCasaE1) / ParametroCasaE4
error_relRancho_CasillaE1 <- 100 * abs(ParametroRanchoCasillaE4 - estimRancho_CasillaE1) / ParametroRan

print(error_relPobE1)
```

```
## [1] 99.57025
```

```
print(error_relCasaE1)
```

```
## [1] 99.58298
```

```
print(error_relRancho_CasillaE1)
```

```
## [1] 99.57468
```

Los errores relativos son muy altos (cercanos al 100%), lo que indica que la estimación está muy alejada del valor poblacional, pudiendo interpretar que el ordenamiento esté afectando la representatividad.

2. Muestreo sistemático, ordenando la tabla por un número pseudo aleatorio

```
# Ordenar por el número aleatorio
Estrategia2E4 <- radios_2010E4 [order(radios_2010E4$aleatorio), ]

# Selección sistemática
muestraE2 <- Estrategia2E4 [seq(aa, N, by = I), ]

# Estimaciones para la muestra en Estrategia 2
estimPobE2 <- I * mean(muestraE2$Pob_radio)
estimCasaE2 <- I * mean(muestraE2$Casa)
estimRancho_CasillaE2 <- I * mean(muestraE2$Rancho + muestraE2$Casilla)

print(estimPobE2)
```

```
## [1] 169821.1
```

```
print(estimCasaE2)
```

```
## [1] 45332.19
```

```
print(estimRancho_CasillaE2)
```

```
## [1] 1575.958
```

```
error_relPobE2 <- 100 * abs(ParametroPobE4 - estimPobE2) / ParametroPobE4
error_relCasaE2 <- 100 * abs(ParametroCasaE4 - estimCasaE2) / ParametroCasaE4
error_relRancho_CasillaE2 <- 100 * abs(ParametroRanchoCasillaE4 - estimRancho_CasillaE2) / ParametroRanc

print(error_relPobE2)
```

```
## [1] 99.57667
```

```
print(error_relCasaE2)
```

```
## [1] 99.57318
```

```
print(error_relRancho_CasillaE2)
```

```
## [1] 99.65868
```

Al parecer ninguna de las estrategias estaría dando resultados dado que los errores relativos son muy altos. La estrategia de muestreo sistemático puede que no sea la más adecuada para este caso.

Hallar CV, deff, sesgo relativo y EMC de cada estrategia, seleccionando todas las muestas posibles

Para evaluar cada estrategia y medir su eficiencia y precisión, podemos calcular el coeficiente de variación (CV), el efecto del diseño (deff), el sesgo relativo y el error medio cuadrático (EMC).

Para ello, calcularemos las I estimaciones posibles, una para cada arranque aleatorio. Definiremos una funcion que seleccione una muestra sistematica, con el arranque aleatorio como variable independiente y devuelva la estimacion

Primero, definimos las funciones para seleccionar la muestra sistemática y para calcular los estimadores necesarios para todas las posibles muestras.

Estrategia 1

```
# Tamaño de la población y de la muestra
N <- nrow(Estrategia1E4)  # total de radios censales
n <- 240                   # tamaño de la muestra
I <- floor(N / n)          # intervalo de selección

# Valor verdadero del parámetro poblacional
parametro_poblacionalEst1 <- sum(Estrategia1E4$Pob_radio)

# Definimos la función de estimación sistemática
estim_sistematico <- function(aa) {
  s = Estrategia1E4[seq(aa, N, I), ]
  estim <- I * sum(s$Pob_radio)
  return(c(estim))
}

estimacionEst1 <- estim_sistematico(2)

print(parametro_poblacionalEst1)
```

```
## [1] 40115211
```

```
print(estimacionEst1)
```

```
## [1] 39802876
```

```
# Calculamos las I estimaciones posibles

lista_arranques <- 1:I

lista_estimaciones <- lapply(lista_arranques, estim_sistematico)

df_estim <- data.frame(matrix(unlist(lista_estimaciones),ncol=1, byrow=TRUE ) )


colnames(df_estim) <- c("Estimacion")

EsperanzaE4   <-  mean(df_estim$Estimacion)
```

```r
SesgoE4    <- EsperanzaE4 - parametro_poblacionalEst1
VarianzaE4 <-  var(df_estim$Estimacion)*(N-1)/N
DSE4       <- sqrt(VarianzaE4)
CVestimsE4        <- 100*DSE4/parametro_poblacionalEst1


print(EsperanzaE4)
```

```
## [1] 40115211
```

```r
print(SesgoE4)
```

```
## [1] 0
```

```r
print(VarianzaE4)
```

```
## [1] 1.032047e+12
```

```r
print(DSE4)
```

```
## [1] 1015897
```

```r
print(CVestimsE4)
```

```
## [1] 2.532449
```

- La esperanza, es decir, la estimación promedio de la población utilizando todas las muestras posible toma el mismo valor que el parametro poblacional, lo cual sugiere que el estimador es insesgado para esta estrategia.
- El sesgo, que es la diferencia entre la esperanza de la estimación y el valor verdadero, da cero, justo lo que se espera dado que el estimador es insesgado. El valor cero confirma que no hay desviación entre la media de las estimaciones y el valor poblacional.
- La varianza es la medida de la dispersión de las estimaciones.
- La desviación Estándar indica la variabilidad de las estimaciones alrededor de la media.
- Finalmente el Coeficiente de Variación (CV) ayuda a comparar la precisión de la estimación en relación con el valor poblacional real, y el valor que toma (4,05) es bajo CV bajo (menor a 10%) por lo cual indica una buena precisión.

Estrategia 2

```r
# Tamaño de la población y de la muestra
N <- nrow(Estrategia2E4)  # total de radios censales
n <- 240                   # tamaño de la muestra
I <- floor(N / n)          # intervalo de selección
```

```r
# Valor verdadero del parámetro poblacional
parametro_poblacionalEst2 <- sum(Estrategia2E4$Pob_radio)

# Definimos la función de estimación sistemática
estim_sistematico2 <- function(aa) {
  s = Estrategia2E4[seq(aa, N, I), ]
  estim2 <- I * sum(s$Pob_radio)
  return(c(estim2))
}

estimacionEst2 <- estim_sistematico2(2)

print(parametro_poblacionalEst2)
```

```
## [1] 40115211
```

```r
print(estimacionEst2)
```

```
## [1] 39316518
```

```r
# Calculamos las I estimaciones posibles

lista_arranques2 <- 1:I

lista_estimaciones2 <- lapply(lista_arranques2, estim_sistematico2)

df_estim2 <- data.frame(matrix(unlist(lista_estimaciones2),ncol=1, byrow=TRUE ) )


colnames(df_estim2) <- c("Estimacion2")

EsperanzaE4.2   <-  mean(df_estim2$Estimacion2)
SesgoE4.2    <- EsperanzaE4.2 - parametro_poblacionalEst2
VarianzaE4.2 <-  var(df_estim2$Estimacion2)*(N-1)/N
DSE4.2       <- sqrt(VarianzaE4.2)
CVestimsE4.2        <- 100*DSE4.2/parametro_poblacionalEst2


EsperanzaE4.2
```

```
## [1] 40115211
```

```r
SesgoE4.2
```

```
## [1] 0
```

```r
VarianzaE4.2
```

```
## [1] 2.843253e+12
```

Estrategia	Esperanza	Sesgo	Varianza	Desviacion_Estandar	Coeficiente_Variacion
Estrategia 1	40115211	0	1.032047e+12	1015897	2.532449
Estrategia 2	40115211	0	2.843253e+12	1686195	4.203380

```
DSE4.2
```

```
## [1] 1686195
```

```
CVestimsE4.2
```

```
## [1] 4.20338
```

Comparación de estrategias

```
tabla_comparacionEstrategiasE4 <- data.frame(
  Estrategia = c("Estrategia 1", "Estrategia 2"),
  Esperanza = c(EsperanzaE4, EsperanzaE4.2),
  Sesgo = c(SesgoE4, SesgoE4.2),
  Varianza = c(VarianzaE4, VarianzaE4.2),
  Desviacion_Estandar = c(DSE4, DSE4.2),
  Coeficiente_Variacion = c(CVestimsE4, CVestimsE4.2)
)

# Mostrar la tabla de comparación
gt(tabla_comparacionEstrategiasE4)
```

El sesgo es cero en ambas estrategias lo cual es un indicio de que ambas estrategias son estimadores insesgados del total poblacional.

La varianza de la Estrategia 2 es mayor que la de la Estrategia 1, lo que indica que el ordenamiento aleatorio puede introducir más variabilidad en las estimaciones.

El CV de ambas estrategias es igual, lo cual sugiere que las dos estrategias tienen un buen nivel de precisión.

La Estrategia 1, es decir ordenar por Provincia y Total de viviendas, presenta menos variabilidad, por lo cual podría ser la estrategia a elegir si queremos reducir la variabilidad en las estimaciones.

Ejercicio V (continuación del ejercicio IV)

Ejercicio VI

Ejercicio VII

Ejercicio VIII