Ejemplo de solución ejercicio prático N°7

Pruebas no paramétricas para variables numéricas

Enunciado

En el trabajo de título de una estudiante del DIINF se reportan tiempos de ejecución (en milisegundos) y la cercanía con la solución óptima (en por ciento) de la mejor solución encontrada con tres versiones de un algoritmo genético para resolver instancias del problema del vendedor viajero disponibles en repositorios públicos. Ahora debe enfrentar el análisis de estos datos, por que está solicitando ayuda de las y los estudiantes de Estadística Inferencial.

Obtengamos los datos en formato ancho.

```
src_dir <- "~/Downloads"
src_basename <- "EP07 Datos.csv"
src_file <- file.path(src_dir, src_basename)
datos <- read.csv(file = src_file, stringsAsFactors = TRUE)
```

Pregunta 1

Observando los datos, la memorista sospecha que hay diferencias significativas en el tiempo de ejecución entre las versiones A y C del algoritmo cuando las instancias tienen 100 o más nodos. ¿Los datos respaldan la intuición de la memorista?

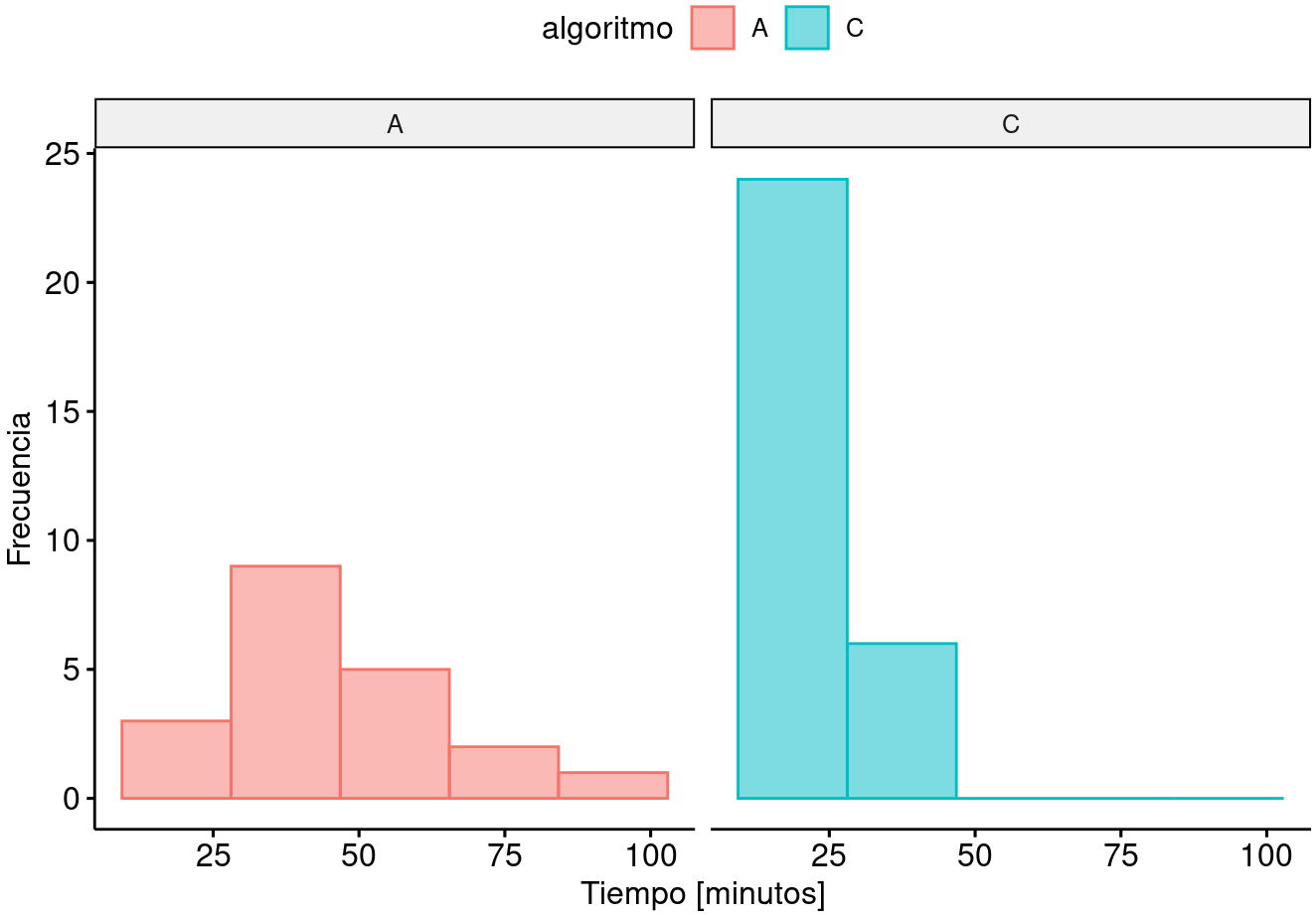
Para responder, filtren los datos para tener las instancias con 100 o más nodos y seleccionen las columnas de los tiempos de ejecución de las versiones A y C en formato ancho. Usando como semilla el valor 213, obtengan muestras aleatorias independientes de 20 tiempos registrados por la versión A y 18 tiempos registrados por la versión C del algoritmo. Realicen un análisis estadístico pertinente (enunciar hipótesis, revisar condiciones, seleccionar prueba) para responder la pregunta planteada, utilizando pruebas no paramétricas de ser necesario.

Primero, filtramos para quedarnos con las instancias que nos interesan y quitar las columnas que no necesitamos.

```
library(dplyr)

nA <- 20
nB <- 30

set.seed(213)
dw1 <- datos %>%
  filter(n.nodos >= 100) %>%
  select(instancia, tiempo.A, tiempo.C) %>%
  sample_n(nA + nB)

dl1 <- data.frame(
  instancia = dw1[["instancia"]],
  algoritmo = c(rep("A", nA), rep("C", nB)),
  tiempo = c(dw1[["tiempo.A"]][1:nA],
             dw1[["tiempo.C"]][(nA+1):(nA+nB)])
)
dl1[["instancia"]] <- factor(dl1[["instancia"]])
dl1[["algoritmo"]] <- factor(dl1[["algoritmo"]])
```

Como es aconsejado, echemos un vistazo a los datos que se están trabajando.

```
library(ggpubr)

# Llevamos los tiempos a minutos para el gráfico
dl1[["tiempo"]] <- dl1[["tiempo"]] / 1000 / 60

# Construimos el gráfico
p1 <- gghistogram(
  dl1,
  x = "tiempo",
  xlab = "Tiempo [minutos]",
  ylab = "Frecuencia",
  color = "algoritmo", fill = "algoritmo",
  bins = 5
)
p1 <- p1 + facet_grid(~ algoritmo)
print(p1)
```



Podemos ver que las muestras no parecen tomadas desde una distribución normal, lo que podemos confirmar con pruebas auxiliares de normalidad.

```
tiempo.A <- dw1[["tiempo.A"]][1:20]
tiempo.C <- dw1[["tiempo.C"]][21:38]

print(shapiro.test(tiempo.A))
```

```
    Shapiro-Wilk normality test

data:  tiempo.A
W = 0.94634, p-value = 0.3149
```

```
print(shapiro.test(tiempo.C))
```

```
    Shapiro-Wilk normality test

data:  tiempo.C
W = 0.79917, p-value = 0.001478
```

Se confirma que los tiempos exhibidos por la versión C están lejos de seguir una distribución normal. Corresponde entonces usar una prueba no paramétrica para analizar estos datos. En este caso, una prueba de **Wilcoxon-Mann-Whitney**, en reemplazo de una prueba t de Student para muestras independientes, con las siguientes hipótesis:

H_0: no hay diferencia en la localización de la tendencia central de los tiempos de ejecución requeridos por ambos algoritmos para instancias con 100 o más nodos.

H_A: la tendencia central de los tiempos de ejecución requeridos por ambos algoritmos para instancias con 100 o más nodos tiene localizaciones distintas.

Verifiquemos que se cumplen las condiciones para aplicar esta prueba no paramétrica con validez:

1. Las observaciones de ambas muestras son independientes. De como hicimos el muestreo más arriba, podemos asegurar que las muestras fueron escogidas de forma aleatoria y no comparten alguna instancia.
2. La escala de medición empleada debe ser a lo menos ordinal. Como la variable en estudio es de tiempo, que corresponde a una medición física, la escala de la medición cumple con condiciones más exigente que solo la ordinal, y tiene sentido hablar de “más/igual/menos tiempo”.

Como se cumplen bien las condiciones, usemos el típico nivel de significación $\alpha = 0{,}05$

Procedamos entonces a realizar la prueba.

```
prueba1 <- wilcox.test(tiempo.A, tiempo.C, paired = FALSE)
print(prueba1)
```

```

    Wilcoxon rank sum exact test

data:  tiempo.A and tiempo.C
W = 310, p-value = 6.053e-05
alternative hypothesis: true location shift is not equal to 0
```

Podemos concluir, entonces, que existe fuerte evidencia en contra de la hipótesis nula ($W = 310, p < 0{,}001$), por lo que la rechazamos en favor de la alternativa. Esto es, la tendecia central de los tiempos que tardan los algoritmos en resolver instancias con 100 o más nodos del problema del vendedor viajero se localizan en valores distintos. Mirando los histogramas de los datos, podemos sugerir que el algoritmo C requiere, en promedio, significativamente menos tiempo de procesamiento.

Pregunta 2

La memorista también sospecha que, al comparar las mismas instancias con 70 a 85 nodos, las mejores soluciones encontradas por las versiones B y C tienen rendimientos distintos. ¿Estará en lo cierto?

Para responder, filtren los datos para tener las instancias que tengan de 70 a 85 nodos y seleccionen las columnas con el mejor rendimiento de las versiones B y C en formato ancho. Usando como semilla el valor 117, obtengan una muestra aleatoria de 24 instancias. Lleven los datos a formato largo y utilicen una prueba no paramétrica apropiada para analizar las muestras obtenidas.

Obtenemos la muestra de datos en formato ancho. Como tenemos que comparar los resultados obtenidos por los algoritmos con *las mismas instancias*, debemos obtener una muestra apareada de 24 observaciones.

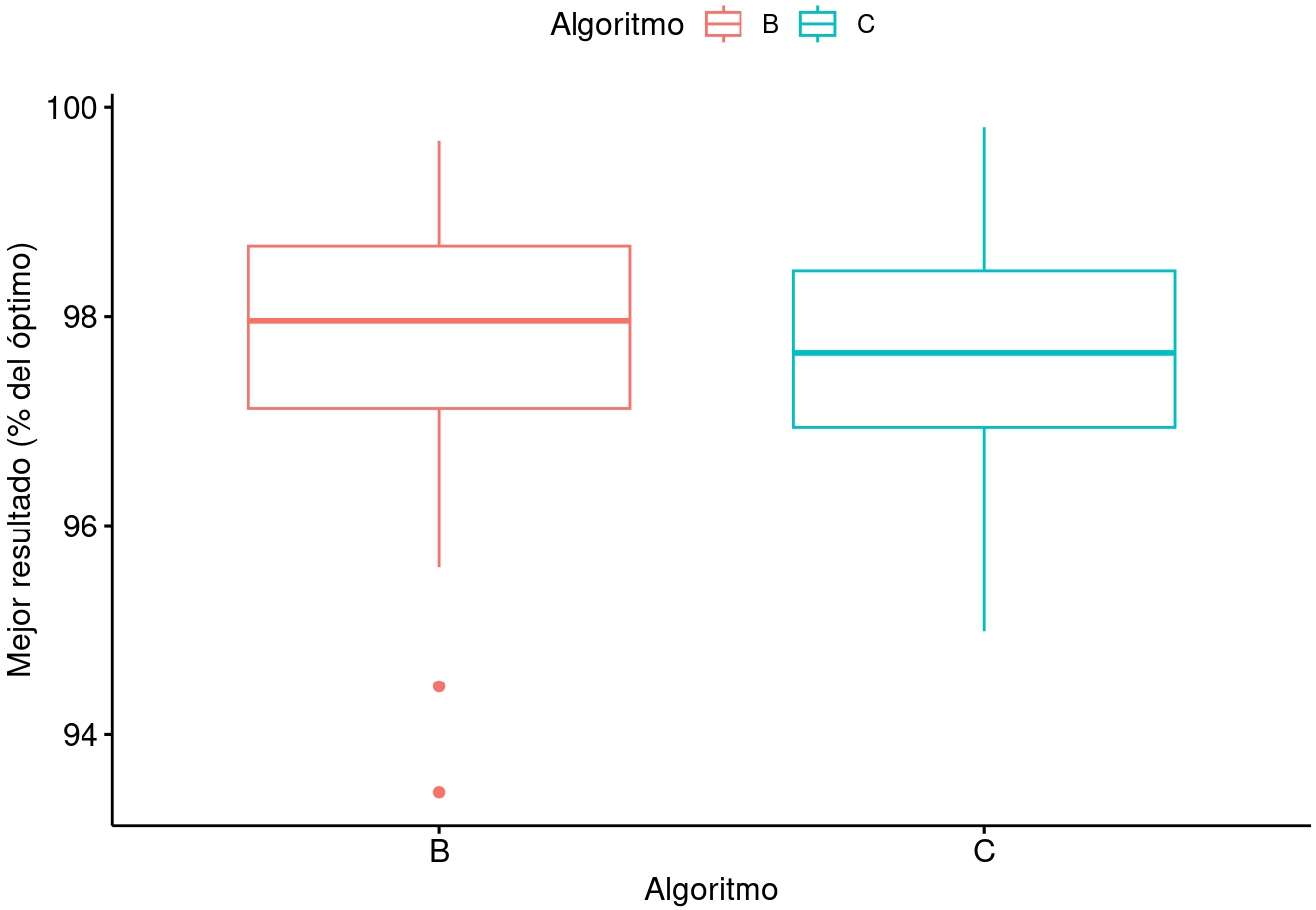
```
set.seed(117)
dw2 <- datos %>%
  filter(n.nodos >= 70 & n.nodos <= 85) %>%
  select(instancia, mejor.B, mejor.C) %>%
  sample_n(24)
```

Llevamos los datos a formato largo

```
library(tidyr)

dl2 <- dw2 %>%
  pivot_longer(
    cols = c("mejor.B", "mejor.C"),
    names_to = "Algoritmo",
    values_to = "Resultado"
  )
dl2[["instancia"]] <- factor(dl2[["instancia"]])
dl2[["Algoritmo"]] <- factor(dl2[["Algoritmo"]], labels = c("B", "C"))
```

Revisemos los datos con un diagrama de cajas

```
p2 <- ggboxplot(
  data = dl2,
  x = "Algoritmo", y = "Resultado",
  xlab = "Algoritmo",
  ylab = "Mejor resultado (% del óptimo)",
  color = "Algoritmo"
)
print(p2)
```



Vemos que los datos para el algoritmo B presentan una leve asimetría y la presencia de valores atípicos. Además, se está trabajando con *porcentajes*, que tiene un rango de valores limitado al intervalo [0, 100]. Sería prudente, entonces, utilizar una prueba no paramétrica para el análisis, como alternativa a una prueba t de Student para muestras apareadas, que en este caso correspondería a una **prueba de rangos con signo de Wilcoxon**, con las siguientes hipótesis:

H_0: no hay diferencia en las mejores soluciones encontradas por las versiones B y C del algoritmo en las mismas instancias con 70 a 85 nodos (tienen igual localización de la tendencia central).

H_A: sí hay diferencias en las mejores soluciones obtenidas por ambas versiones del algoritmo en las mismas instancias con 70 a 85 nodos (sus tendencias centrales se ubican en valores distintos).

Verifiquemos las condiciones:

1. Los pares de observaciones son independientes. Efectivamente, si el experimento fue realizado correctamente por la memorista, cómo se desempeña un algoritmo no debería tener influencia en cómo rinde el segundo.
2. La escala de medición empleada para ambas muestras debe ser a lo menos ordinal. Valores porcentuales cumplen esta condición, pues podemos compararlos y ordenarlos.
3. La escala de medición empleada para las observaciones es intrínsecamente continua. Al tratarse de valores porcentuales, en *teoría* estas mediciones podrían tener un número indeterminado de decimales, por lo que cumpliría con esta condición.

Procedamos con la prueba no paramétrica, ya que se cumplen todas las condiciones.

```
prueba2 <- wilcox.test(
  x = dw2[["mejor.B"]],
  y = dw2[["mejor.C"]],
  paired = TRUE
)
cat("Prueba de rangos con signo de Wilcoxon\n")
```

```
Prueba de rangos con signo de Wilcoxon
```

```
print(prueba2)
```

```

    Wilcoxon signed rank exact test

data:  dw2[["mejor.B"]] and dw2[["mejor.C"]]
V = 167, p-value = 0.6431
alternative hypothesis: true location shift is not equal to 0
```

El resultado de la prueba indica que no hay suficiente evidencia ($V = 167, p = 0{,}643$) para descartar la hipótesis nula en favor de la alternativa. Así, pareciera que el algoritmo B consigue resultados similares a los que se obtienen con el algoritmo C en las mismas instancias con 70 a 85 nodos.

Pregunta 3

La memorista además cree que hay diferencias significativas en el tiempo de ejecución entre las versiones del algoritmo cuando las instancias de prueba tienen 100 o más nodos. ¿Los datos respaldan la intuición de la memorista?

Para responder, filtren los datos para tener las instancias con 100 o más nodos y seleccionen las columnas con los tiempos de ejecución registrados (en formato ancho). Usando como semilla el valor 33, obtengan muestras aleatorias independientes de 12, 13 y 14 tiempos registrados por las versiones A, B y C, respectivamente. Lleven los datos a formato largo y utilicen una prueba no paramétrica para analizar las muestras obtenidas.

Primero, filtramos para quedarnos con las instancias que nos interesan y quitar las columnas que no necesitamos.

```
dw3 <- datos %>%
  filter(n.nodos >= 100) %>%
  select(instancia, tiempo.A, tiempo.B, tiempo.C)
```

Ahora tomamos la muestra solicitada.

```
nA <- 12; nB <- 13; nC <- 14
nI <- nA + nB + nC
```

Es importante que lo hagamos con una sola llamada a la función `sample()`, para evitar que los algoritmos compartan alguna instancia, asegurando así muestras independientes.

```
set.seed(33)
i <- sample(1:nrow(dw3), nI)
seleccion <- dw3[i, ]
tiempo.A <- seleccion[["tiempo.A"]][1:nA]
tiempo.B <- seleccion[["tiempo.B"]][(nA+1):(nA+nB)]
tiempo.C <- seleccion[["tiempo.C"]][(nA+nB+1):nI]
```

Creamos una versión larga de los datos.

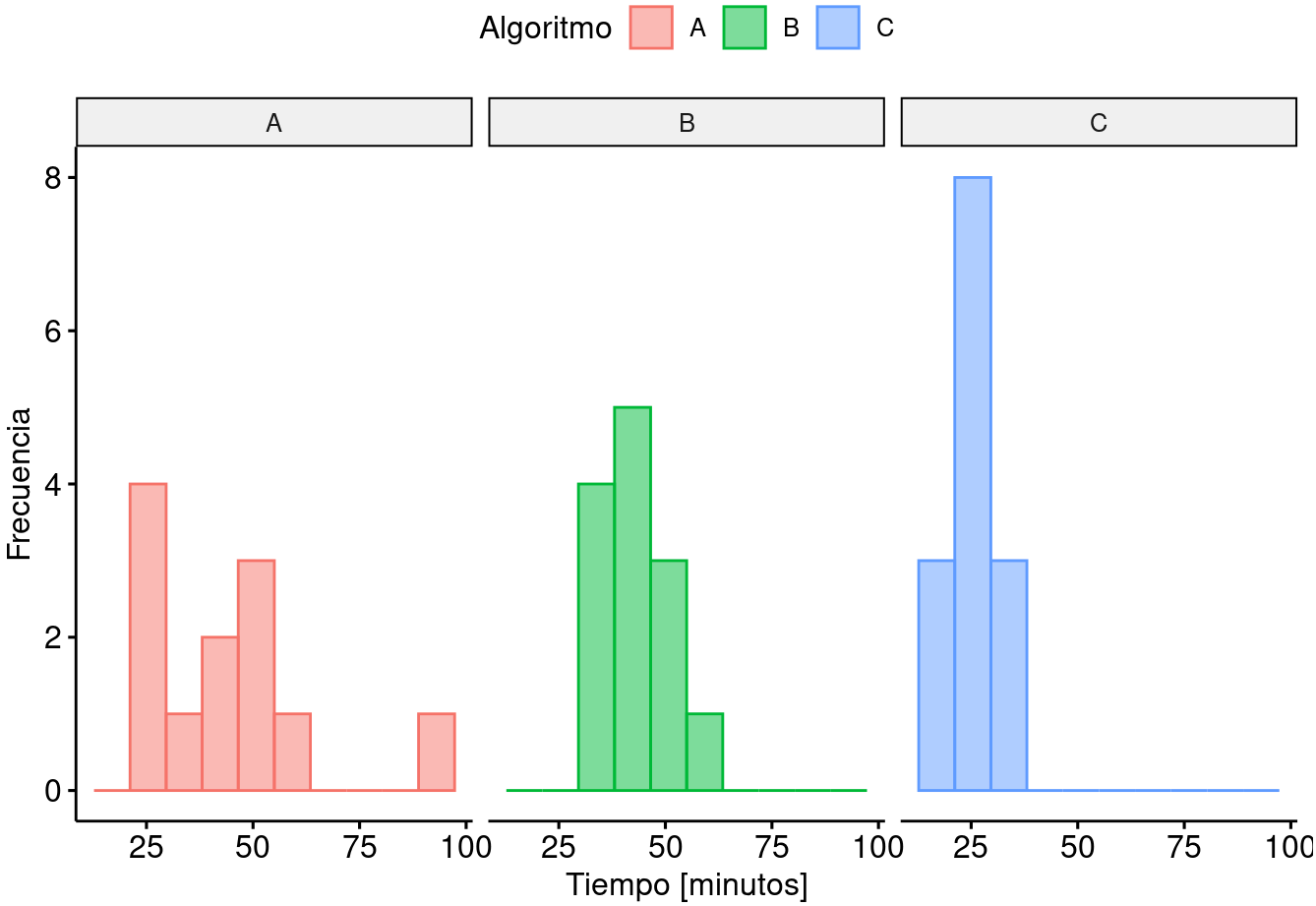
```
dl3 <- data.frame(
  Instancia = seleccion[["instancia"]],
  Algoritmo = c(rep("A", nA), rep("B", nB), rep("C", nC)),
  Tiempo = c(tiempo.A, tiempo.B, tiempo.C)
)
dl3[["Instancia"]] <- factor(dl3[["Instancia"]])
dl3[["Algoritmo"]] <- factor(dl3[["Algoritmo"]])
```

Puesto que cada muestra contiene instancias de prueba distintas, la primera alternativa sería usar ANOVA de una vía para muestras independientes para este análisis. Esta prueba permitiría determinar si la memorista tiene o no razón en pensar que existen diferencias significativas en los tiempos medios de ejecución de los algoritmos.

Verificamos las condiciones:

1. Existe independencia entre las muestras, pues no hay elementos en común y el tiempo que tarda un algoritmo en alguna de las instancias escogida no debería influir en el tiempo que tarda otro algoritmo en otra instancia.
2. También se cumple que la variable dependiente tiene una escala de intervalos iguales, pues es una medición física (tiempo).
3. Veamos si se cumple con las condiciones de normalidad y homocedasticidad por medio de histogramas.

```
# Llevamos los tiempos a minutos para el gráfico
dl3[["Tiempo"]] <- dl3[["Tiempo"]] / 1000 / 60

# Creamos el gráfico
p3 <- gghistogram(
  dl3,
  x = "Tiempo",
  color = "Algoritmo", fill = "Algoritmo",
  bins = 10,
  xlab = "Tiempo [minutos]",
  ylab = "Frecuencia"
)
p3 <- p3 + facet_grid(~ Algoritmo)
print(p3)
```



Podemos ver que las muestras no siguen un comportamiento aproximadamente normal, por lo que no podríamos suponer razonablemente que las poblaciones de donde provienen sí tengan dicha distribución.

Como el problema no parece requerir un valor de las medias estudiadas, sería conveniente bajar las exigencias y optar por una prueba no paramétrica, como se nos indica en el enunciado. En este caso, correspondería una prueba de **Kruskal-Wallis**.

Mirando el gráfico, no podríamos suponer tampoco que las formas de las distribuciones subyacentes sean iguales, por lo que las hipótesis no podrían referirse a medianas. Así, las hipótesis no paramétricas a contrastar serían:

H_0: Los tiempos promedios que tardan los algoritmos en resolver las instancias con 100 o más nodos son similares.

H_A: al menos uno de los algoritmos exhibe tiempos promedios significativamente distintos a uno de los otros dos algoritmos para resolver las instancias con 100 o más nodos.

Verifiquemos las condiciones:

1. Ya sabemos que existe independencia ente las observaciones.
2. También se verifica que la variable independiente tiene más de dos niveles (algoritmos A, B y C).
3. Por último, la escala de la variable dependiente debe ser al menos ordinal, y sabemos que las mediciones físicas cumplen de sobra con tal condición.

Aplicamos la prueba, con el nivel de significación más común.

```
alfa <- 0.05
prueba3 <- kruskal.test(Tiempo ~ Algoritmo, data = dl3)
print(prueba3)
```

```
    Kruskal-Wallis rank sum test

data:  Tiempo by Algoritmo
Kruskal-Wallis chi-squared = 19.802, df = 2, p-value = 5.012e-05
```

La prueba indica que hay suficiente evidencia para rechazar H0 en favor de HA ($\chi^2 = 19{,}8;\ p < 0{,}001$). En consecuencia, podemos concluir con 95% de confianza que los algoritmos difieren significativamente en el tiempo que tardan en resolver las instancias del problema del vendedor viajero con 100 o más nodos.

Como la prueba ómnibus de Kruskal-Wallis detecta diferencias, debemos hacer ahora un **procedimiento post-hoc**. Para ello usaremos múltiples pruebas de Wilcoxon-Mann-Whitney entre pares de grupos, aplicando el ajuste de Benjamini & Hochberg (1995) por tener mayor poder estadístico que varios otros métodos.

```
posthoc1 <- pairwise.wilcox.test(
  dl3[["Tiempo"]],
  dl3[["Algoritmo"]],
  p.adjust.method = "BH",
  paired = FALSE
)
print(posthoc1)
```

```
    Pairwise comparisons using Wilcoxon rank sum exact test

data:  dl3[["Tiempo"]] and dl3[["Algoritmo"]]

  A       B
B 0.64951 -
C 0.00037 1.3e-05

P value adjustment method: BH
```

El procedimiento post-hoc no encuentra diferencias significativas entre los algoritmos A y B ($p = 0{,}650$), pero estos dos algoritmos parecen tardar, en promedio, significativamente más que el algoritmo C ($p < 0{,}001$) en resolver las instancias del problema del vendedor viajero con 100 o más nodos.

Pregunta 4

La memorista también intuye que, al comparar las mismas instancias de prueba con 100 o más nodos, las mejores soluciones encontradas por las diferentes versiones del algoritmo tienen rendimientos distintos. ¿Estará en lo cierto?

Para responder, filtren los datos para tener las instancias con 100 o más nodos y seleccionen las columnas con los mejores rendimientos registrados. Usando como semilla el valor 33, obtengan una muestra aleatoria de 26 instancias. Lleven los datos a formato largo y utilicen una prueba no paramétrica apropiada para analizar los datos obtenidos.

Obtenemos la muestra de datos en formato ancho. Como tenemos que comparar los resultados obtenidos por los algoritmos con *las mismas instancias*, debemos obtener una muestra apareada de 26 observaciones.

```
nD <- 26
set.seed(33)
dw4 <- datos %>%
  filter(n.nodos >= 100) %>%
  select(instancia, mejor.A, mejor.B, mejor.C) %>%
  sample_n(nD)
```

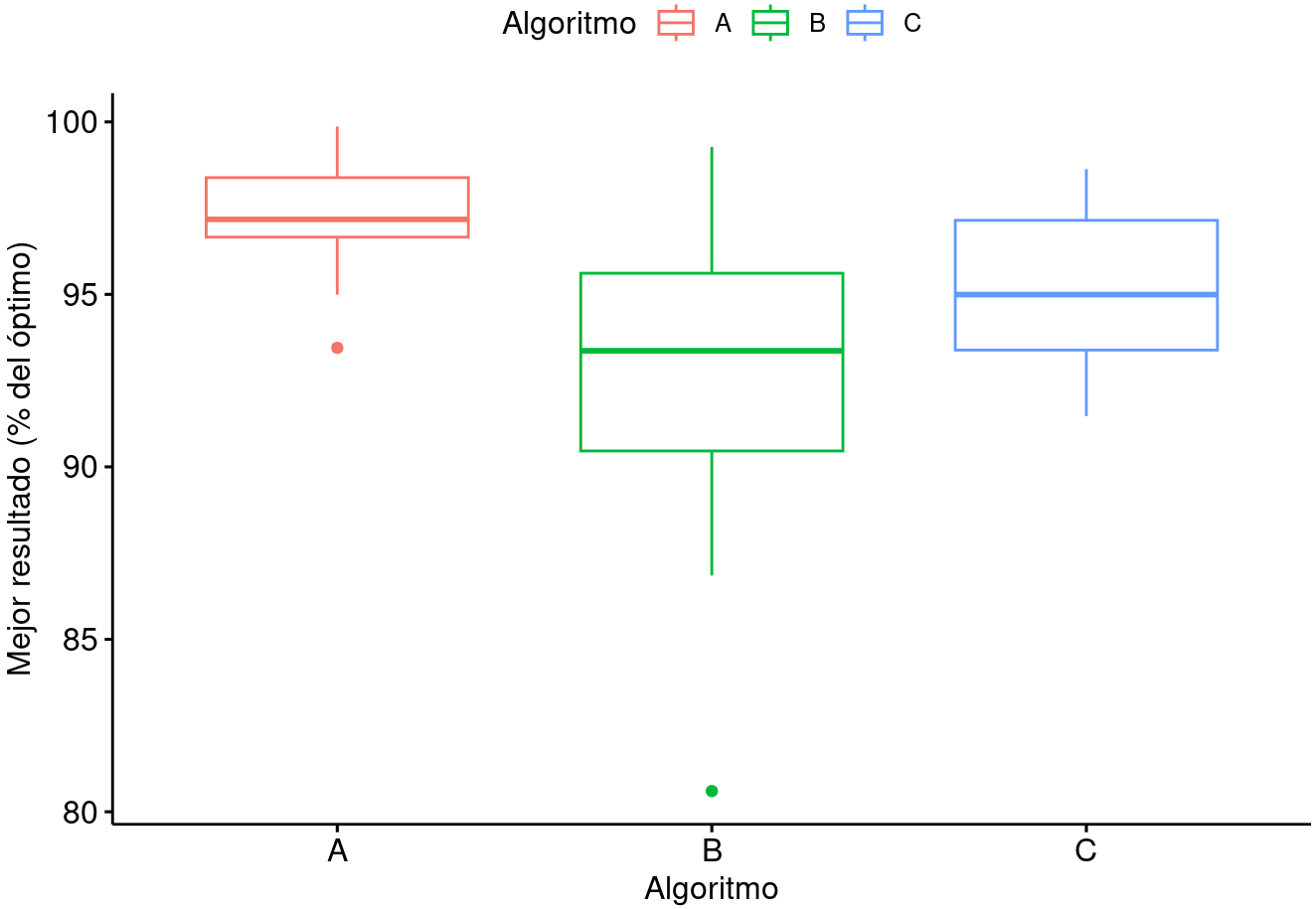
Llevamos los datos a formato largo

```
dl4 <- dw4 %>%
  pivot_longer(
    cols = c("mejor.A", "mejor.B", "mejor.C"),
    names_to = "Algoritmo",
    values_to = "resultado"
  )
dl4[["instancia"]] <- factor(dl4[["instancia"]])
dl4[["Algoritmo"]] <- factor(dl4[["Algoritmo"]], labels = c("A", "B", "C"))
```

Puesto que cada muestra contiene la calidad de la solución obtenida por cada algoritmo al resolver las mismas instancias de prueba, correspondería usar **ANOVA de una vía para medidas repetidas**, que permitiría determinar si la memorista tiene o no razón en pensar que existen diferencias significativas entre los diferentes algoritmos.

Revisemos los datos con un diagrama de cajas.

```
p4 <- ggboxplot(
  dl4,
  x = "Algoritmo", y = "resultado",
  ylab = "Mejor resultado (% del óptimo)",
  color = "Algoritmo"
)
print(p4)
```



Vemos que existen ciertas asimetrías y algunos valores atípicos en los datos. Además, por la forma de las cajas, parece improbable que estos datos cumplan la condición de esfericidad. Luego, preferimos usar una prueba no paramétrica, como nos instruye el enunciado. Al tratarse de medidas repetidas y más de dos grupos, corresponde usar la **prueba de Friedman**.

Verificamos las condiciones:

1. La variable independiente es categórica y tiene a lo menos tres niveles: algoritmos A, B y C.
2. Las tripletas de observaciones son independientes pues, si el experimento fue realizado correctamente, el desempeño de un algoritmo no debería tener influencia en el rendimiento que alcancen los otros dos.
3. La escala de la variable dependiente es a lo menos ordinal, pues valores porcentuales se pueden comparar y ordenar.

En consecuencia, se cumplen las condiciones para aplicar la prueba de Friedman, con las siguientes hipótesis no paramétricas:

H_0: la calidad de las mejores soluciones conseguidas para las mismas instancias de prueba con 100 o más nodos por los tres algoritmos se distribuyen de forma similar.

H_A: al menos uno de los algoritmos entrega mejores soluciones con calidad significativamente distinta que los otros dos al resolver las mismas instancias del problema del vendedor viajero con 100 o más nodos.

Aplicamos la prueba con un nivel de significación exigente por la presencia de valores atípicos ($\alpha = 0{,}01$).

```
prueba4 <- friedman.test(
  resultado ~ Algoritmo | instancia,
  data = dl4
)
print(prueba4)
```

```
    Friedman rank sum test

data:  resultado and Algoritmo and instancia
Friedman chi-squared = 21.689, df = 2, p-value = 1.951e-05
```

La prueba indica que hay suficiente evidencia ($\chi^2 = 21{,}7;\ p < 0{,}001$) para rechazar la hipótesis nula en favor de la alternativa. En consecuencia, podemos concluir con 99% de confianza que al menos uno de los algoritmo tiene un rendimiento distinto a alguno de los otros o a ambos.

Puesto que la prueba ómnibus (de Friedman) detecta diferencias, debemos hacer ahora un procedimiento post-hoc usando múltiples pruebas de rangos con signo de Wilcoxon entre pares de grupos y, al igual que en la pregunta anterior, aplicando el ajuste de Benjamini & Hochberg (1995).

```
posthoc2 <- pairwise.wilcox.test(
  dl4[["resultado"]],
  dl4[["Algoritmo"]],
  p.adjust.method = "BH",
  paired = TRUE
)
```

```
Warning in wilcox.test.default(xi, xj, paired = paired, ...): cannot compute
exact p-value with zeroes
```

```
print(posthoc2)
```

```
    Pairwise comparisons using Wilcoxon signed rank exact test

data:  dl4[["resultado"]] and dl4[["Algoritmo"]]

  A       B
B 2.2e-06 -
C 0.0025  0.0382

P value adjustment method: BH
```

Por ahora vamos a ignorar el warning que nos da la función, puesto que sabemos que si hay empates, el algoritmo interno de esta prueba los descarta antes del cálculo de los rangos (rankings). Si fuéramos muy rigorosos, deberíamos revisar que no son *tantos* empates en cada caso. Si esto es así, los p-valores que se obtienen son buenas aproximaciones de los exactos.

Expresemos la conclusión.

Con 99% confianza, el procedimiento post-hoc no encuentra diferencias significativas entre los algoritmos B y C ($p > 0{,}038$), pero estos dos algoritmos parecen obtener, en promedio, significativamente peores soluciones (más lejanas a la solución óptima) que las que consigue el algoritmo A ($p < 0{,}003$).

Referencias

Benjamini, Y., & Hochberg, Y. (1995). Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal statistical society: series B (Methodological), 57*(1), 289-300.