Laboratorio de Muestreo y Estimadores

Carmen Lancho - Isaac Martín - Víctor Aceña

Grado en Ciencia e Ingeniería de Datos - Inferencia Estadística - Curso 2024/2025

Índice

Objetivo

El objetivo principal de este laboratorio es aprender a aplicar técnicas de **muestreo estadístico** y a calcular **estimaciones puntuales** de parámetros poblacionales a partir de una muestra. Como objetivos secundarios tenemos:

1. Comprender los distintos tipos de **muestreo probabilístico** y su aplicación en diferentes situaciones.
2. Saber calcular **estimaciones puntuales** y **estadísticos muestrales** en R.
3. Identificar y entender las **propiedades** de los estimadores, como **insesgadez**, **eficiencia**, **consistencia** y **suficiencia**, mediante ejemplos y simulaciones.
4. Saber interpretar los resultados obtenidos y su implicación en la inferencia sobre la población.

1. Introducción

En esta sección, introduciremos los conceptos básicos de **muestreo estadístico** y **estimación**. El muestreo es el proceso mediante el cual seleccionamos una parte de la población para obtener información sobre el todo. Los **estimadores** son las herramientas que utilizamos para aproximar los parámetros poblacionales (como la media o la proporción) a partir de la muestra obtenida.

Veremos los siguientes conceptos:

1. Tipos de muestreo: Muestreo aleatorio simple, estratificado, conglomerados y sistemático.
2. Definición de estimadores y su diferencia con los parámetros poblacionales.
3. Propiedades deseables de los estimadores: insesgadez, eficiencia, consistencia.

2. Tipos de Muestreo

Existen diferentes métodos de muestreo que podemos utilizar dependiendo de la situación y la naturaleza de la población. A continuación, se describen brevemente los métodos más comunes:

2.1. Muestreo Aleatorio Simple (MAS)

El **muestreo aleatorio simple** es el método más sencillo. Cada elemento de la población tiene la misma probabilidad de ser seleccionado. Es ideal para poblaciones pequeñas y homogéneas.

> **Ejemplo**: Si tenemos una lista de 100 estudiantes y queremos seleccionar una muestra aleatoria de 20, cada estudiante tiene una probabilidad igual de ser seleccionado.

```r
set.seed(123)
poblacion <- 1:100  # Población de 100 estudiantes
muestra <- sample(poblacion, size = 20, replace = FALSE)  # Seleccionar una muestra aleatoria de 20
```

En este ejemplo, realizamos un **muestreo aleatorio simple** en una población de 100 estudiantes para seleccionar una muestra de 20 estudiantes.

1. **Establecer la semilla para la reproducibilidad**:

   La función `set.seed(123)` fija la semilla de los números aleatorios en R. Esto asegura que cada vez que se ejecute el código, se obtenga la misma muestra aleatoria, permitiendo la **reproducibilidad** de los resultados.

2. **Definir la población**:

   Creamos un vector llamado `poblacion` que contiene los números del 1 al 100. Cada número representa a un estudiante único en la población total de 100 estudiantes.

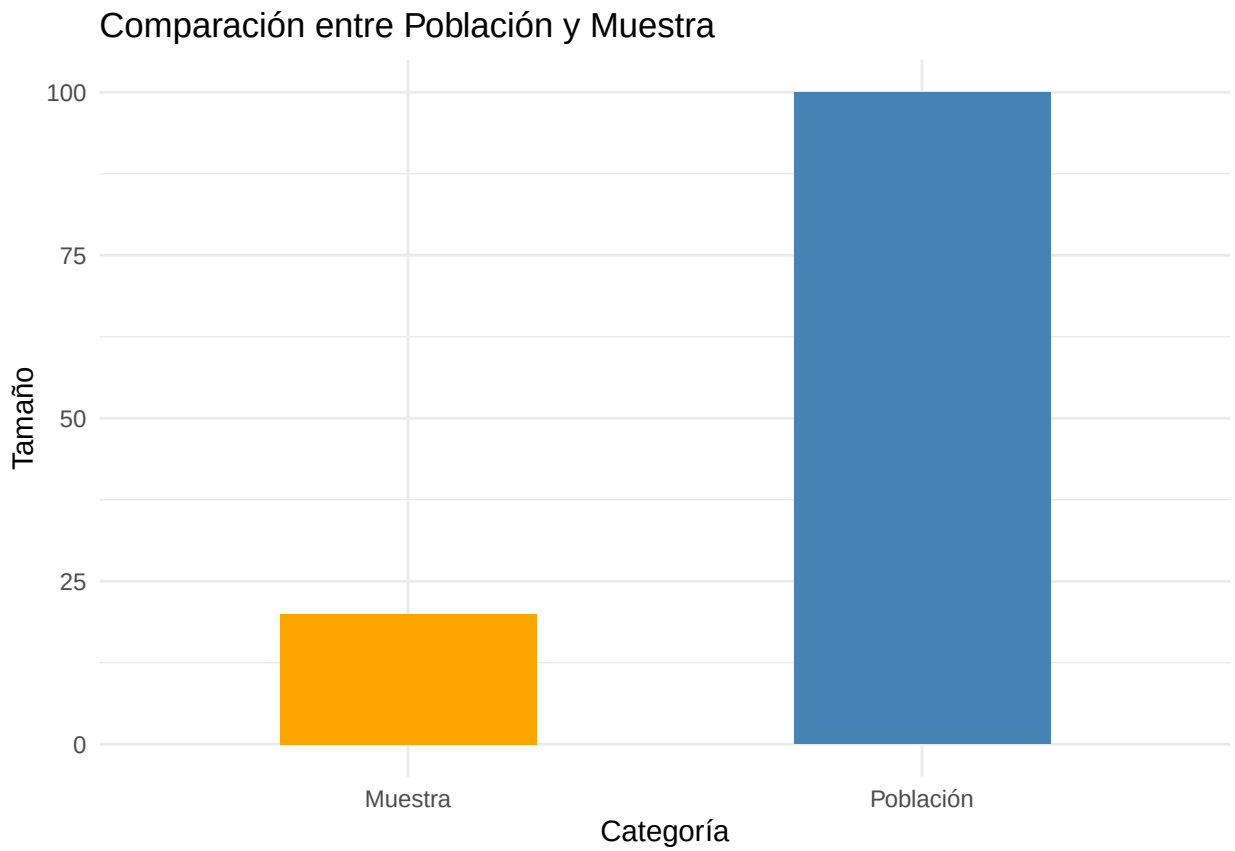
3. **Seleccionar la muestra aleatoria**:

   Utilizamos la función `sample()` para seleccionar una muestra aleatoria de 20 estudiantes de la población definida. El parámetro `size = 20` indica el número de estudiantes a seleccionar, y `replace = FALSE` asegura que la selección sea **sin reemplazo**, es decir, que cada estudiante solo pueda ser seleccionado una vez.

   > Si `replace = TRUE`, los elementos seleccionados podrían repetirse en la muestra, lo que sería un **muestreo con reemplazamiento**.

Este método garantiza que cada estudiante en la población tenga la misma probabilidad de ser seleccionado para la muestra, asegurando una **representación equitativa** y **aleatoria** de la población en la muestra.

```
# Crear un dataframe con la información de la población y la muestra
data_proporcion <- data.frame(
  Categoria = c("Población", "Muestra"),
  Tamano = c(length(poblacion), length(muestra))
)

# Crear el gráfico de barras
ggplot(data_proporcion, aes(x = Categoria, y = Tamano, fill = Categoria)) +
  geom_bar(stat = "identity", width = 0.5) +
  scale_fill_manual(values = c("Población" = "steelblue", "Muestra" = "orange")) +
  labs(title = "Comparación entre Población y Muestra",
       x = "Categoría",
       y = "Tamaño") +
  theme_minimal() +
  theme(legend.position = "none")
```



Este gráfico de barras destaca visualmente la diferencia entre el número total de estudiantes en la población y los que fueron seleccionados para formar parte de la muestra.

¿Cuándo usar el Muestreo Aleatorio Simple?

Este tipo de muestreo es ideal cuando:

- La población es **pequeña** y homogénea.

- Todos los elementos de la población tienen la **misma probabilidad** de ser seleccionados.
- No existen diferencias importantes entre subgrupos dentro de la población.

Sin embargo, puede no ser el método adecuado para poblaciones grandes o cuando existen diferencias significativas entre los subgrupos, en cuyo caso se podrían utilizar otros métodos como el **muestreo estratificado**.

2.2. Muestreo Estratificado

El **muestreo estratificado** se utiliza cuando la población no es homogénea, pero sabemos que podemos dividirla en grupos (estratos) homogéneos. De cada estrato, se selecciona una muestra aleatoria. Este método es útil cuando los subgrupos dentro de la población tienen características que afectan el estudio y queremos asegurarnos de que estén representados adecuadamente.

> **Ejemplo**: Si queremos muestrear a estudiantes de diferentes carreras (estratos), podemos dividir la población de estudiantes en función de la carrera y seleccionar una muestra aleatoria de cada carrera para asegurar que todas las carreras estén representadas en nuestra muestra.

```r
# Simulación de una población de estudiantes clasificados por carrera
set.seed(123)
poblacion <- data.frame(
  id = 1:100,  # Identificación de estudiantes
  carrera = sample(c("Ingeniería", "Medicina", "Derecho", "Ciencias Sociales"), 100, replace = TRUE)
)

# Tamaño de la muestra total
n_muestra <- 20

# Muestreo estratificado: número de estudiantes seleccionados por carrera
muestra_estratificada <- poblacion |>
  group_by(carrera) |>
  sample_n(size = round(n_muestra * n() / nrow(poblacion)), replace = FALSE)
```

En este ejemplo, realizamos un **muestreo estratificado** en una población de 100 estudiantes distribuidos en diferentes carreras.

1. **Crear la población**:

   Creamos un dataframe llamado `poblacion` que contiene 100 estudiantes. Cada estudiante tiene un `id` único del 1 al 100 y está asignado aleatoriamente a una de las 4 carreras (**Ingeniería**, **Medicina**, **Derecho**, **Ciencias Sociales**) utilizando la función `sample()`. Esto simula una población diversificada donde cada carrera puede tener una representación diferente.

2. **Determinar el tamaño de la muestra**:

   Definimos que queremos una muestra total de 20 estudiantes. Este número determina cuántos estudiantes serán seleccionados en total a partir de la población.

3. **Calcular el tamaño de la muestra por estrato**:

   Calculamos cuántos estudiantes se deben seleccionar de cada carrera (estrato) proporcionalmente al tamaño de cada carrera en la población total. Utilizamos la fórmula `round(n_muestra * n() / nrow(poblacion))` para asegurarnos de que la cantidad de estudiantes seleccionados de cada carrera refleje su proporción en la población. Esto garantiza que cada estrato esté representado adecuadamente en la muestra final.

4. **Seleccionar la muestra estratificada**:

   Utilizamos la función `group_by()` para agrupar los estudiantes por su carrera. Luego, aplicamos `sample_n()` para seleccionar una muestra aleatoria dentro de cada grupo estratificado, según el tamaño calculado previamente. Este proceso asegura que la muestra obtenida sea representativa de la diversidad de carreras presentes en la población.

5. **Obtener la muestra final**:

   La muestra estratificada resultante contiene una representación proporcional de cada carrera, de acuerdo con su presencia en la población total. Esto mejora la precisión y representatividad de las estimaciones realizadas a partir de la muestra, ya que cada subgrupo relevante está adecuadamente representado.

Este método asegura que cada carrera esté representada adecuadamente en la muestra final, mejorando la **representatividad** y **precisión** de las estimaciones obtenidas a partir de la muestra.

Detalle de las Funciones Utilizadas

1. `sample_n()`:

   La función `sample_n()` se utiliza para seleccionar una muestra aleatoria dentro de cada estrato. En este caso, garantiza que se seleccione un número de estudiantes proporcional al tamaño de cada grupo en la población total. Esto asegura que las muestras de cada carrera sean representativas y reflejen la distribución real en la población.

2. `group_by()`:

   `group_by()` agrupa a los estudiantes por la variable `carrera`. De esta manera, se pueden aplicar operaciones (como `sample_n()`) a cada grupo por separado. En este ejemplo, primero agrupamos los estudiantes según la carrera y luego aplicamos el muestreo dentro de cada grupo, lo que facilita una selección proporcional y representativa.

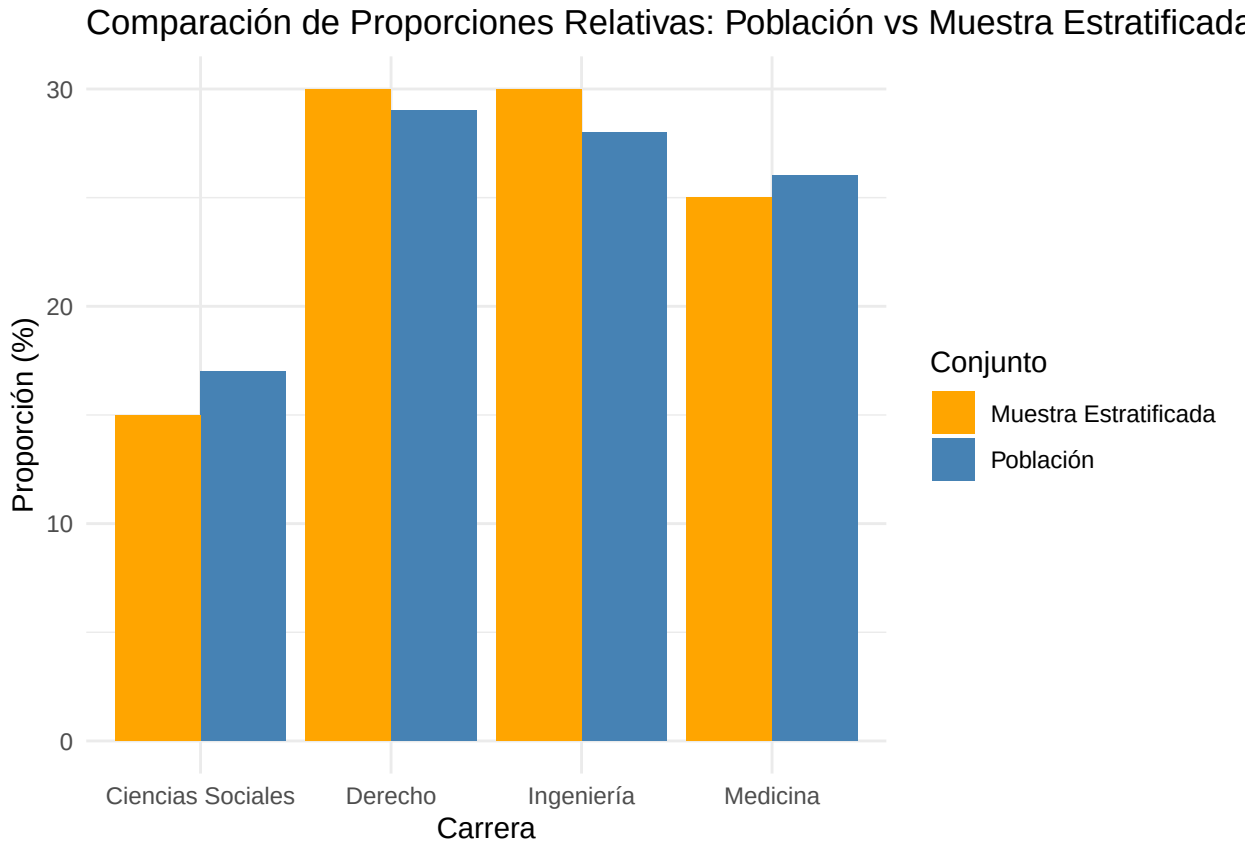
3. `round(n_muestra * n() / nrow(poblacion))`:

   Esta fórmula calcula el número de estudiantes que se deben seleccionar de cada grupo (estrato) en función del tamaño total de la muestra. El uso de `round()` asegura que el número de estudiantes seleccionados sea un número entero, manteniendo la proporcionalidad y evitando fracciones de estudiantes.

Este enfoque de muestreo estratificado mejora la **exactitud** y **eficiencia** de las estimaciones estadísticas al asegurar que todos los subgrupos importantes de la población estén adecuadamente representados en la muestra.

```r
# Calcular la proporción de cada carrera en la población
proporcion_poblacion <- poblacion |>
  count(carrera) |>
  mutate(prop_poblacion = n / sum(n)) |>
  select(carrera, prop_poblacion)

# Calcular la proporción de cada carrera en la muestra, pero con respecto al total de la muestra (n_mue
proporcion_muestra <- muestra_estratificada |>
  count(carrera) |>
  mutate(prop_muestra = n / sum(n_muestra)) |>
  select(carrera, prop_muestra)

# Unir las proporciones de población y muestra
proporciones_comparadas <- left_join(proporcion_poblacion, proporcion_muestra, by = "carrera")
```

```r
# Convertir a formato largo para ggplot
proporciones_long <- proporciones_comparadas |>
  pivot_longer(cols = starts_with("prop"), names_to = "Conjunto", values_to = "Proporción")

# Renombrar las categorías para hacerlas más claras en el gráfico
proporciones_long$Conjunto <- recode(proporciones_long$Conjunto,
                                     "prop_poblacion" = "Población",
                                     "prop_muestra" = "Muestra Estratificada")

# Crear el gráfico de barras comparando las proporciones
ggplot(proporciones_long, aes(x = carrera, y = Proporción * 100, fill = Conjunto)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(title = "Comparación de Proporciones Relativas: Población vs Muestra Estratificada",
       x = "Carrera",
       y = "Proporción (%)") +
  scale_fill_manual(values = c("Población" = "steelblue", "Muestra Estratificada" = "orange")) +
  theme_minimal()
```



Este gráfico de barras compara las proporciones relativas de cada carrera en la **población total** con las proporciones en la **muestra estratificada**. En el eje **X** se representan las diferentes carreras (**Ingeniería**, **Medicina**, **Derecho**, **Ciencias Sociales**), mientras que en el eje **Y** se muestra la proporción porcentual de estudiantes en cada carrera.

- **Población**: Las barras azules indican la proporción de cada carrera dentro de la población completa de 100 estudiantes. Esto refleja la distribución original de las carreras en la población.

- **Muestra Estratificada**: Las barras naranjas representan la proporción de cada carrera en la muestra estratificada de 20 estudiantes. Dado que se ha aplicado un muestreo estratificado proporcional, las proporciones en la muestra reflejan de manera fiel las proporciones observadas en la población.

Este gráfico demuestra que el **muestreo estratificado** ha logrado mantener la representatividad de cada subgrupo (carrera) en la muestra. Al comparar las barras azules y naranjas, se observa que las proporciones en la muestra son consistentes con las de la población, lo que indica una selección equilibrada y proporcional de cada carrera.

Esta representatividad es crucial para asegurar que las estimaciones y conclusiones derivadas de la muestra sean precisas y reflejen adecuadamente las características de la población total. En contraste, métodos de muestreo no estratificados podrían resultar en desequilibrios y sesgos, afectando la validez de los análisis estadísticos posteriores.

¿Cuándo usar el Muestreo Estratificado?

Este tipo de muestreo es ideal cuando:

- La población es **heterogénea** y se puede dividir en **subgrupos (estratos)** que son internamente homogéneos pero diferentes entre sí.
- Queremos asegurar que cada estrato esté **adecuadamente representado** en la muestra.
- Buscamos **mejorar la precisión** de las estimaciones, especialmente cuando los subgrupos tienen características que afectan el estudio.

Sin embargo, el muestreo estratificado requiere **conocer previamente la estructura** de la población para identificar los estratos, lo que puede no ser viable si no tenemos esta información disponible. En poblaciones pequeñas o homogéneas, puede ser más adecuado el **muestreo aleatorio simple**.

2.3. Comparación entre MAS y Muestreo Estratificado

Vamos a realizar una comparación entre el Muestreo Aleatorio Simple (MAS) y el Muestreo Estratificado. Para ello, tomaremos muestras repetidamente de la población y compararemos la proporción de estudiantes de "Ingeniería" en cada muestra utilizando ambos métodos.

2.3.1. Simulación de Muestras Repetidas

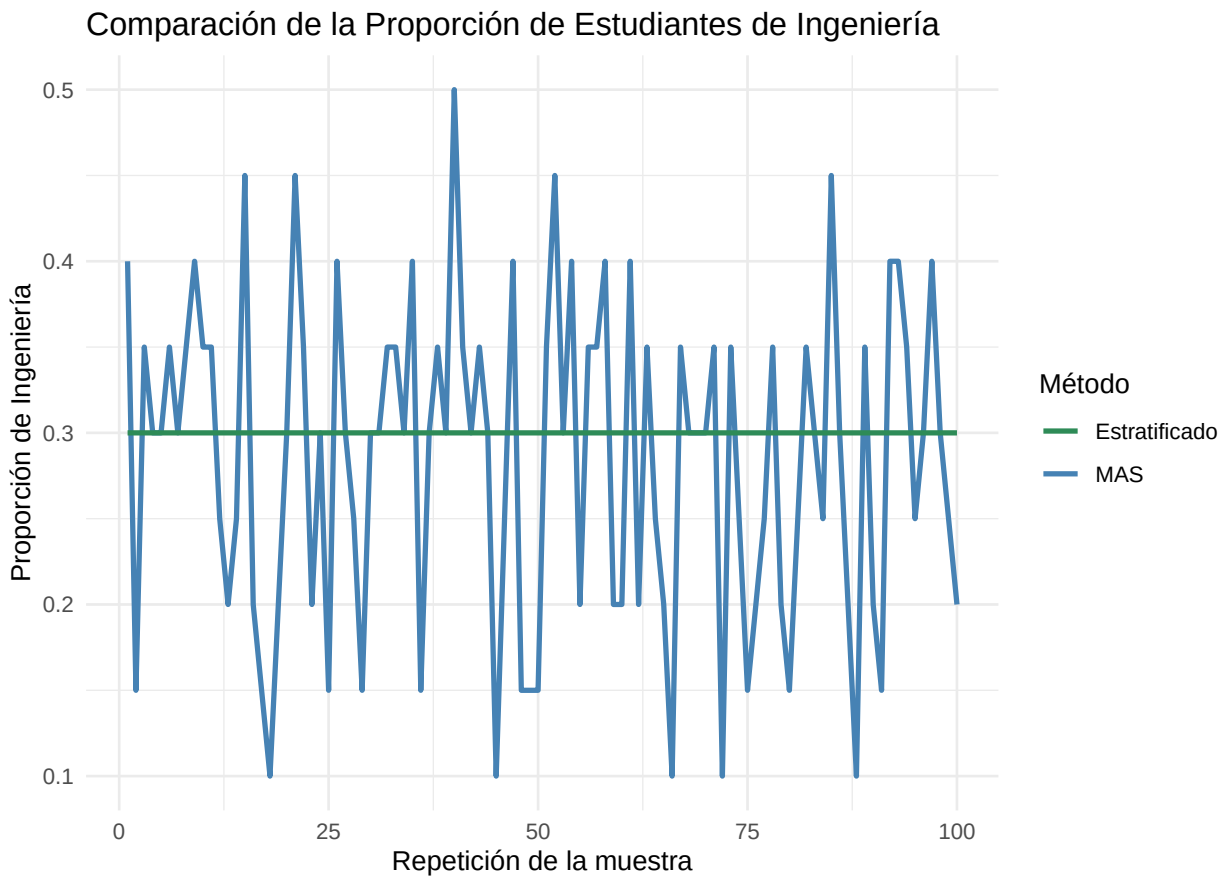
Para entender mejor las diferencias entre MAS y el Muestreo Estratificado, realizaremos una **simulación** en la que extraeremos **100 muestras** utilizando cada método. En cada muestra, calcularemos la **proporción de estudiantes de "Ingeniería"**. Este enfoque nos permitirá observar cómo varían las estimaciones según el método de muestreo utilizado.

Procedimiento de la Simulación

1. **Configuración Inicial**:
   - **Número de Repeticiones** `n_reps`: 100
   - **Tamaño de la Muestran_muestra**: 20 estudiantes
2. **Definición de Funciones de Muestreo**:
   - **MAS**: Selecciona 20 estudiantes al azar de toda la población sin considerar la estructura de la misma.
   - **Estratificado**: Divide la población en estratos (en este caso, carreras) y selecciona una muestra proporcional de cada estrato.

3. **Ejecución de la Simulación**:
   - Utilizamos la función `replicate()` para repetir el proceso de muestreo 100 veces para cada método.
   - Calculamos la proporción de estudiantes de "Ingeniería" en cada muestra obtenida.
4. **Almacenamiento de Resultados**:
   - Creamos un dataframe que contiene las proporciones obtenidas en cada repetición para ambos métodos, lo que nos permitirá compararlos fácilmente.

```r
# Parámetros de la simulación
n_reps <- 100  # Número de repeticiones
n_muestra <- 20  # Tamaño de la muestra total

# Función para obtener la proporción de "Ingeniería" en MAS
mas_sim <- function() {
  muestra_mas <- sample(poblacion$id, size = n_muestra, replace = FALSE)
  prop_ing_mas <- mean(poblacion$carrera[muestra_mas] == "Ingeniería")
  return(prop_ing_mas)
}

# Función para obtener la proporción de "Ingeniería" en Muestreo Estratificado
estrat_sim <- function() {
  muestra_estrat <- poblacion |>
    group_by(carrera) |>
    sample_n(size = round(n_muestra * n() / nrow(poblacion)), replace = FALSE)
  prop_ing_estrat <- mean(muestra_estrat$carrera == "Ingeniería")
  return(prop_ing_estrat)
}

# Simulación
set.seed(123)
resultados_mas <- replicate(n_reps, mas_sim())
resultados_estrat <- replicate(n_reps, estrat_sim())

# Crear un data frame con los resultados
resultados <- data.frame(
  rep = 1:n_reps,
  MAS = resultados_mas,
  Estratificado = resultados_estrat
)

# Graficar los resultados de MAS y Estratificado
ggplot(resultados, aes(x = rep)) +
  geom_line(aes(y = MAS, color = "MAS"), size = 1) +
  geom_line(aes(y = Estratificado, color = "Estratificado"), size = 1) +
  labs(title = "Comparación de la Proporción de Estudiantes de Ingeniería",
       x = "Repetición de la muestra",
       y = "Proporción de Ingeniería",
       color = "Método") +
  scale_color_manual(values = c("MAS" = "steelblue", "Estratificado" = "seagreen")) +
  theme_minimal()
```



Este gráfico de líneas compara la proporción de estudiantes de **Ingeniería** obtenida mediante dos métodos de muestreo: **Muestreo Aleatorio Simple (MAS)** y **Muestreo Estratificado**, a lo largo de 100 repeticiones de muestras.

- **Eje X**: Representa el número de repeticiones de la muestra (de 1 a 100).
- **Eje Y**: Representa la proporción de estudiantes de Ingeniería en cada muestra.

Elementos del Gráfico

- **Línea Azul (MAS)**: Muestra la proporción de Ingeniería obtenida a través del Muestreo Aleatorio Simple en cada una de las 100 repeticiones.
- **Línea Verde (Estratificado)**: Muestra la proporción de Ingeniería obtenida a través del Muestreo Estratificado en cada una de las 100 repeticiones.

Observaciones

1. **Variabilidad**:
   - **MAS**: La línea azul presenta una mayor variabilidad en la proporción de Ingeniería entre las diferentes muestras. Esto indica que las proporciones obtenidas pueden fluctuar significativamente alrededor de la proporción real en la población.
   - **Estratificado**: La línea verde muestra una menor variabilidad, con proporciones de Ingeniería más consistentes y cercanas a la proporción real de la población.
2. **Consistencia**:

- **MAS**: Al ser un método completamente aleatorio, las muestras pueden no reflejar de manera precisa la estructura de la población, lo que resulta en fluctuaciones más amplias en las proporciones.
- **Estratificado**: Al garantizar una representación proporcional de cada estrato (en este caso, cada carrera), el muestreo estratificado logra una mayor consistencia en las proporciones obtenidas en cada repetición.

Implicaciones

- **Precisión y Representatividad**:
  - El **muestreo estratificado** proporciona una representación más precisa y consistente de subgrupos específicos dentro de la población, reduciendo la variabilidad de las estimaciones en comparación con el Muestreo Aleatorio Simple.
- **Elección del Método de Muestreo**:
  - Cuando es importante asegurar la representatividad de subgrupos específicos (como una carrera particular), el muestreo estratificado es preferible debido a su mayor consistencia y menor variabilidad en las estimaciones.
  - Por otro lado, el **MAS** puede ser adecuado en situaciones donde la población es homogénea o cuando se busca simplicidad, aunque a costa de una mayor variabilidad en las estimaciones.

Conclusión

Este gráfico demuestra que el **muestreo estratificado** es más efectivo para obtener proporciones consistentes y representativas de subgrupos específicos dentro de una población. Al reducir la variabilidad en las estimaciones, este método mejora la precisión de las inferencias estadísticas realizadas a partir de la muestra, lo que es fundamental para obtener conclusiones fiables en estudios estadísticos.

2.4. Muestreo por Conglomerados

El **muestreo por conglomerados** se utiliza cuando la población está naturalmente dividida en grupos o **conglomerados** (por ejemplo, aulas, escuelas, barrios). En lugar de muestrear individuos de toda la población, se seleccionan aleatoriamente algunos conglomerados y se muestrean todos (o una muestra) los individuos dentro de esos conglomerados seleccionados. Este método es útil cuando es difícil o costoso enumerar a todos los individuos de la población, pero es más fácil listar los conglomerados.

> **Ejemplo**: Supongamos que queremos muestrear estudiantes de diferentes aulas en una escuela. En lugar de seleccionar estudiantes al azar de toda la escuela, primero seleccionamos algunas aulas al azar y luego muestreamos a todos los estudiantes dentro de esas aulas seleccionadas.

```r
# Simulación de una población de estudiantes clasificados por aula
set.seed(123)
poblacion <- data.frame(
  id = 1:100,  # Identificación de estudiantes
  aula = sample(paste("Aula", 1:10), 100, replace = TRUE)
)

# Tamaño de la muestra total
n_muestra <- 20

# Número de conglomerados a seleccionar (por ejemplo, 2 aulas)
```

```r
n_conglomerados <- 2

# Seleccionar aleatoriamente los conglomerados
conglomerados_seleccionados <- sample(unique(poblacion$aula), size = n_conglomerados)

# Seleccionar todos los estudiantes dentro de los conglomerados seleccionados
muestra_conglomerados <- poblacion |>
  filter(aula %in% conglomerados_seleccionados)
```

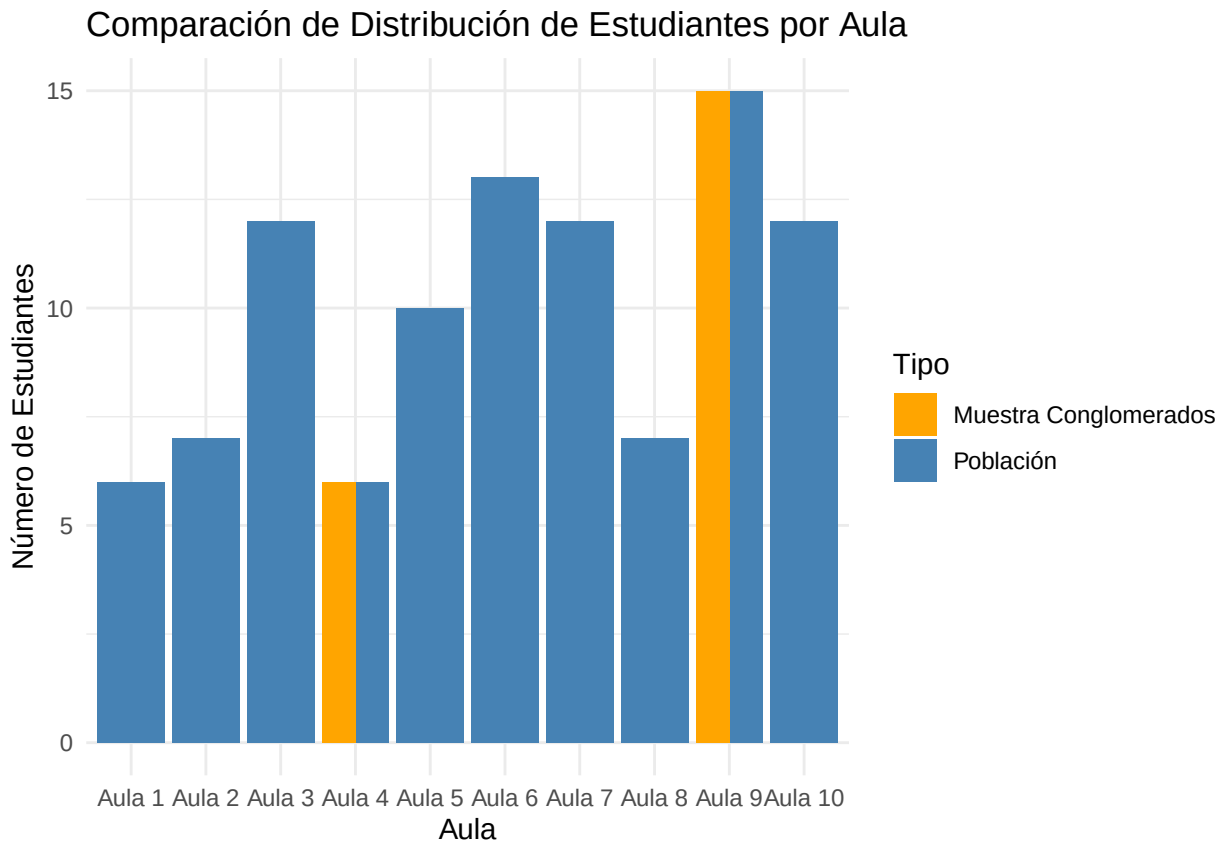
En este ejemplo, realizamos un **muestreo por conglomerados** en una población de 100 estudiantes distribuidos en 10 aulas.

1. **Crear la población**: Creamos un dataframe llamado `poblacion` que contiene 100 estudiantes, cada uno con un `id` del 1 al 100 y asignándolos aleatoriamente a una de las 10 aulas usando la función `sample()`.
2. **Determinar el tamaño de la muestra**: Definimos que queremos una muestra total de 20 estudiantes.
3. **Determinar el número de conglomerados a seleccionar**: Decidimos seleccionar 2 aulas al azar de las 10 disponibles.
4. **Seleccionar los conglomerados**: Usamos la función `sample()` para seleccionar aleatoriamente 2 aulas de las 10.
5. **Seleccionar todos los estudiantes dentro de los conglomerados seleccionados**: Filtramos el dataframe `poblacion` para incluir solo a los estudiantes que están en las aulas seleccionadas, formando así la muestra por conglomerados.

Este método asegura que la muestra esté concentrada en los conglomerados seleccionados, lo que puede ser eficiente en términos de costos y logística cuando los conglomerados son internamente heterogéneos pero homogéneos entre sí.

```r
# Contar el número de estudiantes por aula en la población
poblacion_aulas <- poblacion |>
  count(aula) |>
  mutate(Tipo = "Población")

# Contar el número de estudiantes por aula en la muestra por conglomerados
muestra_aulas <- muestra_conglomerados |>
  count(aula) |>
  mutate(Tipo = "Muestra Conglomerados")

# Unir los datos para el gráfico
comparacion_aulas <- bind_rows(poblacion_aulas, muestra_aulas)

# Definir el orden de las aulas correctamente, como "Aula 1", "Aula 2", etc.
orden_aulas <- paste("Aula", 1:10)

# Convertir la variable 'aula' en factor con el orden deseado
comparacion_aulas$aula <- factor(comparacion_aulas$aula, levels = orden_aulas)

# Crear el gráfico de barras
ggplot(comparacion_aulas, aes(x = aula, y = n, fill = Tipo)) +
  geom_bar(stat = "identity", position = "dodge") +
```

```
  labs(title = "Comparación de Distribución de Estudiantes por Aula",
       x = "Aula",
       y = "Número de Estudiantes",
       fill = "Tipo") +
  scale_fill_manual(values = c("Población" = "steelblue", "Muestra Conglomerados" = "orange")) +
  theme_minimal()
```



Este gráfico de barras compara la distribución de estudiantes por aula en la población total y en la muestra seleccionada por conglomerados. Podemos ver cómo los estudiantes se agrupan en los conglomerados seleccionados, lo que es característico del muestreo por conglomerados.

¿Cuándo usar el Muestreo por Conglomerados?

Este tipo de muestreo es ideal cuando:

- La población está naturalmente dividida en **grupos o conglomerados**.
- Es **costoso o difícil enumerar** a todos los individuos de la población.
- Los conglomerados son **internamente heterogéneos** pero **homogéneos entre sí**.

Sin embargo, puede no ser el método adecuado si los conglomerados son **internamente homogéneos**, ya que podría aumentar la **variabilidad de las estimaciones**.

2.5. Muestreo Sistemático

El **muestreo sistemático** se utiliza cuando la población está **ordenada** de cierta manera (por ejemplo, lista alfabética, numérica). En lugar de seleccionar elementos al azar de toda la población, se elige un punto

de inicio aleatorio y luego se seleccionan elementos a intervalos regulares a partir de ese punto. Este método es útil cuando se desea una **distribución uniforme** de la muestra a lo largo de la población y cuando es más sencillo aplicar una regla sistemática de selección que un muestreo completamente aleatorio.

Para implementar el muestreo sistemático, se utiliza el parámetro k, conocido como el **intervalo de muestreo**, que se calcula dividiendo el tamaño de la población N entre el tamaño de la muestra deseada n:

$$k = \frac{N}{n}$$

Donde N es el tamaño total de la población y n el de la muestra.

Pasos para el Muestreo Sistemático

1. **Calcular el intervalo k**:

$$k = \frac{N}{n}$$

2. **Seleccionar un punto de inicio aleatorio**:
   Se elige un número aleatorio entre 1 y k para determinar el primer elemento de la muestra.

3. **Seleccionar cada k-ésimo elemento**:
   A partir del punto de inicio, se selecciona cada k-ésimo elemento para formar la muestra.

> **Ejemplo**: Supongamos que queremos seleccionar cada 5º estudiante de una lista ordenada de 100 estudiantes. Primero, calculamos el intervalo `k = 5` y luego seleccionamos un punto de inicio aleatorio entre 1 y 5. A partir de ese punto, tomamos cada 5º estudiante para formar la muestra.

```r
# Simulación de una población ordenada de estudiantes
set.seed(123)
poblacion <- data.frame(
  id = 1:100,  # Identificación de estudiantes
  nombre = paste("Estudiante", 1:100)
)

# Tamaño de la muestra total
n_muestra <- 20

# Calcular el intervalo k
k <- floor(nrow(poblacion) / n_muestra)

# Seleccionar un punto de inicio aleatorio entre 1 y k
inicio <- sample(1:k, 1)

# Seleccionar cada k-ésimo estudiante a partir del punto de inicio
indices <- seq(from = inicio, to = nrow(poblacion), by = k)
muestra_sistematico <- poblacion[indices, ]
```

En este ejemplo, realizamos un **muestreo sistemático** de una población ordenada de 100 estudiantes para seleccionar una muestra de 20 estudiantes.

1. **Crear la población**: Creamos un dataframe llamado `poblacion` que contiene 100 estudiantes, cada uno con un `id` del 1 al 100 y un `nombre` correspondiente.

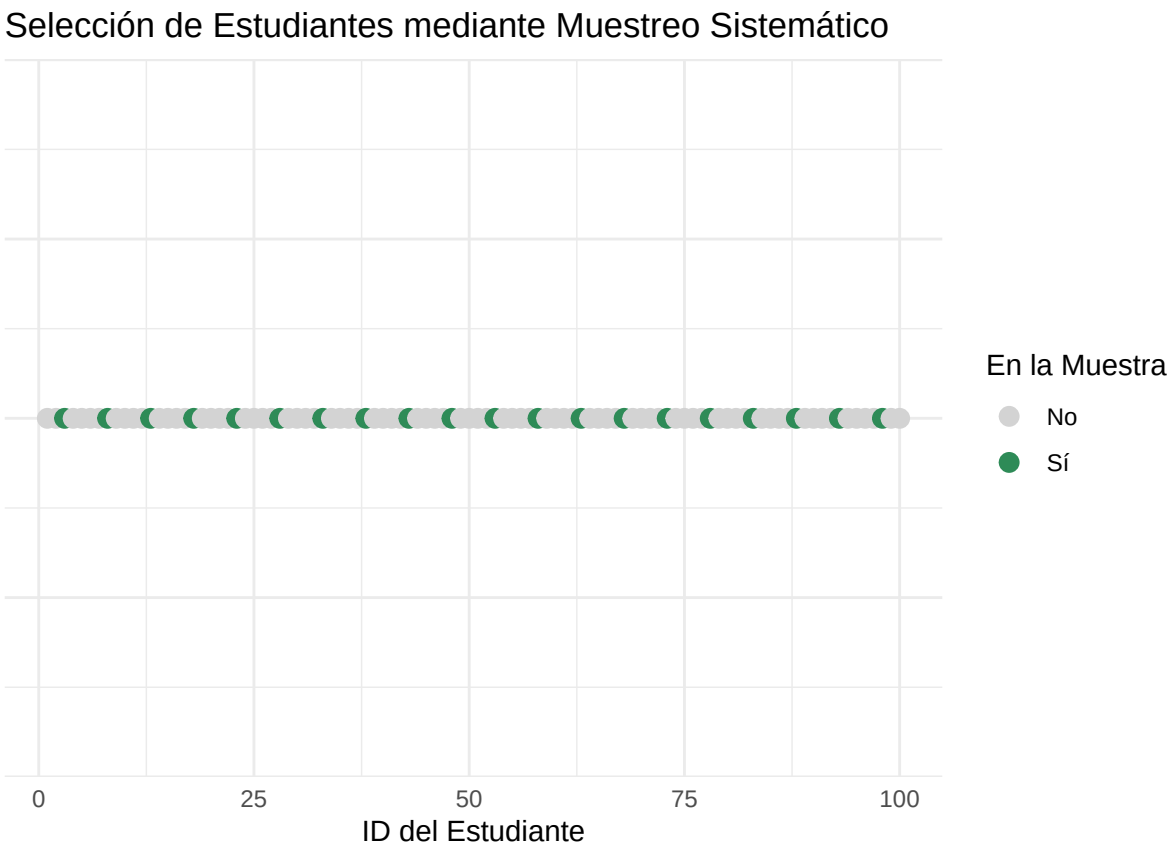
2. **Determinar el tamaño de la muestra**: Definimos que queremos una muestra de 20 estudiantes.

3. **Calcular el intervalo `k`**: Calculamos el intervalo `k` dividiendo el tamaño de la población entre el tamaño de la muestra, es decir, `k = floor(100 / 20) = 5`. Esto significa que seleccionaremos cada 5^{o} estudiante.

4. **Seleccionar el punto de inicio**: Elegimos aleatoriamente un punto de inicio entre 1 y `k` (1 y 5). Por ejemplo, si el punto de inicio es 3, seleccionaremos estudiantes con IDs 3, 8, 13, . . . , 98.

5. **Seleccionar cada `k`-ésimo estudiante**: Usamos la función `seq()` para generar una secuencia de índices desde el punto de inicio hasta el tamaño de la población, con un intervalo de `k`. Luego, seleccionamos esos estudiantes para formar la muestra sistemática.

```r
# Añadir una columna para indicar si el estudiante está en la muestra
poblacion$Muestra_Sistematico <- ifelse(poblacion$id %in% muestra_sistematico$id, "Sí", "No")

ggplot(poblacion, aes(x = id, y = 1)) +
  geom_point(aes(color = Muestra_Sistematico), size = 3) +
  scale_color_manual(values = c("Sí" = "seagreen", "No" = "lightgray")) +
  labs(title = "Selección de Estudiantes mediante Muestreo Sistemático",
       x = "ID del Estudiante",
       y = "",
       color = "En la Muestra") +
  theme_minimal() +
  theme(axis.text.y = element_blank(),
        axis.ticks.y = element_blank())
```



Este gráfico muestra la selección de estudiantes mediante **muestreo sistemático** en una población ordenada. Podemos ver cómo los estudiantes son seleccionados a intervalos regulares `k`, lo que garantiza una **distribución uniforme** de la muestra a lo largo de la población. Esto es característico del muestreo sistemático, que busca una representación equitativa mediante una selección sistemática de elementos.

¿Cuándo usar el Muestreo Sistemático?

Este tipo de muestreo es ideal cuando:

- La población está **ordenada** de alguna manera (por ejemplo, lista alfabética, numérica).
- Se desea una **distribución uniforme** de la muestra a lo largo de la población.
- Es más **sencillo y rápido** que el muestreo aleatorio simple, especialmente para poblaciones grandes.

Sin embargo, puede no ser el método adecuado si existe un **patrón periódico** en la población que coincide con el intervalo de muestreo, lo que podría sesgar la muestra.

3. Estimadores

En esta sección, profundizaremos en los conceptos fundamentales relacionados con los **estimadores** y sus **propiedades** en el contexto del muestreo estadístico. Comprender estos conceptos es esencial para realizar inferencias precisas sobre una población a partir de una muestra.

3.1. ¿Qué es un Estimador?

Un **estimador** es una regla, fórmula o función utilizada para calcular una **estimación** de un **parámetro poblacional** basándose en los datos de una **muestra**. Los estimadores son funciones de los datos muestrales

y permiten inferir características de la población completa sin necesidad de examinar todos sus elementos.

3.1.1. Características de los Estimadores

- **Función de la Muestra**: Un estimador es una función matemática que toma los datos de la muestra como entrada y produce una estimación del parámetro poblacional. Por ejemplo, la media muestral $\bar{x}$ se calcula sumando todos los valores de la muestra y dividiendo entre el tamaño de la muestra.
- **Objetivo**: El objetivo principal de un estimador es proporcionar una estimación lo más cercana posible al verdadero parámetro poblacional. Para lograr esto, los estimadores deben poseer ciertas propiedades que garantizan su calidad y precisión.
- **Estimación vs Parámetro**:
  - **Estimación**: Es el valor calculado a partir de la muestra, como la media muestral $\bar{x}$ o la proporción muestral $\hat{p}$.
  - **Parámetro**: Es el valor real en la población, como la media poblacional μ o la proporción poblacional p.

3.2. Estimadores Comunes

A continuación, se presentan algunos de los estimadores más utilizados en estadística, junto con sus fórmulas y descripciones:

Media Muestral $\bar{x}$

La **media muestral** es uno de los estimadores más comunes para la **media poblacional** μ. Representa el promedio de los valores en la muestra.

$$\bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i$$

- n: Tamaño de la muestra.
- x_i: Valores individuales en la muestra.

Interpretación: La media muestral proporciona una estimación central de los datos de la muestra y, bajo ciertas condiciones, es un buen estimador de la media poblacional.

Proporción Muestral $\hat{p}$

La **proporción muestral** estima la **proporción poblacional** p. Es especialmente útil en estudios de encuestas y análisis de características categóricas.

$$\hat{p} = \frac{\text{Número de éxitos en la muestra}}{n}$$

- **Número de éxitos**: Cantidad de observaciones que cumplen con la característica de interés.
- n: Tamaño de la muestra.

Interpretación: La proporción muestral indica la fracción de la muestra que posee una determinada característica, proporcionando una estimación directa de la proporción en la población.

Cuasivarianza s^2

La **cuasivarianza** estima la **varianza poblacional** σ^2. Mide la dispersión de los datos alrededor de la media muestral.

$$s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

- n: Tamaño de la muestra.
- x_i: Valores individuales en la muestra.
- $\bar{x}$: Media muestral.

Interpretación: La cuasivarianza cuantifica la variabilidad de los datos en la muestra, proporcionando información sobre la dispersión de los valores individuales respecto a la media.

3.3. Propiedades de los Estimadores

Los estimadores poseen ciertas **propiedades** que determinan su calidad y adecuación para inferir parámetros poblacionales. Las propiedades más importantes son:

3.3.1. Insesgadez

Un estimador es **insesgado** si su valor esperado es igual al parámetro que estima. Es decir, en promedio, el estimador no sobreestima ni subestima el parámetro poblacional.

$$E(\hat{\theta}) = \theta$$

> **Ejemplo**: La media muestral $\bar{x}$ es un estimador insesgado de la media poblacional μ. Esto implica que, en promedio, la media muestral coincide con la media poblacional, garantizando que $\bar{x}$ no tiende a sobrestimar ni subestimar el valor real de μ. Para ilustrar la **insesgadez** de la media muestral, realizaremos una simulación en R con 1000 repeticiones de una distribución normal $N(50, 10)$, donde compararemos la media de las medias muestrales con la media poblacional para verificar que ambas son prácticamente iguales.

A continuación, presentamos una simulación que verifica esta propiedad.

```r
# Simulación para demostrar la insesgadez de la media muestral
set.seed(123)        # Fijar la semilla para reproducibilidad
n <- 30              # Tamaño de la muestra
mu <- 50             # Media poblacional
sigma <- 10          # Desviación estándar poblacional
reps <- 1000         # Número de repeticiones

# Generar múltiples muestras y calcular sus medias
medias_muestrales <- replicate(reps, {
  muestra <- rnorm(n, mean = mu, sd = sigma)  # Generar una muestra de tamaño n
  mean(muestra)                               # Calcular la media muestral
})

# Calcular la media de las medias muestrales
media_de_medias <- mean(medias_muestrales)
```

Configuración Inicial:

- `set.seed(123)`: Fija la semilla de los números aleatorios para asegurar que los resultados sean reproducibles.
- `n <- 30`: Define el tamaño de cada muestra.
- `mu <- 50`: Establece la media poblacional.
- `sigma <- 10`: Define la desviación estándar poblacional.
- `reps <- 1000`: Determina el número de repeticiones de la simulación.

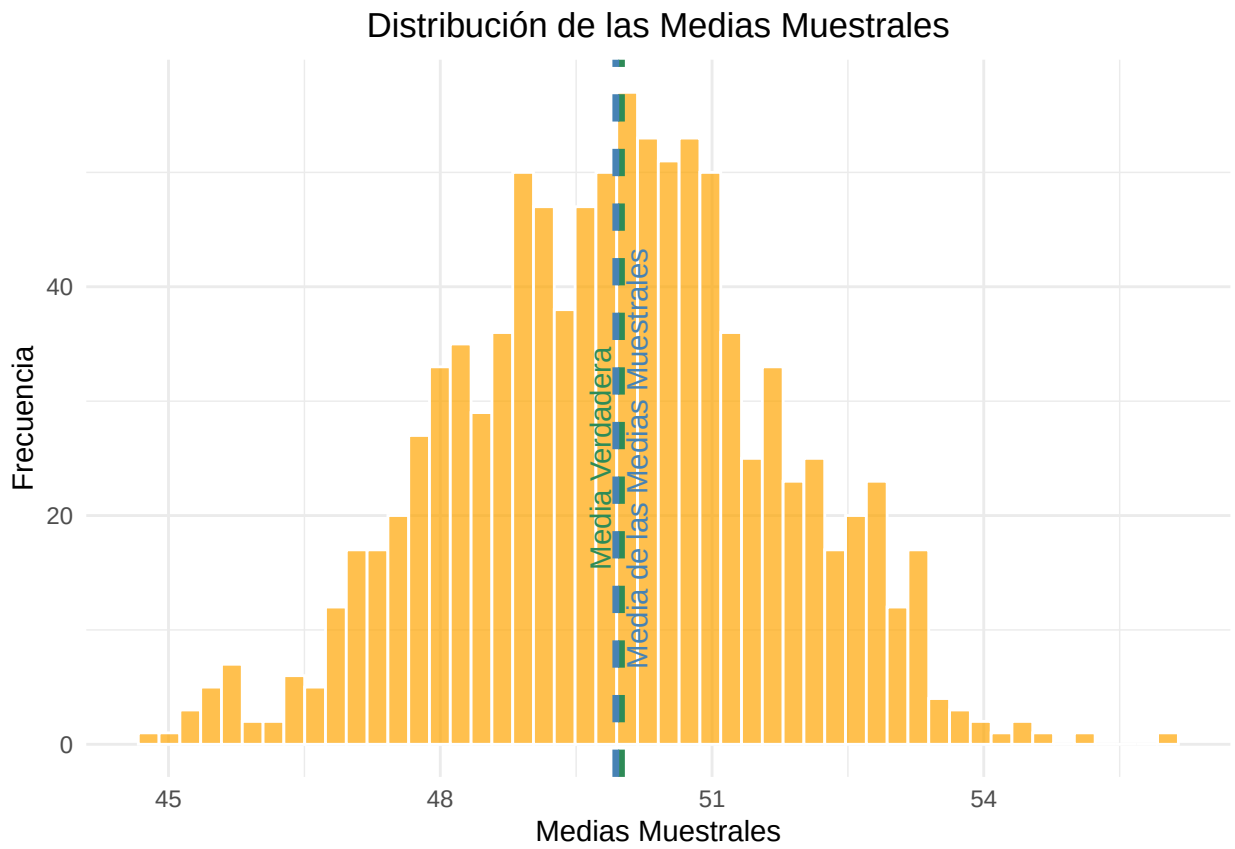
Generación de Muestras y Cálculo de Medias:

- `replicate(reps, { ... })`: Repite el proceso de generación de muestras y cálculo de sus medias 1000 veces.
  - Dentro de `replicate`:
    - `rnorm(n, mean = mu, sd = sigma)`: Genera una muestra aleatoria de tamaño `n` de una distribución normal con media `mu` y desviación estándar `sigma`.
    - `mean(muestra)`: Calcula la media de la muestra generada.

Cálculo de la Media de las Medias Muestrales:

- `media_de_medias <- mean(medias_muestrales)`: Calcula la media de todas las medias muestrales obtenidas en las 1000 repeticiones.

```
# Crear un data frame para ggplot
datos <- data.frame(medias_muestrales)

# Calcular la media de las medias muestrales
media_de_medias <- mean(datos$medias_muestrales)

# Definir la media poblacional
media_poblacional <- mu

# Graficar las medias muestrales usando ggplot2
ggplot(datos, aes(x = medias_muestrales)) +
  geom_histogram(bins = 50, fill = "orange", color = "white", alpha = 0.7) +
  geom_vline(aes(xintercept = media_poblacional), color = "seagreen", linetype = "dashed", size = 1.2) +
  geom_vline(aes(xintercept = media_de_medias), color = "steelblue", linetype = "dashed", size = 1.2) +
  labs(title = "Distribución de las Medias Muestrales",
       x = "Medias Muestrales",
       y = "Frecuencia") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5)) +
  annotate("text", x = media_poblacional, y = max(table(datos$medias_muestrales)) * 25,
           label = "Media Verdadera", color = "seagreen", angle = 90, vjust = -0.5) +
  annotate("text", x = media_de_medias, y = max(table(datos$medias_muestrales)) * 25,
           label = "Media de las Medias Muestrales", color = "steelblue", angle = 90, vjust = 1.5)
```



El gráfico resultante muestra la distribución de las medias muestrales obtenidas a lo largo de las 1000 repeticiones. Observamos lo siguiente:

- **Histograma**: Representa la frecuencia de las diferentes medias muestrales. La mayoría de las medias muestrales se concentran alrededor de la media poblacional $\mu = 50$, formando una distribución aproximadamente normal debido al **Teorema del Límite Central**.
- **Línea Verde (Media Verdadera)**: Indica la media poblacional $\mu = 50$. Es el valor al cual esperamos que las medias muestrales se acerquen en promedio.
- **Línea Azul (Media de las Medias Muestrales)**: Representa la media de todas las medias muestrales calculadas en la simulación. Debería estar muy cercana a la media poblacional, confirmando que $\bar{x}$ es un estimador insesgado de μ.

Este gráfico visualiza cómo, a pesar de la variabilidad inherente en cada muestra individual, la media de las medias muestrales converge hacia la media poblacional, demostrando la **insesgadez** de la media muestral.

3.3.2. Eficiencia

Entre los estimadores insesgados, el **eficiente** es aquel que tiene la menor varianza posible. Un estimador eficiente proporciona estimaciones más precisas al reducir la dispersión en torno al valor del parámetro poblacional.

$$\text{Var}(\hat{\theta}_1) \leq \text{Var}(\hat{\theta}_2)$$

Ejemplo: Dentro de los estimadores insesgados de la media, la **media muestral** $\bar{x}$ es más eficiente que la **mediana muestral**. Para demostrar la **eficiencia** de la media muestral, compararemos la varianza de ambos estimadores utilizando una simulación de 1000 repeticiones de una distribución normal $N(50, 10)$.

A continuación, presentamos una simulación en R para comparar la varianza de la media y la mediana muestral.

```
# Simulación para comparar la eficiencia de la media y la mediana muestral
set.seed(123)
n <- 30  # Tamaño de la muestra
mu <- 50  # Media poblacional
sigma <- 10  # Desviación estándar poblacional
reps <- 1000  # Número de repeticiones

# Generar muestras y calcular la media y mediana muestral
resultados <- replicate(reps, {
  muestra <- rnorm(n, mean = mu, sd = sigma)
  c(mean = mean(muestra), median = median(muestra))
})

# Extraer las medias y medianas muestrales
medias_muestrales <- resultados[1, ]  # Primer fila: medias
medianas_muestrales <- resultados[2, ]  # Segunda fila: medianas

# Calcular las varianzas
var_media <- var(medias_muestrales)
var_mediana <- var(medianas_muestrales)

# Imprimir las varianzas
var_media
```

```
## [1] 3.15235
```

```
var_mediana
```

```
## [1] 4.75556
```

Configuración Inicial:

- `set.seed(123)`: Fija la semilla de los números aleatorios para asegurar que los resultados sean reproducibles.
- `n <- 30`: Define el tamaño de cada muestra.
- `mu <- 50`: Establece la media poblacional.
- `sigma <- 10`: Define la desviación estándar poblacional.
- `reps <- 1000`: Determina el número de repeticiones de la simulación.

Generación de Muestras y Cálculo de Medias:

- `replicate(reps, { ... })`: Repite el proceso de generación de muestras y cálculo de sus medias 1000 veces.

- Dentro de `replicate`:
  - `rnorm(n, mean = mu, sd = sigma)`: Genera una muestra aleatoria de tamaño `n` de una distribución normal con media `mu` y desviación estándar `sigma`.
  - `mean(muestra)`: Calcula la media de la muestra generada.

Generación de Muestras y Cálculo de Medianas:

- Similar al proceso anterior, pero en lugar de calcular la media, se calcula la mediana de cada muestra.

Cálculo de las Varianzas de los Estimadores:

- `var_media <- var(medias_muestrales)`: Calcula la varianza de las medias muestrales obtenidas en las 1000 repeticiones.
- `var_mediana <- var(medianas_muestrales)`: Calcula la varianza de las medianas muestrales obtenidas en las 1000 repeticiones.

Mostrar las Varianzas:

- Al imprimir `var_media` y `var_mediana`, podemos comparar las varianzas de ambos estimadores.

Para una comparación más detallada de la eficiencia entre la media muestral y la mediana muestral, realizaremos una simulación de bootstrap.

El **bootstrap** es un método estadístico que permite estimar la distribución de un estadístico (como la media o la varianza) a partir de muestras repetidas generadas de la misma muestra original. En lugar de depender de suposiciones teóricas sobre la distribución de los datos, el bootstrap genera muchas réplicas simuladas mediante el muestreo con reemplazo de la muestra original. Esto nos permite construir una estimación empírica de la distribución del estadístico que estamos evaluando.

En este caso, el bootstrap nos permitirá estimar la distribución de las diferencias de varianza entre la media muestral y la mediana muestral, lo que facilitará una comparación más detallada de la eficiencia entre ambos estimadores.

```r
# Configuración inicial
set.seed(123)
n_muestra <- 100  # Tamaño de los grupos de muestreo
reps_bootstrap <- 1000  # Número de repeticiones para el bootstrap

# Inicializar un vector para almacenar las diferencias de varianzas
diferencias_varianzas <- numeric(reps_bootstrap)

# Realizar el muestreo con reemplazo sobre medias_muestrales y medianas_muestrales
for (i in 1:reps_bootstrap) {
  # Generar índices de muestreo con reemplazamiento
  indices <- sample(1:reps, size = n_muestra, replace = TRUE)

  # Muestrear las medias y las medianas usando los mismos índices
  muestra_media <- medias_muestrales[indices]
  muestra_mediana <- medianas_muestrales[indices]

  # Calcular las varianzas de las muestras
  var_media <- var(muestra_media)
  var_mediana <- var(muestra_mediana)
```

```r
  # Almacenar la diferencia de varianzas entre media y mediana
  diferencias_varianzas[i] <- var_media - var_mediana
}

# Crear un dataframe para ggplot
datos_varianzas <- data.frame(diferencias_varianzas)
```

Configuración Inicial:

- `set.seed(123)`: Fija la semilla de los números aleatorios para asegurar la reproducibilidad de la simulación.
- `n_muestra <- 100`: Define el tamaño de cada grupo de muestreo en el bootstrap.
- `reps_bootstrap <- 1000`: Establece el número de repeticiones para el proceso de bootstrap.

Inicialización del Vector de Diferencias de Varianzas:

- `diferencias_varianzas <- numeric(reps_bootstrap)`: Crea un vector vacío para almacenar las diferencias de varianza en cada repetición del bootstrap.

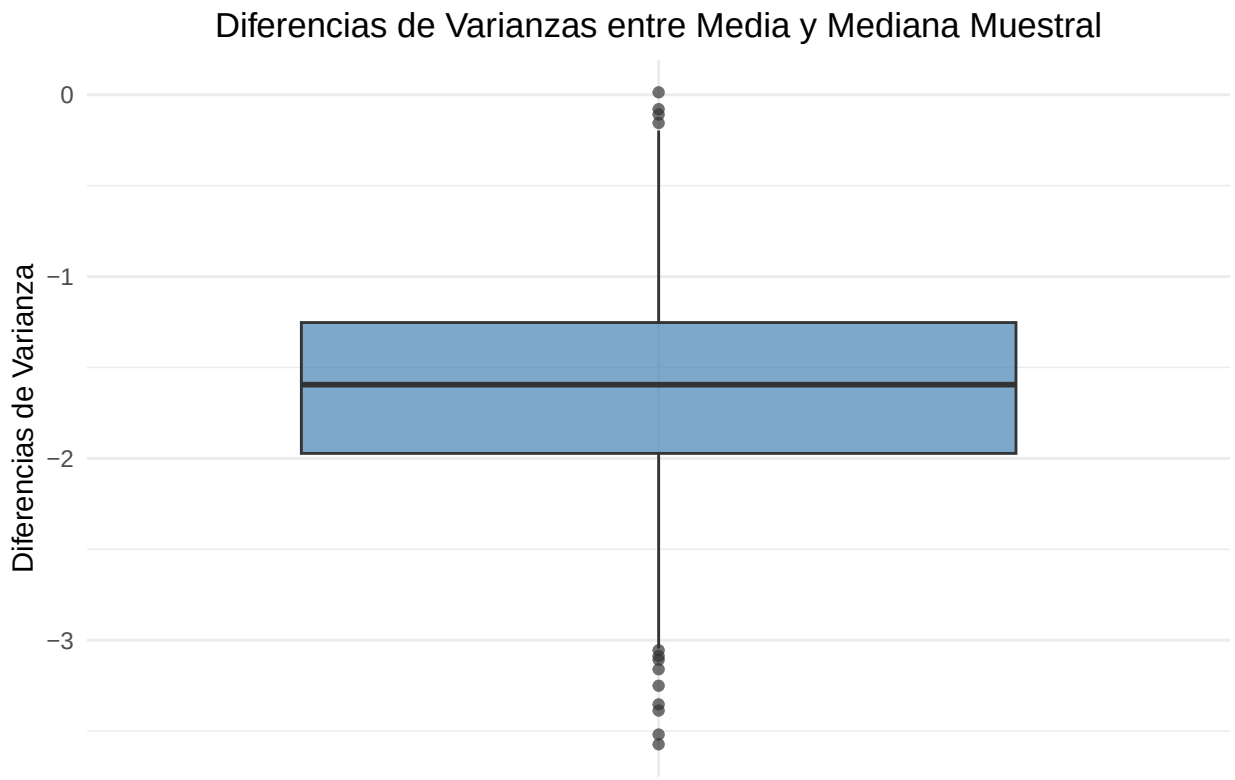
Proceso de Bootstrap:

- **Muestreo con Reemplazo:**
  - `indices <- sample(1:reps, size = n_muestra, replace = TRUE)`: Genera 100 índices aleatorios con reemplazo para muestrear de las `medias_muestrales` y `medianas_muestrales`.
- **Muestreo de Medias y Medianas:**
  - `muestra_media <- medias_muestrales[indices]`: Selecciona las medias muestrales correspondientes a los índices seleccionados.
  - `muestra_mediana <- medianas_muestrales[indices]`: Selecciona las medianas muestrales correspondientes a los mismos índices, asegurando que ambas muestras sean comparables.
- **Cálculo de Varianzas:**
  - `var_media <- var(muestra_media)`: Calcula la varianza de la muestra de medias muestrales.
  - `var_mediana <- var(muestra_mediana)`: Calcula la varianza de la muestra de medianas muestrales.
- **Almacenamiento de Diferencias de Varianzas:**
  - `diferencias_varianzas[i] <- var_media - var_mediana`: Calcula y almacena la diferencia de varianzas entre la media y la mediana para cada repetición.

Creación del Dataframe para la Visualización:

- `datos_varianzas <- data.frame(diferencias_varianzas)`: Convierte el vector de diferencias de varianzas en un dataframe para facilitar su manejo con `ggplot2`.

```r
# Graficar las diferencias de varianzas entre media y mediana con un boxplot
ggplot(datos_varianzas, aes(x = "", y = diferencias_varianzas)) +
  geom_boxplot(fill = "steelblue", alpha = 0.7) +
  labs(title = "Diferencias de Varianzas entre Media y Mediana Muestral",
       y = "Diferencias de Varianza",
       x = "") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5))
```



El gráfico resultante muestra la **distribución de las diferencias de varianza** entre la media y la mediana muestral a lo largo de las 1000 repeticiones. Se destacan los siguientes puntos:

- **Boxplot**: Representa la dispersión de estas diferencias. Cada punto en el gráfico corresponde a una diferencia de varianza calculada a partir de un grupo de muestras. El boxplot también muestra la mediana y los valores atípicos, proporcionando una visión clara de la dispersión.
- **Mediana Negativa**: Sugiere que, en la mayoría de las simulaciones, la varianza de la media muestral es menor que la de la mediana, lo que confirma la mayor eficiencia de la media.
- **Outliers**: Señalan casos donde la diferencia de varianza fue significativamente mayor o menor de lo esperado, posiblemente debido a la variabilidad en muestras pequeñas.

En resumen, el gráfico demuestra que la varianza de la media muestral tiende a ser menor que la de la mediana, confirmando la **eficiencia** de la media como estimador.

3.3.3. Consistencia

Un estimador es **consistente** si, a medida que el tamaño de la muestra aumenta, el estimador converge en probabilidad al verdadero valor del parámetro poblacional.

$$\hat{\theta}_n \xrightarrow{P} \theta \quad \text{cuando} \quad n \to \infty$$

Esto implica que, con muestras suficientemente grandes, el estimador proporcionará una estimación arbitrariamente cercana al parámetro que se desea estimar

Ejemplo: La **media muestral** $\bar{x}$ es un estimador consistente de la **media poblacional** μ. Para demostrar la **consistencia** de $\bar{x}$, realizaremos una simulación en R donde generaremos múltiples muestras de una distribución normal $N(50, 10)$ con diferentes tamaños de muestra, y observaremos cómo la media muestral se aproxima cada vez más a la media poblacional μ conforme aumenta el tamaño de la muestra n. Específicamente, utilizaremos tamaños de muestra que van desde 10 hasta 1000 en incrementos de 10, y realizaremos 1000 repeticiones para cada tamaño de muestra.

```r
# Simulación para demostrar la consistencia de la media muestral
set.seed(123)          # Fijar la semilla para reproducibilidad
mu <- 50               # Media poblacional
sigma <- 10            # Desviación estándar poblacional

# Definir una secuencia de tamaños de muestra
n_values <- seq(10, 10000, by = 10)

# Calcular la media muestral para cada tamaño de muestra utilizando sapply
media_convergente <- sapply(n_values, function(n) {
  # Generar una muestra de tamaño n de una distribución normal N(50, 10)
  muestra <- rnorm(n, mean = mu, sd = sigma)
  # Calcular la media muestral
  mean(muestra)
})

# Crear un data frame para graficar
datos_convergencia <- data.frame(
  Tamaño_Muestra = n_values,
  Media_Muestral = media_convergente
)
```

Configuración Inicial:

- `set.seed(123)`: Fija la semilla de los números aleatorios para asegurar la reproducibilidad de la simulación.
- `mu <- 50`: Establece la media poblacional.
- `sigma <- 10`: Define la desviación estándar poblacional.
- `n_values <- seq(10, 10000, by = 10)`: Crea una secuencia de tamaños de muestra desde 10 hasta 10000, incrementando de 10 en 10.

Proceso de Simulación:

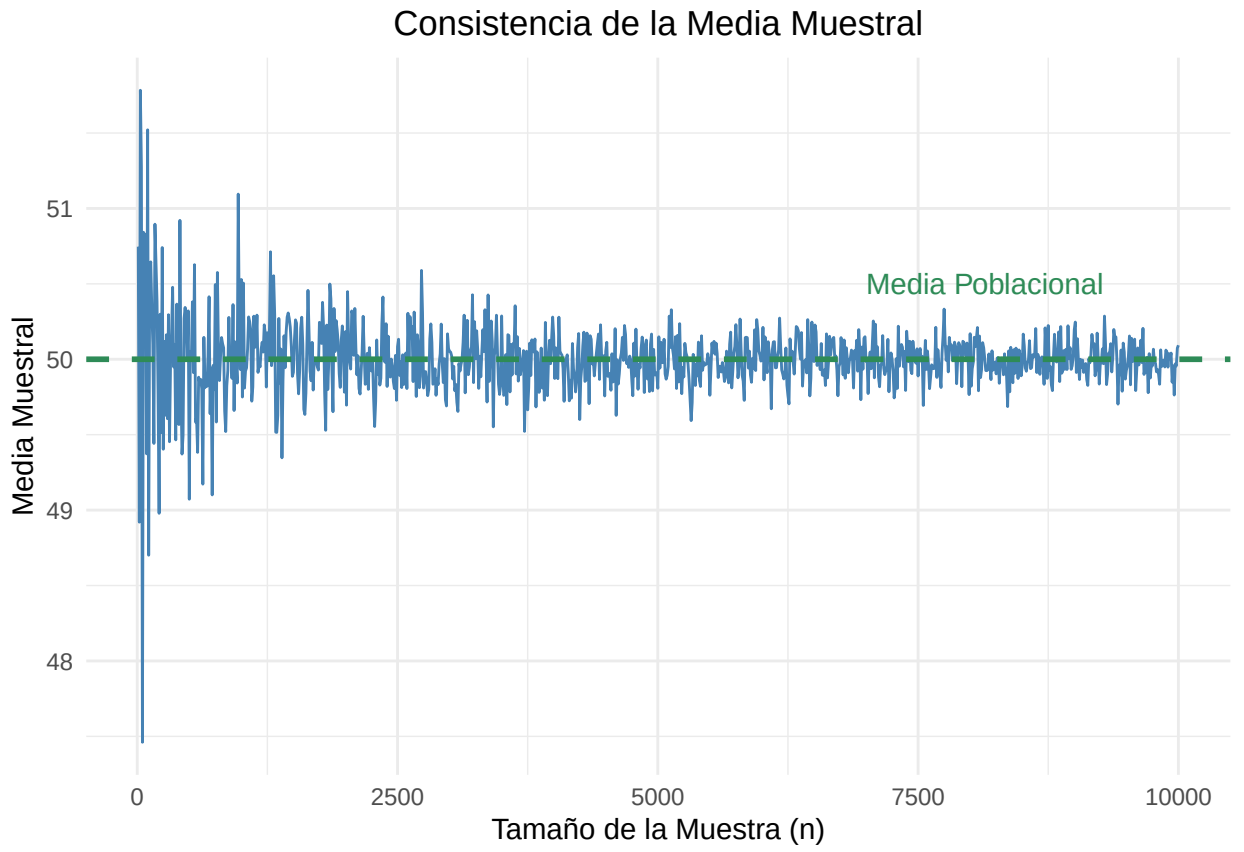
1. Generación de Muestras y Cálculo de Medias Muestrales:

- `media_convergente <- sapply(n_values, function(n) { ... })`: Para cada tamaño de muestra n en `n_values`, se genera una muestra aleatoria de tamaño n de una distribución normal $N(\mu, \sigma)$ y se calcula su media muestral.

2. Almacenamiento de Resultados:

- Se crea un data frame `datos_convergencia` que contiene los tamaños de muestra y las correspondientes medias muestrales.

```r
# Graficar la convergencia de la media muestral hacia la media poblacional

ggplot(datos_convergencia, aes(x = Tamaño_Muestra, y = Media_Muestral)) +
  geom_line(color = "steelblue") +
  geom_hline(aes(yintercept = mu), color = "seagreen", linetype = "dashed", size = 1) +
  labs(title = "Consistencia de la Media Muestral",
       x = "Tamaño de la Muestra (n)",
       y = "Media Muestral") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5)) +
  annotate("text", x = max(n_values) * 0.7, y = mu + 0.5,
           label = "Media Poblacional", color = "seagreen", hjust = 0)
```



El gráfico resultante muestra cómo la media muestral se comporta a medida que aumenta el tamaño de la muestra. Observamos lo siguiente:

- **Línea Azul (Media Muestral):** Representa la media muestral obtenida para cada tamaño de muestra n. A medida que n incrementa, las fluctuaciones en las medias muestrales tienden a reducirse, acercándose más al valor verdadero de la media poblacional $\mu = 50$.
- **Línea Verde Discontinua (Media Poblacional):** Indica el valor verdadero de la media poblacional. Es el punto al cual esperamos que la media muestral converja conforme n aumenta.

La simulación demuestra que, al incrementar el tamaño de la muestra, la **media muestral** $\bar{x}$ se aproxima cada vez más a la **media poblacional** μ. Esto confirma la propiedad de **consistencia** de la media muestral, ya que $\bar{x}$ converge en probabilidad a μ cuando $n \to \infty$.

Además, esta propiedad asegura que, con muestras suficientemente grandes, podemos confiar en que la media muestral proporcionará una estimación precisa del parámetro poblacional, reduciendo la incertidumbre asociada a la variabilidad muestral.

3.3.4. Suficiencia

Un estimador suficiente es aquel que garantiza que dicho estimador utiliza toda la información relevante contenida en la muestra respecto al parámetro que se está estimando. Un estimador suficiente no pierde información útil para la estimación del parámetro poblacional. Formalmente, un estimador $T(X)$ es **suficiente** para un parámetro θ si la distribución condicional de la muestra X dado el estimador $T(X)$ no depende de θ. En otras palabras, $T(X)$ captura toda la información relevante de la muestra sobre θ.

Además, según el **Teorema de Factorización de Fisher**, un estadístico $T(X)$ es suficiente para θ si y solo si la **función de densidad** $f(x|\theta)$ puede factorizarse de la siguiente manera:

$$f(x|\theta) = g(T(x), \theta) \cdot h(x)$$

donde $g(T(x), \theta)$ es una función que depende de la muestra x únicamente a través de $T(x)$ y del parámetro θ, mientras que $h(x)$ es una función que depende de la muestra x pero no del parámetro θ.

Propiedades de Estimadores Suficientes

- **Propiedad Teórica**: La suficiencia se demuestra mediante la estructura de la función de densidad o masa de probabilidad y su factorización, no a través de cálculos empíricos.
- **Limitaciones**: Un estimador suficiente no es necesariamente el mejor en términos de otras propiedades como la varianza mínima. Sin embargo, dentro de la clase de estimadores suficientes, es posible identificar estimadores que son óptimos bajo ciertos criterios.
- **Reducción de Datos**: La suficiencia permite reducir los datos de la muestra a un estadístico suficiente sin pérdida de información relevante para la estimación del parámetro. Esto simplifica el análisis al trabajar con un resumen conciso de los datos originales.

Relación con Otras Propiedades de los Estimadores

- **Insesgadez**: Un estimador suficiente puede ser insesgado, pero la suficiencia no implica necesariamente insesgadez.
- **Eficiencia**: Dentro de la clase de estimadores suficientes, se puede buscar el estimador que minimiza la varianza (estimador eficiente).
- **Consistencia**: La suficiencia no garantiza la consistencia, aunque muchos estimadores consistentes también son suficientes.

Importancia de la Suficiencia

- **Optimización de Datos**: Permite reducir la muestra a un estadístico suficiente sin perder información relevante, facilitando el análisis.
- **Base para Inferencia**: Muchos métodos de inferencia estadística, como los intervalos de confianza y las pruebas de hipótesis, se basan en estimadores suficientes para garantizar la validez de los resultados.

- **Teorema de Basu**: Un teorema importante que establece que cualquier estadístico independiente de un estimador suficiente también es independiente de cualquier función de ese estimador. Esto ayuda a separar la información relevante de la irrelevante en la muestra.

La propiedad de **suficiencia** asegura que un estimador captura toda la información necesaria de la muestra respecto al parámetro de interés, evitando la pérdida de información y optimizando el proceso de estimación. Comprender y identificar estimadores suficientes es esencial para desarrollar inferencias estadísticas sólidas y eficientes.

> **Ejemplo**: La **media muestral** $\bar{x}$ es un estimador suficiente de la **media poblacional** μ en una Distribución Normal $N(\mu, \sigma)$ con σ^2 conocida.

Utilizamos el **Teorema de Factorización de Fisher** para demostrar que $\bar{x}$ es un estadístico suficiente para μ.

Función de Densidad de la Distribución Normal:

La función de densidad para una distribución normal $N(\mu, \sigma^2)$ es:

$$f(x|\mu) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x_i - \mu)^2}{2\sigma^2}\right)$$

Factorización de la Función de Densidad:

Podemos factorizar la función de densidad de la siguiente manera:

$$f(x|\mu) = \left(\frac{1}{(2\pi\sigma^2)^{n/2}} \exp\left(-\frac{n(\bar{x} - \mu)^2}{2\sigma^2}\right)\right) \cdot \exp\left(-\frac{\sum_{i=1}^{n}(x_i - \bar{x})^2}{2\sigma^2}\right)$$

Donde:

- $g(\bar{x}, \mu) = \frac{1}{(2\pi\sigma^2)^{n/2}} \exp\left(-\frac{n(\bar{x}-\mu)^2}{2\sigma^2}\right)$ depende de los datos solo a través de $\bar{x}$ y del parámetro μ.
- $h(x) = \exp\left(-\frac{\sum_{i=1}^{n}(x_i-\bar{x})^2}{2\sigma^2}\right)$ depende de los datos x pero no de μ.

Según el **Teorema de Factorización de Fisher**, esto implica que $\bar{x}$ es un estadístico suficiente para μ, ya que la función de densidad se factoriza en una función que depende de los datos solo a través de $\bar{x}$ y otra que no depende de μ.

3.4. Conclusión

Este laboratorio nos ha permitido comprender y aplicar de manera práctica los conceptos fundamentales de **muestreo** y **estimación estadística**, prestando especial atención a las propiedades de los estimadores y su suficiencia.

A través de ejemplos y simulaciones realizadas en R, abordamos diversos tipos de muestreo, como el **muestreo aleatorio simple**, **estratificado**, **por conglomerados** y **sistemático**, destacando cuándo es conveniente aplicar cada método y cómo influyen en la representatividad de las muestras obtenidas.

Adicionalmente, exploramos las propiedades de los estimadores más comunes:

- **Insesgadez**: Se verificó que la media muestral $\bar{x}$ es un estimador insesgado de la media poblacional μ. En promedio, la media muestral converge al valor real del parámetro, lo cual se demostró mediante simulaciones.

- **Eficiencia**: Comparando la media y la mediana muestral, observamos que la media tiene una menor varianza, lo que la convierte en un estimador más eficiente en nuestra muestra simulada.
- **Consistencia**: Simulaciones con tamaños de muestra crecientes mostraron que la media muestral se aproxima cada vez más a la media poblacional, confirmando su consistencia a medida que el tamaño de la muestra aumenta.
- **Suficiencia**: Aplicando el **Teorema de Factorización de Fisher**, demostramos que la media muestral es un estimador suficiente para la media de una población normal. Este concepto asegura que $\bar{x}$ utiliza toda la información relevante de la muestra sin perder ningún detalle necesario para estimar con precisión el parámetro poblacional.

Implicaciones Prácticas:

- **Selección de métodos de muestreo**: La elección correcta del método de muestreo, dependiendo de la estructura de la población, es crucial para obtener muestras representativas y realizar inferencias precisas.
- **Uso de estimadores eficientes y suficientes**: La eficiencia y suficiencia de los estimadores permiten realizar inferencias estadísticamente óptimas, reduciendo errores y maximizando la información obtenida de los datos muestrales.
- **Decisiones informadas basadas en datos**: Contar con estimadores consistentes, insesgados y eficientes es esencial para garantizar la confiabilidad de los análisis estadísticos, lo cual es fundamental en la toma de decisiones en diversos campos, desde la investigación científica hasta la industria.

En resumen, este laboratorio ha integrado teoría y práctica para mejorar nuestra comprensión del **muestreo** y las **estimaciones**. Al dominar tanto los conceptos teóricos como las simulaciones en R, los estudiantes están mejor preparados para aplicar estas técnicas en contextos reales, asegurando la precisión y validez de los resultados obtenidos en sus análisis estadísticos.