2. Genere una muestra aleatoria $X_1, \ldots, X_n$, de tamaño $n = 30$, de una población con distribución $\mathcal{N}(5, 4)$. Genere otra muestra aleatoria $Y_1, \ldots, Y_m$, de tamaño $m = 50$, de una población con distribución $\mathcal{N}(2, 4)$. Obtenga los intervalos de confianza para $\mu_x - \mu_y$ tales que

a) Intervalo del 80% de confianza, suponiendo σ^2 conocida.

b) Intervalo del 80% de confianza, suponiendo σ^2 desconocida común.

c) Intervalo del 95% de confianza, suponiendo σ^2 conocida.

d) Intervalo del 95% de confianza, suponiendo σ^2 desconocida común.

Repita el proceso generando cada una de las muestras 100 veces. ¿Cómo son los intervalos? Identifique los intervalos con mayor longitud y con menos longitud. Compare y explique los resultados.

Solución:

a) Usaremos que un intervalo al $100(1-\alpha)\%$ de confianza para $\mu_x - \mu_y$ cuando σ^2 es conocida es

$$\left((\bar{X} - \bar{Y}) - z_{1-\alpha/2}\sqrt{\frac{\sigma^2}{n} + \frac{\sigma^2}{m}}, (\bar{X} - \bar{Y}) + z_{1-\alpha/2}\sqrt{\frac{\sigma^2}{n} + \frac{\sigma^2}{m}} \right) \quad (1) \qquad (1)$$

De tal manera comenzamos generando las muestras aleatorias con las indicaciones dadas

```
set.seed(123)

# tamaño de las muestras
n <- 30
m <- 50

# generación de las muestras
muestra_x <- rnorm(n, mean = 5, sd=2)
muestra_y <- rnorm(m, mean = 2, sd=2)
```

y después procedemos a hacer los cálculos pertinentes

```
x_bar <- sum(muestra_x) / n
y_bar <- sum(muestra_y) / m

# notemos que alpha = 0.2
uno_alphamedios <- 1 - 0.2 / 2
z <- qnorm(uno_alphamedios, mean = 0, sd = 1)

raiz <- ( (4 / n) + (4 / m) ) ** 0.5
```

Posteriormente definimos y calculamos los límites inferior y superior del intervalo de confianza dado en (1):

```
limite_inferior <- x_bar - y_bar - z * raiz
limite_superior <- x_bar - y_bar + z * raiz
```

Finalmente vemos el intervalo del 80% de confianza para $\mu_x - \mu_y$:

```
print(paste("(", limite_inferior, ",", limite_superior, ")"))
```

```
## [1] "( 2.19579682261423 , 3.37964344877033 )"
```

Sabemos en realidad que $\mu_x - \mu_y = 3$, por lo que el intervalo obtenido anteriormente es bueno pues dentro de él está el valor verdadero de $\mu_x - \mu_y$. Finalmente vemos la longitud de dicho intervalo

```
limite_superior-limite_inferior
```

```
## [1] 1.183847
```

b) Usaremos que un intervalo al $100(1-\alpha)\%$ para $\mu_x - \mu_y$, cuando σ^2 es desconocido es

$$\left((\bar{X} - \bar{Y}) - t^{1-\alpha/2}_{n+m-2} \sqrt{\left(\frac{1}{n} + \frac{1}{m}\right) S^2_p}, (\bar{X} - \bar{Y}) + t^{1-\alpha/2}_{n+m-2} \sqrt{\left(\frac{1}{n} + \frac{1}{m}\right) S^2_p} \right) \tag{2}$$

Donde

$$S^2_p = \frac{(n-1)S^2_x + (m-1)S^2_y}{n+m-2}, \quad S^2_x = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - \bar{X})^2 \quad \text{y} \quad S^2_x = \frac{1}{m-1}\sum_{i=1}^{m}(Y_i - \bar{Y})^2$$

De forma análoga al inciso anterior haremos los respectivos cálculos:

```
# notemos que alpha = 0.2
uno_alphamedios2 <- 1 - 0.2 / 2
t <- qt(uno_alphamedios2, df = n + m - 2)

# S^2:

Scuad_x <- sum( (muestra_x - x_bar) ** 2 ) / (n-1)
Scuad_y <- sum( (muestra_y - y_bar) ** 2 ) / (m-1)

# S^2_p:

Scuad_p <- ( (n-1) * Scuad_x + (m-1) * Scuad_y ) / (n + m -2)

raiz2 <- ( ((1 / n) + (1 / m)) * Scuad_p ) ** 0.5
```

para posteriormente crear y ver el intervalo de confianza

```
# limites

limite_inferior2 <- x_bar - y_bar - t * raiz2
limite_superior2 <- x_bar - y_bar + t * raiz2

# intervalo

print(paste("(", limite_inferior2, ",", limite_superior2, ")"))
```

```
## [1] "( 2.23362853551926 , 3.3418117358653 )"
```

el cual igualmente es bueno y cuya longitud es

```
limite_superior2 - limite_inferior2
```

```
## [1] 1.108183
```

mayor respecto al intervalo dado en el inciso *a*).

Incisos *c*) y *d*): resolveremos estos ejercicios a la par. De hecho, bastará con reutilizar el código de los incisos *a*) y *b*) modificando únicamente el valor de α, donde pasaremos de considerar $\alpha = 0.2$ a $\alpha = 0.05$. Para ello:

```
# inciso c)

# Tamaños:
n <- 30
```

```r
m <- 50

# Muestras:
muestra_x <- rnorm(n, mean = 5, sd=2)
muestra_y <- rnorm(m, mean = 2, sd=2)

# Cálculos:

x_bar <- sum(muestra_x) / n
y_bar <- sum(muestra_y) / m

# En este caso alpha = 0.05
uno_alphamedios <- 1 - 0.05 / 2
z <- qnorm(uno_alphamedios, mean = 0, sd = 1)

raiz <- ( (4 / n) + (4 / m) ) ** 0.5

# Límites:

limite_inferior <- x_bar - y_bar - z * raiz
limite_superior <- x_bar - y_bar + z * raiz

# Resultado:
print(paste("(", limite_inferior, ",", limite_superior, ")"))
```

```
## [1] "( 2.62947263754802 , 4.4400098120699 )"
```

cuya longitud es

```r
limite_superior - limite_inferior
```

```
## [1] 1.810537
```

siendo este intervalo más grande respecto al intervalo del inciso *a*). Recordemos que al tener un intervalo del 95% de confianza, tendremos que un gran número de intervalos contendrán el valor de $\mu_x - \mu_y$ el 95% de las veces, versus el 80% de las veces del inciso *a*).

Continuando

```r
# inciso d)

# Tamaños:
n <- 30
m <- 50

# Muestras:
muestra_x <- rnorm(n, mean = 5, sd=2)
muestra_y <- rnorm(m, mean = 2, sd=2)

# Cálculos

x_bar <- sum(muestra_x) / n
y_bar <- sum(muestra_y) / m

# notemos que alpha = 0.05
uno_alphamedios2 <- 1 - 0.05 / 2
```

```
t <- qt(uno_alphamedios2, df = n + m - 2)

# S^2:

Scuad_x <- sum( (muestra_x - x_bar) ** 2 ) / (n-1)
Scuad_y <- sum( (muestra_y - y_bar) ** 2 ) / (m-1)

# S^2_p:

Scuad_p <- ( (n-1) * Scuad_x + (m-1) * Scuad_y ) / (n + m -2)

raiz <- ( ((1 / n) + (1 / m)) * Scuad_p ) ** 0.5

# limites

limite_inferior2 <- x_bar - y_bar - t * raiz
limite_superior2 <- x_bar - y_bar + t * raiz

# intervalo

print(paste("(", limite_inferior2, ",", limite_superior2, ")"))
```

```
## [1] "( 2.43226557288083 , 4.19277921705026 )"
```

cuya longitud es

```
limite_superior2 - limite_inferior2
```

```
## [1] 1.760514
```

mayor respecto a la longitud del intervalo del inciso *b*).

Ahora bien, repetiremos el proceso generando cada una de las muestras 100 veces, esto es, repetiremos los 4 incisos pero con la simulación de 100 muestras.

a.2) Comenzaremos con la simulación de las muestras 100 veces para un intervalo de 80% de confianza para $\mu_x - \mu_y = 3$. Ahora, cargamos los paquetes que utilizaremos y definimos dos vectores vacíos que guardaran los límites inferiores y superiores, respectivamente, de los intervalos de confianza que generaremos

```
# Cargamos las librerías
library(ggplot2)
library(dplyr)

# Tamaño de las muestras
n <- 30
m <- 50

# Nos auxiliamos de dos vectores vacíos
limite_inf <- c()
limite_sup <- c()
```

Luego, mediante un bucle `for` simularemos las muestras 100 veces, donde dentro del bucle colocaremos el código del inciso *a*)

```
for( i in 1:100 ){
```

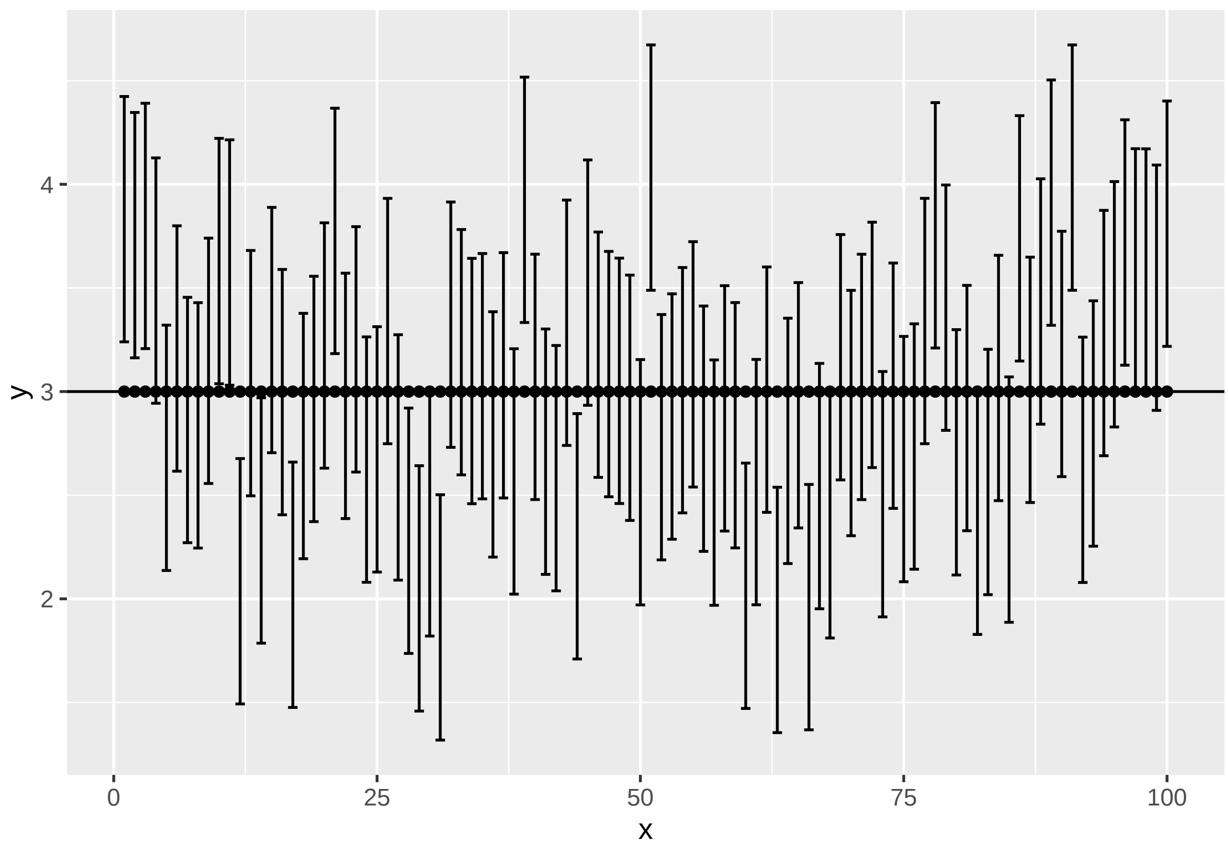
```
  # Muestras
  muestra_x <- rnorm(n, mean = 5, sd=2)
  muestra_y <- rnorm(m, mean = 2, sd=2)

  # Cálculos
  x_bar <- sum(muestra_x) / n
  y_bar <- sum(muestra_y) / m

  # alpha = 0.2
  uno_alphamedios <- 1 - (0.2 / 2)
  z <- qnorm(uno_alphamedios, mean = 0, sd = 1)
  raiz <- ( (4 / n) + (4 / m) ) ** 0.5

  # límites
  l_inferior <- x_bar - y_bar - z * raiz
  l_superior <- x_bar - y_bar + z * raiz

  # guardamos dichos límites en los vectores "vacíos"
  limite_inf[i] <- l_inferior
  limite_sup[i] <- l_superior
}
```

procedemos después a crear un `dataframe` que almacene los límites inferiores y superiores cálculados anteriormente, asimismo calculamos las longitudes de los intervalos:

```
# Creamos el dataframe con 100 filas
x <- 100
df <- data.frame(x = x,
                 y = 3,
                 limite_inferior = limite_inf,
                 limite_superior = limite_sup)

# Calculamos la longitud de cada intervalo y agregamos
# dichos valores al dataframe

df <- df %>% mutate(long_int = limite_superior - limite_inferior)
df <- df %>% arrange(long_int)
df['x'] <- 1:100
```

Finalmente graficamos:

```
ggplot(df, aes(x,y)) + geom_point() +
  geom_errorbar(aes(ymin = limite_inferior, ymax = limite_superior)) +
  geom_hline(yintercept = 3)
```



Si bien la gran mayoría de los intervalos contiene al valor real de $\mu_x - \mu_y$, se aprecia que varios no intersectan la recta $y = 3$. Lo cual es claro puesto que, teóricamente, el 80% de los intervalos contendrán el valor real de $\mu_x - \mu_y$.

Ahora vemos que la longitud de los intervalos, en este caso, coinciden

```
# longitud máxima
max(df$long_int)
```

```
## [1] 1.183847
```

```
# longitud mínima
min(df$long_int)
```

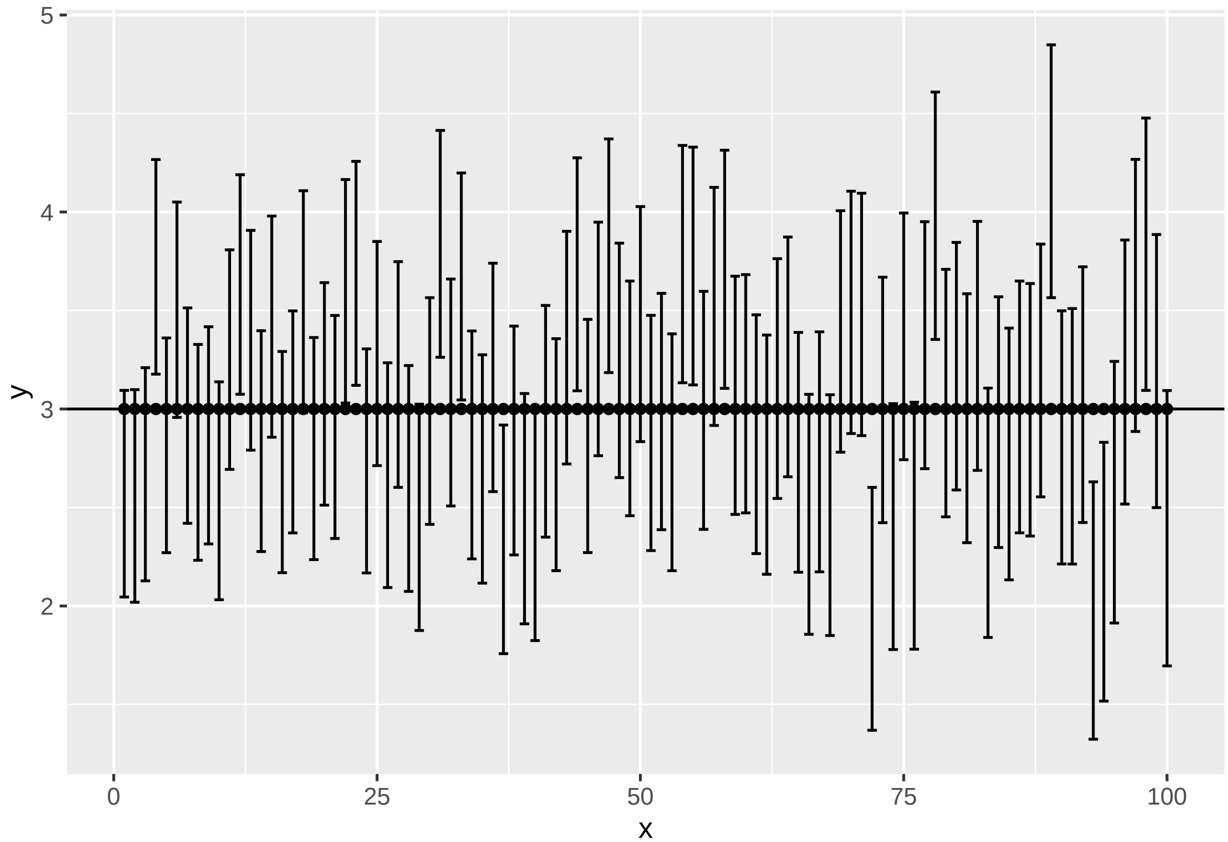
```
## [1] 1.183847
```

lo cual se le atribuye a que la varianza en este caso es conocida.

Incisos $b.2), c.2)$ y $d.2)$: Estos ejercicios los resolveremos "al mismo tiempo" ya que sólo debemos copiar el código del inciso anterior y adaptarlo según sea el caso. Comenzamos por los intervalos de confianza para $\mu_x - \mu_y$ cuando σ^2 es desconocida; el código es

```
# Volvemos a escribir todo el código:

n <- 30
m <- 50

# Nos auxiliamos de dos vectores vacíos
limite_inf <- c()
limite_sup <- c()
```

```r
for( i in 1:100 ){

  # Muestras
  muestra_x <- rnorm(n, mean = 5, sd=2)
  muestra_y <- rnorm(m, mean = 2, sd=2)

  # Cálculos
  x_bar <- sum(muestra_x) / n
  y_bar <- sum(muestra_y) / m

  # notemos que alpha = 0.2
  uno_alphamedios2 <- 1 - 0.2 / 2
  t <- qt(uno_alphamedios2, df = n + m - 2)

  # S^2:

  Scuad_x <- sum( (muestra_x - x_bar) ** 2 ) / (n-1)
  Scuad_y <- sum( (muestra_y - y_bar) ** 2 ) / (m-1)

  # S^2_p:

  Scuad_p <- ( (n-1) * Scuad_x + (m-1) * Scuad_y ) / (n + m -2)

  raiz <- ( ((1 / n) + (1 / m)) * Scuad_p ) ** 0.5

  # limites

  l_inferior <- x_bar - y_bar - t * raiz
  l_superior <- x_bar - y_bar + t * raiz

  # guardamos dichos límites en los vectores "vacíos"
  limite_inf[i] <- l_inferior
  limite_sup[i] <- l_superior
}

# Creamos el dataframe con 100 filas
x <- 100
df <- data.frame(x = x,
                 y = 3,
                 limite_inferior = limite_inf,
                 limite_superior = limite_sup)

# Calculamos la longitud de cada intervalo y agregamos
# dichos valores al dataframe

df <- df %>% mutate(long_int = limite_superior - limite_inferior)
df <- df %>% arrange(long_int)
df['x'] <- 1:100

ggplot(df, aes(x,y)) + geom_point() +
  geom_errorbar(aes(ymin = limite_inferior, ymax = limite_superior)) +
  geom_hline(yintercept = 3)
```



Luego, la longitud máxima y mínima de los intervalos son

```
max(df$long_int)
```

```
## [1] 1.397895
```

```
min(df$long_int)
```

```
## [1] 1.048868
```

Cabe resaltar que en este caso las longitudes máximas y mínimas no coinciden, de donde en general las longitudes de todos los intervalos no coincidirán, puesto que en este caso la varianza es desconocida.

Procedemos a identificar los intervalos con mayor y menor longitud

```
# intervalo de longitud mayor
filter(df, long_int == max(long_int))
```

```
##     x y limite_inferior limite_superior long_int
## 1 100 3        1.695842        3.093738 1.397895
```

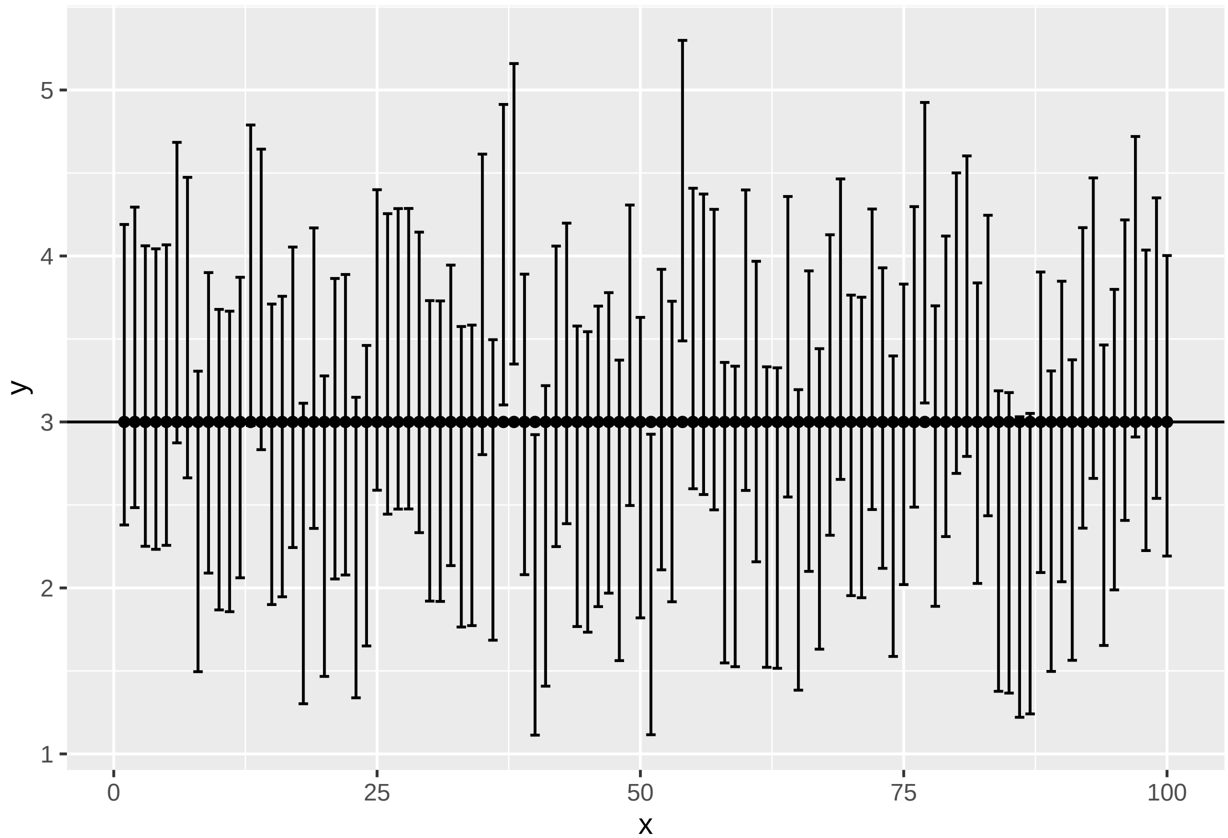
```
# intervalo de longitud menor
filter(df, long_int == min(long_int))
```

```
##   x y limite_inferior limite_superior long_int
## 1 1 3        2.045583        3.094451 1.048868
```

Pasamos después a los intervalos del 95% de confianza. Primero para el caso en que σ^2 es conocida.

```
# Tamaño de las muestras
n <- 30
```

```r
m <- 50

# Nos auxiliamos de dos vectores vacíos
limite_inf <- c()
limite_sup <- c()

for( i in 1:100 ){

  # Muestras
  muestra_x <- rnorm(n, mean = 5, sd=2)
  muestra_y <- rnorm(m, mean = 2, sd=2)

  # Cálculos
  x_bar <- sum(muestra_x) / n
  y_bar <- sum(muestra_y) / m

  # ahora alpha = 0.05
  uno_alphamedios <- 1 - (0.05 / 2)
  z <- qnorm(uno_alphamedios, mean = 0, sd = 1)
  raiz <- ( (4 / n) + (4 / m) ) ** 0.5

  # límites
  l_inferior <- x_bar - y_bar - z * raiz
  l_superior <- x_bar - y_bar + z * raiz

  # guardamos dichos límites en los vectores "vacíos"
  limite_inf[i] <- l_inferior
  limite_sup[i] <- l_superior
}

# Creamos el dataframe con 100 filas
x <- 100
df <- data.frame(x = x,
                 y = 3,
                 limite_inferior = limite_inf,
                 limite_superior = limite_sup)

# Calculamos la longitud de cada intervalo y agregamos
# dichos valores al dataframe

df <- df %>% mutate(long_int = limite_superior - limite_inferior)
df <- df %>% arrange(long_int)
df['x'] <- 1:100

ggplot(df, aes(x,y)) + geom_point() +
  geom_errorbar(aes(ymin = limite_inferior, ymax = limite_superior)) +
  geom_hline(yintercept = 3)
```



del cual claramente se aprecia mejora en cuanto al número de intersecciones de los intervalos con la recta $y = 3$ versus la gráfica del inciso $a.2$).

De nuevo la longitud de los intervalos será la misma:

```
max(df$long_int)
```

```
## [1] 1.810537
```

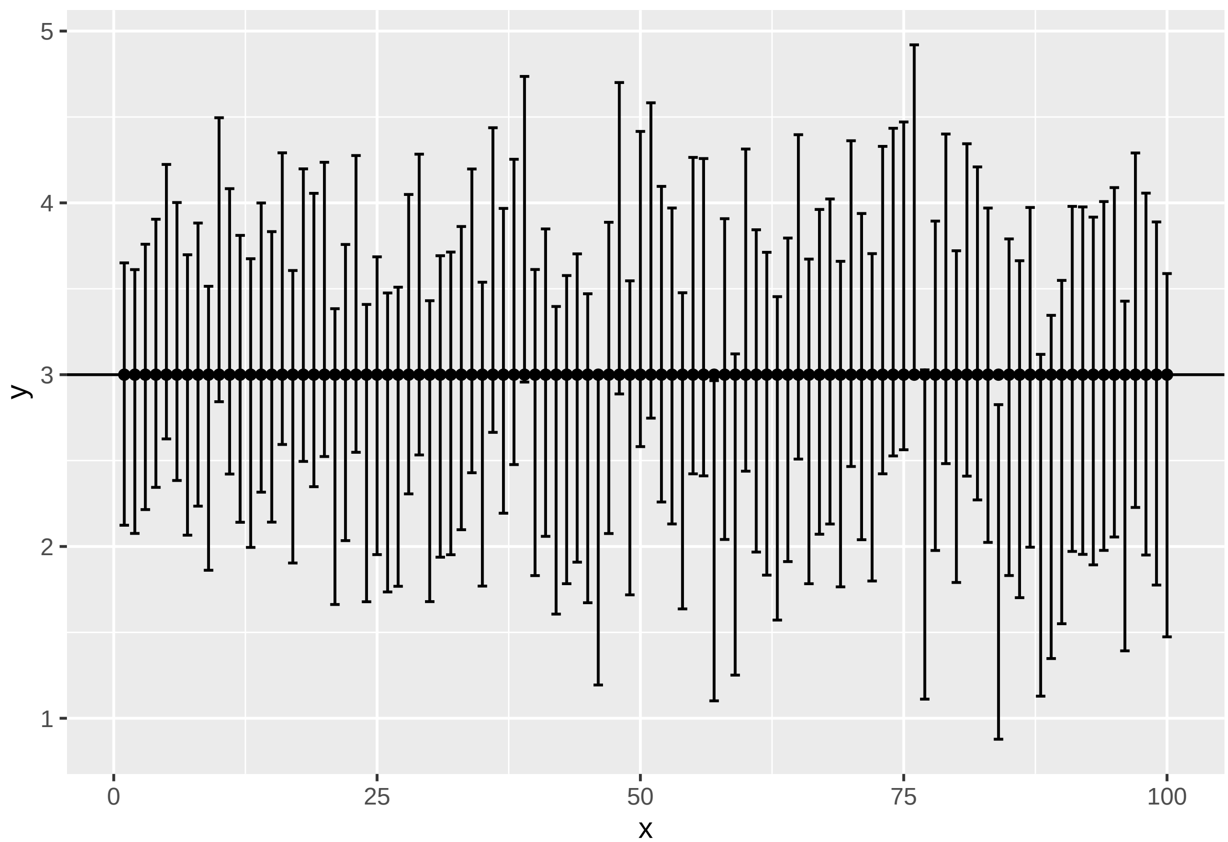
```
min(df$long_int)
```

```
## [1] 1.810537
```

Finalmente el caso en el que σ^2 es desconocida:

```
# Volvemos a escribir todo el código:

n <- 30
m <- 50

# Nos auxiliamos de dos vectores vacíos
limite_inf <- c()
limite_sup <- c()

for( i in 1:100 ){

  # Muestras
  muestra_x <- rnorm(n, mean = 5, sd=2)
  muestra_y <- rnorm(m, mean = 2, sd=2)
```

```r
  # Cálculos
  x_bar <- sum(muestra_x) / n
  y_bar <- sum(muestra_y) / m

  # notemos que alpha = 0.05
  uno_alphamedios2 <- 1 - 0.05 / 2
  t <- qt(uno_alphamedios2, df = n + m - 2)

  # S^2:

  Scuad_x <- sum( (muestra_x - x_bar) ** 2 ) / (n-1)
  Scuad_y <- sum( (muestra_y - y_bar) ** 2 ) / (m-1)

  # S^2_p:

  Scuad_p <- ( (n-1) * Scuad_x + (m-1) * Scuad_y ) / (n + m -2)

  raiz <- ( ((1 / n) + (1 / m)) * Scuad_p ) ** 0.5

  # limites

  l_inferior <- x_bar - y_bar - t * raiz
  l_superior <- x_bar - y_bar + t * raiz

  # guardamos dichos límites en los vectores "vacíos"
  limite_inf[i] <- l_inferior
  limite_sup[i] <- l_superior
}

# Creamos el dataframe con 100 filas
x <- 100
df <- data.frame(x = x,
                 y = 3,
                 limite_inferior = limite_inf,
                 limite_superior = limite_sup)

# Calculamos la longitud de cada intervalo y agregamos
# dichos valores al dataframe

df <- df %>% mutate(long_int = limite_superior - limite_inferior)
df <- df %>% arrange(long_int)
df['x'] <- 1:100

ggplot(df, aes(x,y)) + geom_point() +
  geom_errorbar(aes(ymin = limite_inferior, ymax = limite_superior)) +
  geom_hline(yintercept = 3)
```



de igual manera se ve la mejoría respecto a la gráfica del inciso $b.2$) puesto que 95, de los 100, intervalos intersectan a $y = 3$ versus los 80 intervalos que intersectan a dicha recta en la gráfica del inciso $b.2$).

Después

```r
max(df$long_int)
```

```
## [1] 2.114582
```

```r
min(df$long_int)
```

```
## [1] 1.527108
```

donde tendremos intervalos de diferentes longitudes:

```r
# intervalo de longitud mayor
filter(df, long_int == max(long_int))
```

```
##     x y limite_inferior limite_superior long_int
## 1 100 3        1.473893        3.588475 2.114582
```

```r
# intervalo de longitud menor
filter(df, long_int == min(long_int))
```

```
##   x y limite_inferior limite_superior long_int
## 1 1 3        2.123596        3.650704 1.527108
```

Con todo lo visto en este ejercicio se ilustra de manera práctica los conceptos de intervalos de confianza y se ve de manera contundente lo que significa el *nivel de confianza* en este tema.