```r
library(tidyverse)
library(knitr)
```

2. Simulación para el cálculo de tamaños de muestra

En el conteo rápido del estado de Guanajuato, se calculó el tamaño de muestra fijando como objetivo que los intervalos del 95% de confianza tuvieran una longitud máxima de 2 puntos porcentuales para todos los candidatos. En este ejercicio calcularás el tamaño de muestra mínimo que cumpla con el objetivo usando 3 diseños de muestreo distintos: 1) muestreo aleatorio simple (MAS), 2) estratificando con distrito local y 3) estratificando con distrito federal.

Utilizarás simulación y los resultados de las elecciones de gobernador correspondientes al 2012.

```r
# Cargamos la base de datos gto_2012
gto_2012 <- read.csv("data/gto_2012.csv", stringsAsFactors = FALSE)
```

En el caso de **MAS**, para cada tamaño de muestra $n = 50, 100, 200, 300, 400, 500, 600, 700$:

i. Simula una muestra aleatoria de tamaño n.

ii. Calcula el estimador de razón (correspondiente a muestreo aleatorio simple) para cada candidato:

$$\hat{p} = \frac{\sum_i Y_i}{\sum_i X_i}$$

$$\hat{p} = \frac{\sum_h \frac{N_h}{n_h} \sum_i Y_{hi}}{\sum_h \frac{N_h}{n_h} \sum_i X_{hi}}$$

donde:

- $\hat{p}$ es la estimación de la proporción de votos que recibió el candidato en la elección.
- Y_i es el número total de votos que recibió el candidato en la i-ésima casilla.
- X_i es el número total de votos en la i-ésima casilla.

iii. Repite los pasos i y ii 1000 veces para estimar el error estándar para una muestra de tamaño n.

Muestreo Aleatorio Simple

```r
# Definimos la semilla
set.seed(4556788)
# Definimos el vector n que contiene los tamaños de muestra
n   <- c(50,100,200,300,400,500,600,700)
# Juntamos los partidos en una sola variable recordando los principios
# de los datos limpios
y_i <- gto_2012 %>% gather(key="partido", value = "votos",
                           pri_pvem, pan_na, prd, pt, mc, otros) %>%
  group_by(partido) %>%
  # Agregamos la variable de proporción de votos para cada partido
  summarise("prop" = sum(votos)/sum(total))

# Nos quedamos únicamente con las proporciones por partido para
# poder realizar el muestreo aleatorio simple
p_i <- y_i %>%select("prop") %>% unlist()

# Definimos una función para las simulaciones
```

```r
sim_p_hat <- function(n, p, n_sims = 1000){
  # Simulamos de una multinomial con n el tamaño de
  # muestra y p las proporciones de votos para
  # partido
  sim_muestra <- rmultinom(n_sims, n, p)
  # Obtenemos el estimador de razón para este
  # tipo de muestreo
  p_razon     <- sim_muestra/n
  # Obtenemos el error estándar para cada candidato
  SE_cand     <- c()
  for(i in 1:nrow(sim_muestra)){
    SE_cand <- c(SE_cand, sd(p_razon[i,])/sqrt(n))
  }
  return(SE_cand)
}

# Simulamos los errores estándar dadas las proporciones
# observadas
SE <- sim_p_hat(n[1], p_i)

# Juntamos los errores estándar para los distintos tamaños de muestra
for (j in 2:length(n)) {
  SE <- cbind(SE, sim_p_hat(n[j], p_i))
}

# Le damos formato para generar una tabla con los SE
colnames(SE) <- as.character(n)
IClength <- 2*1.96*SE
SE <- formatC(SE, digits = 4, format = "f")
SE
```

```
##      50       100      200      300      400      500      600
## [1,] "0.0019" "0.0010" "0.0005" "0.0003" "0.0002" "0.0002" "0.0002"
## [2,] "0.0037" "0.0019" "0.0010" "0.0006" "0.0005" "0.0004" "0.0003"
## [3,] "0.0095" "0.0048" "0.0025" "0.0016" "0.0012" "0.0010" "0.0008"
## [4,] "0.0043" "0.0022" "0.0011" "0.0008" "0.0005" "0.0004" "0.0004"
## [5,] "0.0093" "0.0049" "0.0024" "0.0017" "0.0012" "0.0010" "0.0008"
## [6,] "0.0022" "0.0011" "0.0006" "0.0004" "0.0003" "0.0002" "0.0002"
##      700
## [1,] "0.0001"
## [2,] "0.0003"
## [3,] "0.0007"
## [4,] "0.0003"
## [5,] "0.0007"
## [6,] "0.0002"
```

```r
# Le damos formato para generar una tabla con la longitud de los ICs
IClength <- formatC(IClength, digits = 4, format = "f")
IClength
```

```
##      50       100      200      300      400      500      600
## [1,] "0.0076" "0.0038" "0.0020" "0.0012" "0.0010" "0.0008" "0.0006"
## [2,] "0.0145" "0.0075" "0.0037" "0.0024" "0.0018" "0.0015" "0.0012"
## [3,] "0.0374" "0.0189" "0.0100" "0.0063" "0.0049" "0.0040" "0.0032"
## [4,] "0.0170" "0.0085" "0.0044" "0.0029" "0.0021" "0.0017" "0.0014"
```

```
## [5,] "0.0364" "0.0192" "0.0096" "0.0065" "0.0048" "0.0039" "0.0032"
## [6,] "0.0085" "0.0042" "0.0023" "0.0015" "0.0011" "0.0008" "0.0007"
##      700
## [1,] "0.0006"
## [2,] "0.0011"
## [3,] "0.0029"
## [4,] "0.0013"
## [5,] "0.0027"
## [6,] "0.0006"
```

Para muestro aleatorio simple y dado el estimador de razón para cada candidato $\hat{p}_i$ se tiene que el tamaño de muestra tal que la longitud del intervalo de confianza sea menor a dos puntos porcentuales es al menos 400.

Para cada posible **estratificación** (`distrito_fed_17` y `distrito_loc_17`) y tamaño de muestra $n = 50, 100, 200, 300, 400, 500, 600, 700$:

i. Simula una muestra estratificada de tamaño n, donde el tamaño de muestra en cada estrato se asigna proporcional al tamaño del estrato, esto es, sea N_h el número de casillas en el h-ésimo estrato, entonces para el estrato h el número de casillas en la muestra será:

$$n_h = N_h \cdot \frac{n}{\sum_j N_j}$$

ii. Calcula el estimador de razón combinado (correspondiente a muestreo estratificado) para cada candidato:

$$\hat{p} = \frac{\sum_h \frac{N_h}{n_h} \sum_i Y_{hi}}{\sum_h \frac{N_h}{n_h} \sum_i X_{hi}}$$

donde:

- $\hat{p}$ es la estimación de la proporción de votos que recibió el candidato en la elección.
- Y_{hi} es el número total de votos que recibió el candidato en la i-ésima casillas, que pertence al h-ésimo estrato.
- X_{hi} es el número total de votos en la i-ésima casilla, que pertence al h-ésimo estrato.
- N_h es el número total de casillas en el h-ésimo estrato.
- n_h es el número de casillas del h-ésimo estrato que se seleccionaron en la muestra.

iii. Repite los pasos i y ii 1000 veces para estimar el error estándar para una muestra de tamaño n.

Muestreo estratificado

```
# Nuevamente seguimos los principios de los datos limpios
# para definir una única variable con el partido
# Creamos dos variables "estrato_local" y "estrato_federal"
# para poder tener un mejor manejo de la base.

base2 <- gto_2012 %>%  gather(key="partido", value = "votos",
                              pri_pvem, pan_na, prd, pt, mc, otros) %>%
  mutate("estrato_local" = distrito_loc_17) %>%
  mutate("estrato_federal" = distrito_fed_17)
```

Muestreo estratificado por distrito local

```r
# Creamos una función que hace un muestreo de las casillas por estrato
Muestrasim <- function(base,i, nh){
  Muestra <- base %>% filter(estrato_local==estrato_local[i])
  sample_n( Muestra, nh[i],replace = TRUE)
}
```

```r
# Creamos una función que calcula el estimador de razon para el estrato local
# y tamaño de muestra n
propRazon <- function(n, base){

  # Se cuentan las casillas por estrato y se dividen entre
  # el tamaño de muestra, tomamos el techo, ya que no se pueden
  # tomar fracciones de las casillas
  Nh0 <- base %>%  group_by(estrato_local) %>%
    summarise(Nh=n()) %>%
    mutate("nh" = ceiling(Nh*n/sum(Nh)))

  nh <- unlist(Nh0$nh)

  # Muestreamos las casillas que utilizaremos
  baseMuestra  <- Muestrasim(base,1,nh)

  # Se repite para todos los estratos
  for(i in 2:nrow(Nh0)){
    baseMuestra <- rbind(baseMuestra, Muestrasim(base,i,nh))
  }

  # Obtenemos el numerador del estimador de razón
  Nh <- baseMuestra %>%  group_by(estrato_local) %>%
    summarise(Nh=n()) %>%
    mutate("nh" = ceiling(Nh*n/sum(Nh)))

  numerador <- baseMuestra %>% group_by(estrato_local, partido) %>%
    summarise(Votos_estrato = sum(votos)) %>%
    left_join(Nh,"estrato_local") %>%
    group_by(partido) %>%
    summarise(numerador = sum(Votos_estrato*Nh/nh))

  # Obtenemos el denominador del estimador de razón
  denominador <- baseMuestra %>% group_by(estrato_local) %>%
    summarise(votos_Estrato=sum(votos)) %>%
    left_join(Nh, "estrato_local") %>%
    ungroup() %>%
    summarise(denom=sum(votos_Estrato*Nh/nh))
  # Calculamos el estimador de razón
  prop <- numerador$numerador/denominador$denom
  return(prop)
}
```

```r
# Simulamos el error estándar para el estrato local
Sim_cand_estrato_Loc <- function(i){
  # Generamos 1000 simulaciones del estimador de razón para el tamaño de muestra i
  Propsim <- rerun(1000, propRazon(i, base2)) %>% unlist() %>%
    matrix(ncol = 1000, byrow = FALSE)
```

```
  # Para cada candidato obtenemos el error estándar en el
  # estrato local

  SE_cand     <- c()

  for(j in 1:nrow(Propsim)){
    SE_cand <- c(SE_cand, sd(Propsim[j,])/sqrt(i))
  }
  return(SE_cand)
}
```

```
# Hacemos las simulaciones para cada estrato
SE_cand_estrato_Loc <- Sim_cand_estrato_Loc(n[1])

for (j in 2:length(n)) {
  SE_cand_estrato_Loc  <- cbind(SE_cand_estrato_Loc, Sim_cand_estrato_Loc(n[j]))
}

# Se le da formato a la tabla de error estándar por estrato local
colnames(SE_cand_estrato_Loc) <- as.character(n)
IClength_estrato_Loc <- 2* 1.96*SE_cand_estrato_Loc

SE_cand_estrato_Loc <- formatC(SE_cand_estrato_Loc, digits = 4, format = "f")
SE_cand_estrato_Loc
```

```
##      50       100      200      300      400      500      600
## [1,] "0.0004" "0.0002" "0.0001" "0.0001" "0.0000" "0.0000" "0.0000"
## [2,] "0.0026" "0.0014" "0.0007" "0.0004" "0.0004" "0.0003" "0.0002"
## [3,] "0.0148" "0.0078" "0.0040" "0.0028" "0.0021" "0.0017" "0.0014"
## [4,] "0.0029" "0.0015" "0.0008" "0.0005" "0.0004" "0.0003" "0.0003"
## [5,] "0.0143" "0.0076" "0.0039" "0.0028" "0.0021" "0.0016" "0.0014"
## [6,] "0.0012" "0.0006" "0.0003" "0.0002" "0.0002" "0.0001" "0.0001"
##      700
## [1,] "0.0000"
## [2,] "0.0002"
## [3,] "0.0011"
## [4,] "0.0002"
## [5,] "0.0011"
## [6,] "0.0001"
```

```
# Se le da formato a la tabla de tamaño del IC por estrato local
IClength_estrato_Loc <- formatC(IClength_estrato_Loc, digits = 4, format = "f")
IClength_estrato_Loc
```

```
##      50       100      200      300      400      500      600
## [1,] "0.0014" "0.0007" "0.0004" "0.0002" "0.0002" "0.0001" "0.0001"
## [2,] "0.0102" "0.0055" "0.0027" "0.0017" "0.0014" "0.0011" "0.0009"
## [3,] "0.0580" "0.0305" "0.0156" "0.0109" "0.0082" "0.0066" "0.0053"
## [4,] "0.0115" "0.0058" "0.0031" "0.0020" "0.0015" "0.0012" "0.0010"
## [5,] "0.0561" "0.0299" "0.0153" "0.0108" "0.0081" "0.0064" "0.0054"
## [6,] "0.0048" "0.0024" "0.0013" "0.0008" "0.0006" "0.0005" "0.0004"
##      700
## [1,] "0.0001"
## [2,] "0.0007"
```

```
## [3,] "0.0045"
## [4,] "0.0009"
## [5,] "0.0044"
## [6,] "0.0004"
```

Muestreo estratificado por distrito federal

```r
# Creamos un código para muestrear casillas del estrato federal
MuestrasimFed <- function(base,i, nh){
  Muestra <- base %>% filter(estrato_federal==estrato_federal[i])
  sample_n( Muestra, nh[i],replace = TRUE)
}
```

```r
# Creamos una función para calcular el estimador de razón
# para el estrato federal y tamaño de muestra n
propRazonFed <- function(n, base){
  # Se cuentan las casillas por estrato y se dividen entre
  # el tamaño de muestra, tomamos el techo, ya que no se pueden
  # tomar fracciones de las casillas
  Nh0 <- base %>%  group_by(estrato_federal) %>%
    summarise(Nh=n()) %>%
    mutate("nh" = ceiling(Nh*n/sum(Nh)))

  nh <- unlist(Nh0$nh)

  # Muestreamos las casillas que utilizaremos
  baseMuestra  <- MuestrasimFed(base,1,nh)

  # Se repite para todos los estratos
  for(i in 2:nrow(Nh0)){
    baseMuestra <- rbind(baseMuestra, MuestrasimFed(base,i,nh))
  }

  # Obtenemos el numerador del estimador de razón
  Nh <- baseMuestra %>%  group_by(estrato_federal) %>%
    summarise(Nh=n()) %>%
    mutate("nh" = ceiling(Nh*n/sum(Nh)))

  numerador <- baseMuestra %>% group_by(estrato_federal, partido) %>%
    summarise(Votos_estrato = sum(votos)) %>%
    left_join(Nh,"estrato_federal") %>%
    group_by(partido) %>%
    summarise(numerador = sum(Votos_estrato*Nh/nh))

  # Obtenemos el denominador para el estimador de razón
  denominador <- baseMuestra %>% group_by(estrato_federal) %>%
    summarise(votos_Estrato=sum(votos)) %>%
    left_join(Nh, "estrato_federal") %>%
    ungroup() %>%
    summarise(denom=sum(votos_Estrato*Nh/nh))

  # Calculamos el estimador de razón
  prop <- numerador$numerador/denominador$denom
  return(prop)
```

```
}
```

```
# Simulamos el error estándar para el estrato federal
Sim_cand_estrato_Federal <- function(i){
  # Generamos 1000 simulaciones del estimador de razón para el tamaño de muestra i
  Propsim <- rerun(1000, propRazonFed(i, base2)) %>% unlist() %>%
    matrix(ncol = 1000, byrow = FALSE)

  SE_cand     <- c()

  # Para cada candidato obtenemos el error estándar en el
  # estrato local
  for(j in 1:nrow(Propsim)){
    SE_cand <- c(SE_cand, sd(Propsim[j,])/sqrt(i))
  }
  return(SE_cand)
}
```

```
# Hacemos las simulaciones para cada estrato
SE_cand_estrato_Fed <- Sim_cand_estrato_Federal(n[1])

for (j in 2:length(n)) {
  SE_cand_estrato_Fed  <- cbind(SE_cand_estrato_Fed, Sim_cand_estrato_Federal(n[j]))
}
# Se le da formato a la tabla de error estándar por estrato federal
colnames(SE_cand_estrato_Fed) <- as.character(n)
IClength_estrato_Fed <- 2*1.96*SE_cand_estrato_Fed

SE_cand_estrato_Fed <- formatC(SE_cand_estrato_Fed, digits = 4, format = "f")
SE_cand_estrato_Fed
```

```
##      50       100      200      300      400      500      600
## [1,] "0.0012" "0.0006" "0.0003" "0.0002" "0.0001" "0.0001" "0.0001"
## [2,] "0.0035" "0.0016" "0.0009" "0.0006" "0.0004" "0.0003" "0.0003"
## [3,] "0.0156" "0.0077" "0.0040" "0.0027" "0.0020" "0.0016" "0.0013"
## [4,] "0.0037" "0.0018" "0.0009" "0.0006" "0.0005" "0.0004" "0.0003"
## [5,] "0.0153" "0.0075" "0.0040" "0.0027" "0.0020" "0.0016" "0.0013"
## [6,] "0.0011" "0.0005" "0.0003" "0.0002" "0.0001" "0.0001" "0.0001"
##      700
## [1,] "0.0001"
## [2,] "0.0002"
## [3,] "0.0011"
## [4,] "0.0003"
## [5,] "0.0011"
## [6,] "0.0001"
```

```
# Se le da formato a la tabla de longitud del IC por estrato federal
IClength_estrato_Fed <- formatC(IClength_estrato_Fed, digits = 4, format = "f")
IClength_estrato_Fed
```

```
##      50       100      200      300      400      500      600
## [1,] "0.0045" "0.0022" "0.0012" "0.0008" "0.0006" "0.0005" "0.0004"
## [2,] "0.0137" "0.0064" "0.0034" "0.0022" "0.0017" "0.0013" "0.0011"
## [3,] "0.0613" "0.0301" "0.0157" "0.0107" "0.0080" "0.0065" "0.0052"
## [4,] "0.0145" "0.0071" "0.0036" "0.0023" "0.0018" "0.0015" "0.0012"
```

```
## [5,] "0.0599" "0.0296" "0.0156" "0.0104" "0.0079" "0.0063" "0.0050"
## [6,] "0.0043" "0.0021" "0.0011" "0.0008" "0.0006" "0.0004" "0.0003"
##      700
## [1,] "0.0003"
## [2,] "0.0009"
## [3,] "0.0044"
## [4,] "0.0010"
## [5,] "0.0043"
## [6,] "0.0003"
```

Ahora:

1. Reporta en una tabla el error estándar para cada candidato, tamaño de muestra y diseño (MAS/estratificaciones).

```
# Juntamos las distintas tablas en una tabla con toda la información
tablaTodos <- data.frame(rbind(cbind("Diseño"=rep("MAS",6), "Candidato"=cbind(y_i$partido), SE),
                               cbind("Diseño"=rep("Local",6), "Candidato"=cbind(y_i$partido), SE_cand_es
                               cbind("Diseño"=rep("Federal",6), "Candidato"=cbind(y_i$partido), SE_cand_
names(tablaTodos) <- c("Diseño", "Candidato", paste0("TamMuestra_", n))
kable(tablaTodos)
```

Diseño	Candidato	TamMuestra_50	TamMuestra_100	TamMuestra_200	TamMuestra_300	TamMuestra_400
MAS	mc	0.0019	0.0010	0.0005	0.0003	0.0002
MAS	otros	0.0037	0.0019	0.0010	0.0006	0.0005
MAS	pan_na	0.0095	0.0048	0.0025	0.0016	0.0012
MAS	prd	0.0043	0.0022	0.0011	0.0008	0.0005
MAS	pri_pvem	0.0093	0.0049	0.0024	0.0017	0.0012
MAS	pt	0.0022	0.0011	0.0006	0.0004	0.0003
Local	mc	0.0004	0.0002	0.0001	0.0001	0.0000
Local	otros	0.0026	0.0014	0.0007	0.0004	0.0004
Local	pan_na	0.0148	0.0078	0.0040	0.0028	0.0021
Local	prd	0.0029	0.0015	0.0008	0.0005	0.0004
Local	pri_pvem	0.0143	0.0076	0.0039	0.0028	0.0021
Local	pt	0.0012	0.0006	0.0003	0.0002	0.0002
Federal	mc	0.0012	0.0006	0.0003	0.0002	0.0001
Federal	otros	0.0035	0.0016	0.0009	0.0006	0.0004
Federal	pan_na	0.0156	0.0077	0.0040	0.0027	0.0020
Federal	prd	0.0037	0.0018	0.0009	0.0006	0.0005
Federal	pri_pvem	0.0153	0.0075	0.0040	0.0027	0.0020
Federal	pt	0.0011	0.0005	0.0003	0.0002	0.0001

2. Grafica los datos de la tabla: realiza una gráfica de paneles (con `facet_wrap()`), cada partido en un panel, en el eje horizontal grafica el tamaño de muestra y en el eje vertical el error estándar, tendrás en una misma gráfica tres curvas, una para muestreo aleatorio simple y una para cada estratificación.

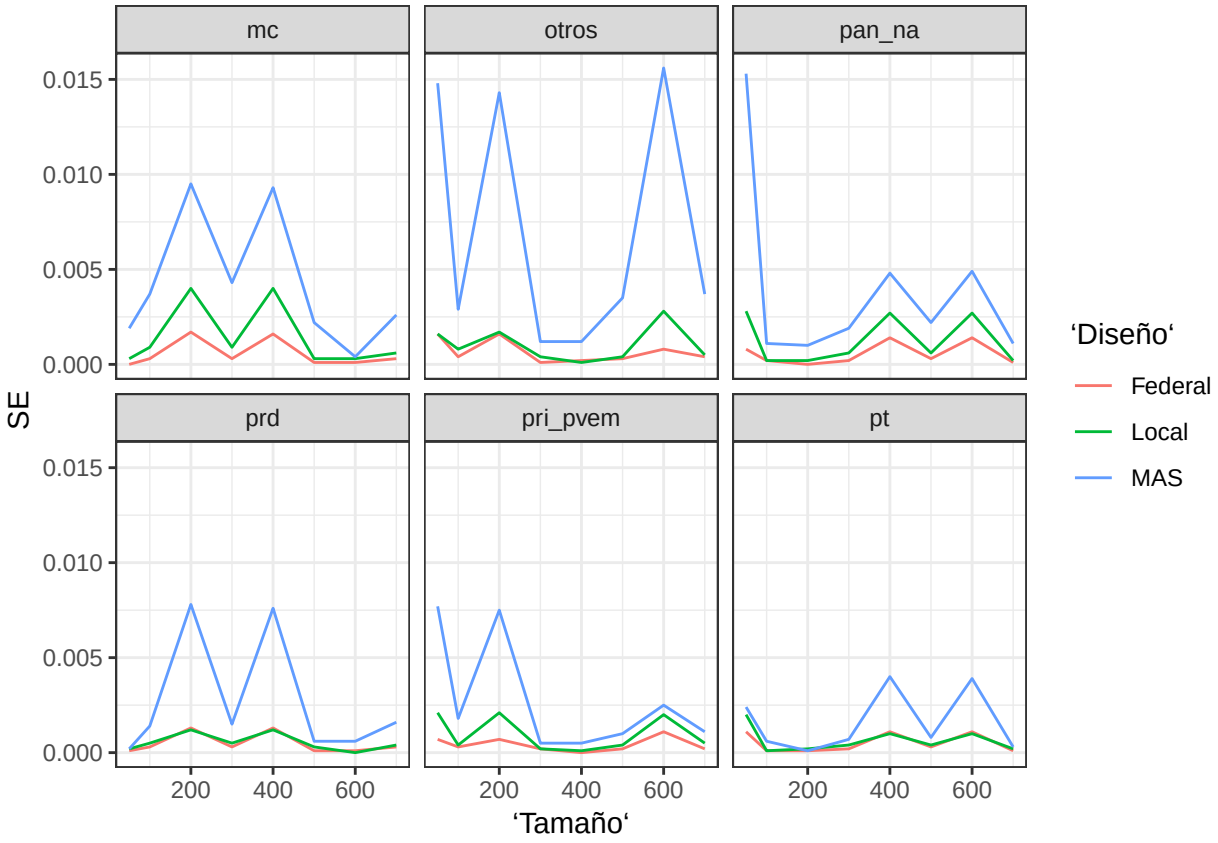
```
# Graficamos el error estándar por partido, muestreo y tamaño de muestra

tablaTodos[,3:ncol(tablaTodos)]<- matrix(unlist(lapply(3:ncol(tablaTodos), function(i){as.numeric(tabla
                                         nrow = nrow(tablaTodos), byrow = TRUE)

tablaTodosgraf <- tablaTodos %>% gather("Tamaño", "SE" ,TamMuestra_50:TamMuestra_700) %>%
  mutate("Tamaño"=as.numeric(substr(Tamaño, 12,14)))

ggplot(tablaTodosgraf)+theme_bw()+
```

```
  geom_line(aes(Tamaño, SE, col=Diseño))+
  facet_wrap(~Candidato)
```



3. ¿Qué diseño y tamaño de muestra elegirías? Explica tu respuesta y de ser necesario repite los pasos i-iii para otros valores de n.

HAY QUE HACERLO DESPUÉS DE CORRERLO!!!!!