Tarea 10

Eduardo Navarro

Noviembre 2021

1. Introducción

En esta práctica se evaluaron las instancias a diferentes reglas para el problema de la mochila en donde se observaron resultados interesantes para diferentes casos.

2. Desarrollo

Con las instrucciones de la tarea [5] y lo visto en clase [6] se le hicieron modificaciones al código para obtener las reglas [3] y evaluar a un tiempo determinado [1]. Así mismo se añadió un `for` para hacer las repeticiones y otro `for` para las instancias para cada una de las reglas. Posteriormente se realizó el correspondiente análisis estadístico.

Listing 1: Código para las reglas.

```
#Modificacion para la regla 1
generador.pesos <- function(cuantos, min, max) {
  return(sort(round(runif(cuantos) * (max - min) + min)))
}

generador.valores <- function(cuantos, min, max) {
  return(round(runif(cuantos) * (max - min) + min))
}

#Modificacion para la regla 2
generador.pesos <- function(valores, min, max) {
  n <- length(valores)
  pesos <- double()
  for (i in 1:n) {
    media <- valores[i]
    desv <- runif(1, max=.1)
    ruido <- rnorm(1, sd=.1)
    pesos <- c(pesos, rnorm(1, (1/media), desv) + ruido)
  }
  pesos <- normalizar(pesos) * (max - min) + min
  return(pesos)
}

generador.valores <- function(cuantos, min, max) {
  return(sort(round(normalizar(rexp(cuantos)) * (max - min) + min)))
}

    n <- 50
    pesos <- generador.pesos(valores, 15, 80)
    valores <- generador.valores(n, 10, 500)
```

```
    capacidad <- round(sum(pesos) * 0.65)

#Modificacion para la regla 3
generador.pesos <- function(cuantos, min, max) {
  return(sort(round(normalizar(rnorm(cuantos)) * (max - min) + min)))
}

generador.valores <- function(pesos, min, max) {
  n <- length(pesos)
  valores <- double()
  for (i in 1:n) {
    media <- pesos[i]
    desv <- runif(1)
    ruido <- rnorm(1, sd=.1)
    valores <- c(valores, rnorm(1, media^2, desv) + ruido)
  }
  valores <- normalizar(valores) * (max - min) + min
  return(valores)
}
```

Para cada regla se agregó la probabilidad de mutación, cantidad de cruzamiento y tamaño de población a distintos valores dejando los otros 2 constantes a los cuales se le incluyeron réplicas.

Listing 2: Código para las instancias.

```
#probabilidad de mutacion
pmutant<-c(0.05,0.1,0.5)
j<- 1:3
for (pm in pmutant){
for (replicas in j){

n <- 50
pesos <- generador.pesos(n, 15, 80)
valores <- generador.valores(pesos, 10, 500)
capacidad <- round(sum(pesos) * 0.65)
optimo <- knapsack(capacidad, pesos, valores)
init <- 30
p <- poblacion.inicial(n, init)
tam <- dim(p)[1]
assert(tam == init)

rep <- 10


#cantidad de cruzamiento
reproducc<-c(10, 20, 30)
j<- 1:3
for (rep in reproducc){
for (replicas in j){

n <- 50
pesos <- generador.pesos(n, 15, 80)
valores <- generador.valores(pesos, 10, 500)
capacidad <- round(sum(pesos) * 0.65)
optimo <- knapsack(capacidad, pesos, valores)
init <- 30
p <- poblacion.inicial(n, init)
```

```
tam <- dim(p)[1]
assert(tam == init)
pm <- 0.05

#tamano de poblacion
poblacion<-c(20, 30, 40)
j<- 1:3
for (init in poblacion){
for (replicas in j){

n <- 50
pesos <- generador.pesos(n, 15, 80)
valores <- generador.valores(pesos, 10, 500)
capacidad <- round(sum(pesos) * 0.65)
optimo <- knapsack(capacidad, pesos, valores)

p <- poblacion.inicial(n, init)
tam <- dim(p)[1]
assert(tam == init)
pm <- 0.05
rep <- 10
```

Para su evaluación se cambiaron las iteraciones por un límite de tiempo que se agregó con un `while` a 5 segundos.

Listing 3: Código para el límite de tiempo.

```
mejores <- double()

timer = 5
start = Sys.time()

while(TRUE) {
  elapsed = as.numeric(difftime(Sys.time(), start, units = 'secs'))
  remaining = timer - round(elapsed)
  Sys.sleep(.1)

  print(remaining)

  p$obj <- NULL
  p$fact <- NULL
  ....
  print(paste(mejor, (optimo - mejor) / optimo))


  faltaaloptimo<-(optimo - mejor) / optimo
  segundos<-round(elapsed)

  if (remaining <= 0) break
```

Se capturó todo en un `data.frame` respectivo para cada regla con cada instancia.

Listing 4: Ejemplo de código para el `data.frame`.

```
datos = data.frame()

pmutant<-c(0.05,0.1,0.5)
j<- 1:3
....
  faltaaloptimo<-(optimo - mejor) / optimo
```

```
  segundos<-round(elapsed)

  if (remaining <= 0) break


  resultado = c(replicas, segundos, mejor,faltaaloptimo,pm,optimo)
  datos = rbind(datos, resultado)
  names(datos) = c("replicas", "segundo", "mejor", "%optimo", "pm","Optimo")
}
```

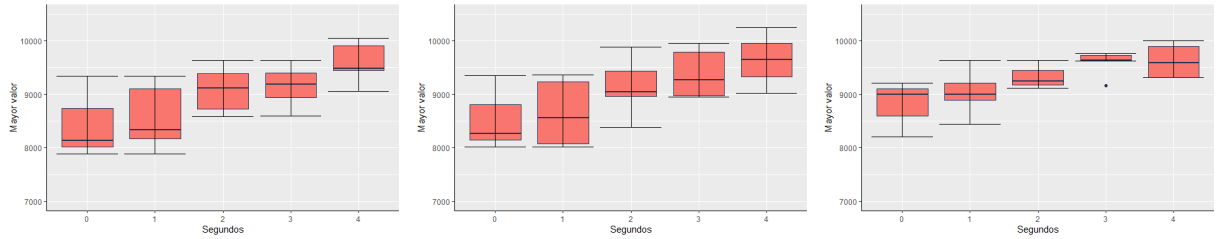
Después de la aplicación de una regla los siguientes valores se analizaron manteniendo el mismo entorno al correr el programa a partir del último `data.frame`. A partir del programa se obtuvo una tabla 1.

Tabla 1: Ejemplo de datos obtenidos.

replicas	**segundo**	**mejor**	**%optimo**	**pm**	**Optimo**
1	0	9341	0.194811	0.05	11601
1	1	9341	0.194811	0.05	11601
1	1	9341	0.194811	0.05	11601
1	2	9475	0.183260	0.05	11601

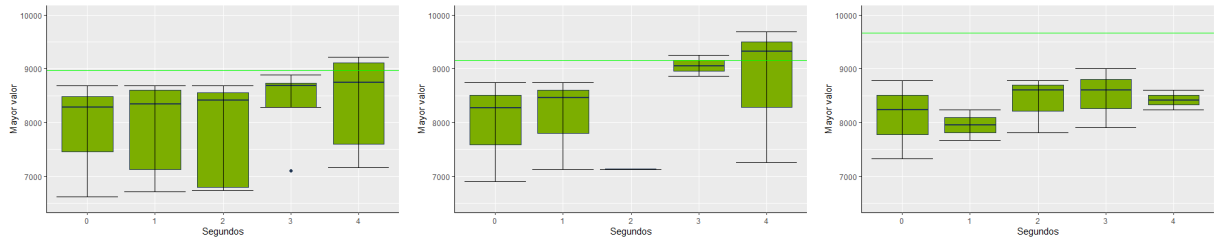
Con los datos de la tabla que se generan para cada regla en cada una de las instancias se procedió a graficar primero para la regla 1.

Figura 1: Variación de probabilidad de mutación para la regla 1.



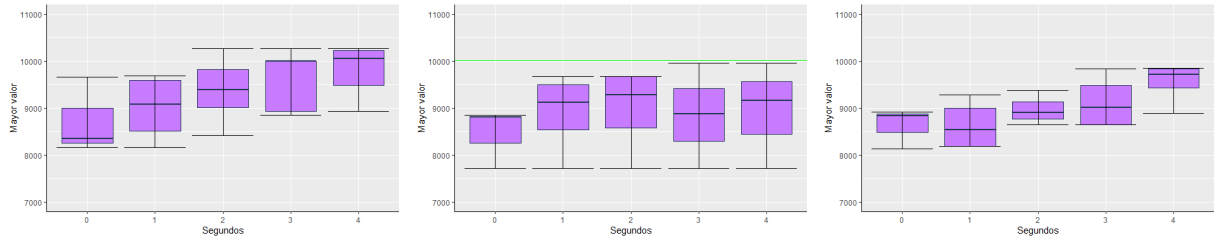
(a) 0.05 (b) 0.1 (c) 0.5

Figura 2: Variación de cruzamientos para la regla 1.



(a) 10 (b) 20 (c) 30

Figura 3: Variación de población para regla 1.



(a) 20 (b) 30 (c) 40

Con estas gráficas se procedió a hacer el análisis estadístico con Shapiro–Wilk [7] y Wilcox [4].

Listing 5: Código para gráficas y análisis estadístico.

```
#Mutaciones
library(ggplot2)
datos$segundo = as.factor(datos$segundo)
datoss = split.data.frame(datos, f = datos$pm)

ggplot(datoss$`0.05`, aes(x= segundo, y= mejor)) +
  geom_boxplot(fill = "#F8766D", colour = "#1F3552")+
  stat_boxplot(geom = "errorbar", width = 0.9)+
  theme(axis.line = element_line(colour = "black", size = 0.25))+
  geom_hline(aes(yintercept=Optimo), colour="green")+
  labs(x = "Segundos", y = "Mayor valor")+
  coord_cartesian(ylim = c(7000, 10500))

ggplot(datoss$`0.1`, aes(x= segundo, y= mejor)) +
  geom_boxplot(fill = "#F8766D", colour = "#1F3552")+
  stat_boxplot(geom = "errorbar", width = 0.9)+
  geom_hline(aes(yintercept=Optimo), colour="green")+
  theme(axis.line = element_line(colour = "black", size = 0.25))+
  labs(x = "Segundos", y = "Mayor valor")+
  coord_cartesian(ylim = c(7000, 10500))

ggplot(datoss$`0.5`, aes(x= segundo, y= mejor)) +
  geom_boxplot(fill = "#F8766D", colour = "#1F3552")+
  stat_boxplot(geom = "errorbar", width = 0.9)+
  geom_hline(aes(yintercept=Optimo), colour="green")+
  theme(axis.line = element_line(colour = "black", size = 0.25))+
  labs(x = "Segundos", y = "Mayor valor")+
  coord_cartesian(ylim = c(7000, 10500))

cshapiro<-tapply(datos$mejor, datos$segundo, shapiro.test)

cwilcox<-pairwise.wilcox.test(datos$mejor, datos$segundo)

#cruzamientos
library(ggplot2)
datos$segundo = as.factor(datos$segundo)
datoss = split.data.frame(datos, f = datos$cruzamientos)

ggplot(datoss$`10`, aes(x= segundo, y= mejor)) +
  geom_boxplot(fill = "#7CAE00", colour = "#1F3552")+
  stat_boxplot(geom = "errorbar", width = 0.9)+
```

```
  theme(axis.line = element_line(colour = "black", size = 0.25))+
  geom_hline(aes(yintercept=Optimo), colour="green")+
  labs(x = "Segundos", y = "Mayor valor")+
  coord_cartesian(ylim = c(6500, 10000))

ggplot(datoss$`20`, aes(x= segundo, y= mejor)) +
  geom_boxplot(fill = "#7CAE00", colour = "#1F3552")+
  stat_boxplot(geom = "errorbar", width = 0.9)+
  geom_hline(aes(yintercept=Optimo), colour="green")+
  theme(axis.line = element_line(colour = "black", size = 0.25))+
  labs(x = "Segundos", y = "Mayor valor")+
  coord_cartesian(ylim = c(6500, 10000))

ggplot(datoss$`30`, aes(x= segundo, y= mejor)) +
  geom_boxplot(fill = "#7CAE00", colour = "#1F3552")+
  stat_boxplot(geom = "errorbar", width = 0.9)+
  geom_hline(aes(yintercept=Optimo), colour="green")+
  theme(axis.line = element_line(colour = "black", size = 0.25))+
  labs(x = "Segundos", y = "Mayor valor")+
  coord_cartesian(ylim = c(6500, 10000))

cshapiro<-tapply(datos$mejor, datos$segundo, shapiro.test)

cwilcox<-pairwise.wilcox.test(datos$mejor, datos$segundo)

#poblacion

library(ggplot2)
datos$segundo = as.factor(datos$segundo)
datoss = split.data.frame(datos, f = datos$poblacion)

ggplot(datoss$`20`, aes(x= segundo, y= mejor)) +
  geom_boxplot(fill = "#C77CFF", colour = "#1F3552")+
  stat_boxplot(geom = "errorbar", width = 0.9)+
  theme(axis.line = element_line(colour = "black", size = 0.25))+
  geom_hline(aes(yintercept=Optimo), colour="green")+
  labs(x = "Segundos", y = "Mayor valor")+
  coord_cartesian(ylim = c(7000, 11000))

ggplot(datoss$`30`, aes(x= segundo, y= mejor)) +
  geom_boxplot(fill = "#C77CFF", colour = "#1F3552")+
  stat_boxplot(geom = "errorbar", width = 0.9)+
  geom_hline(aes(yintercept=Optimo), colour="green")+
  theme(axis.line = element_line(colour = "black", size = 0.25))+
  labs(x = "Segundos", y = "Mayor valor")+
  coord_cartesian(ylim = c(7000, 11000))

ggplot(datoss$`40`, aes(x= segundo, y= mejor)) +
  geom_boxplot(fill = "#C77CFF", colour = "#1F3552")+
  stat_boxplot(geom = "errorbar", width = 0.9)+
  geom_hline(aes(yintercept=Optimo), colour="green")+
  theme(axis.line = element_line(colour = "black", size = 0.25))+
  labs(x = "Segundos", y = "Mayor valor")+
  coord_cartesian(ylim = c(7000, 11000))
```

```
cshapiro<-tapply(datos$mejor, datos$segundo, shapiro.test)

cwilcox<-pairwise.wilcox.test(datos$mejor, datos$segundo)
```

Tabla 2: Resultados de la prueba Shapiro–Wilk para la regla 1 en probabilidad de mutaciones.

seg	W	P
0	0.8388	0.0561
1	0.9136	0.0997
2	0.9590	0.6753
3	0.9397	0.3796
4	0.9425	0.4901

Tabla 3: Resultados de la prueba Shapiro–Wilk para la regla 1 en cruzamiento.

seg	W	P
0	0.8510	0.0765
1	0.8317	0.0246
2	0.8289	0.0434
3	0.8300	0.0446
4	0.8851	0.1207

Tabla 4: Resultados de la prueba Shapiro–Wilk para la regla 1 en población.

seg	W	P
0	0.9478	0.6668
1	0.9170	0.1992
2	0.9616	0.7495
3	0.9169	0.2617
4	0.8706	0.0534

Tabla 5: Resultados de la prueba por parejas de Wilcox para la regla 1 en probabilidad de mutaciones.

seg	0	1	2	3
1	0.4864	-	-	-
2	0.2127	0.2127	-	-
3	0.0608	0.0214	0.4740	-
4	0.0120	0.0025	0.0992	0.4740

Tabla 6: Resultados de la prueba por parejas de Wilcox para la regla 1 en cruzamientos.

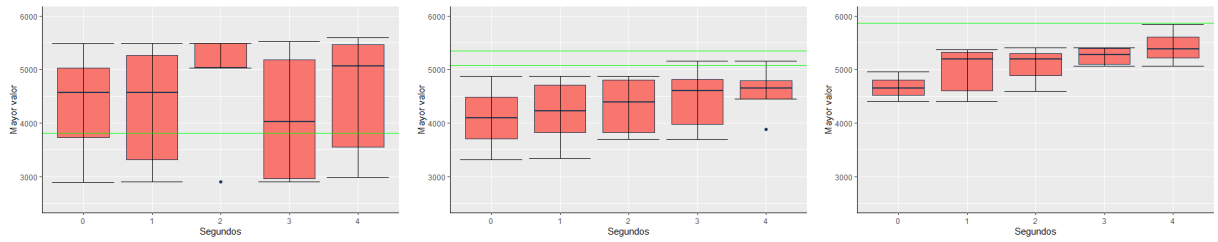
seg	0	1	2	3
1	1.00	-	-	-
2	1.00	1.00	-	-
3	0.74	0.46	0.46	-
4	1.00	0.74	0.74	1.00

Tabla 7: Resultados de la prueba por parejas de Wilcox para la regla 1 en población.

seg	0	1	2	3
1	0.926	-	-	-
2	0.313	0.926	-	-
3	0.313	0.823	0.926	-
4	0.037	0.137	0.793	0.926

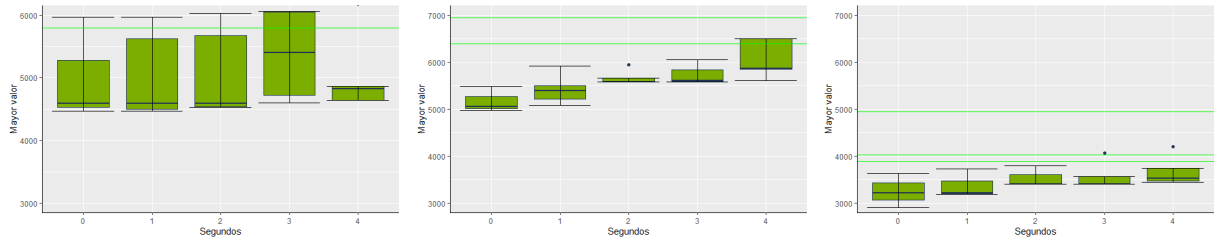
Del mismo modo se hizo para la regla 2.

Figura 4: Variación de probabilidad de mutación para la regla 2.



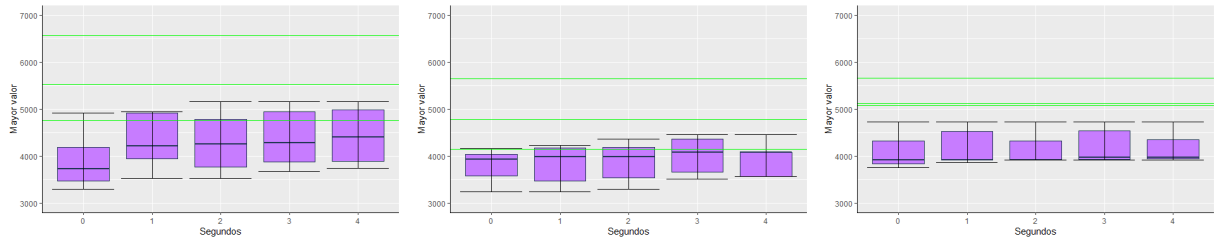
(a) 0.05 (b) 0.1 (c) 0.5

Figura 5: Variación de cruzamientos para la regla 2.



(a) 10 (b) 20 (c) 30

Figura 6: Variación de población para regla 2.



(a) 20 (b) 30 (c) 40

Tabla 8: Resultados de la prueba Shapiro–Wilk para la regla 2 en probabilidad de mutaciones.

seg	W	P
0	0.9393	0.5754
1	0.9062	0.0738
2	0.9003	0.1141
3	0.8231	0.0056
4	0.8596	0.0380

Tabla 9: Resultados de la prueba Shapiro–Wilk para la regla 2 en cruzamiento.

seg	W	P
0	0.9565	0.7612
1	0.9188	0.2116
2	0.8860	0.0861
3	0.8531	0.0469
4	0.9203	0.2222

Tabla 10: Resultados de la prueba Shapiro–Wilk para la regla 2 en población.

seg	W	P
0	0.9271	0.4201
1	0.9299	0.2169
2	0.9603	0.6983
3	0.9379	0.2677
4	0.9176	0.2033

Tabla 11: Resultados de la prueba por parejas de Wilcox para la regla 2 en probabilidad de mutaciones.

seg	0	1	2	3
1	1	-	-	-
2	1	1	-	-
3	1	1	1	-
4	1	1	1	1

Tabla 12: Resultados de la prueba por parejas de Wilcox para la regla 2 en cruzamientos.

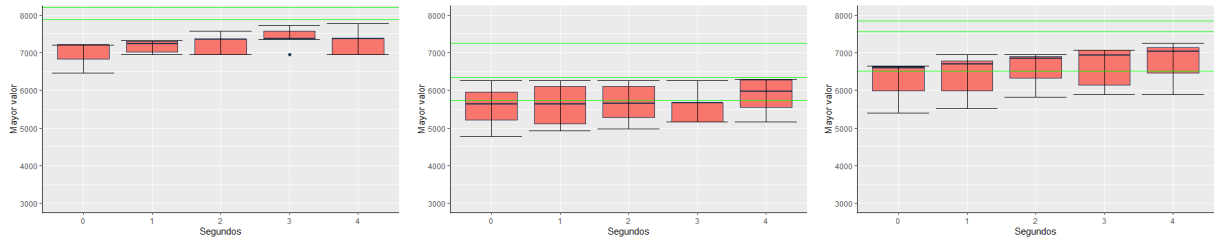
seg	0	1	2	3
1	1	-	-	-
2	1	1	-	-
3	1	1	1	-
4	1	1	1	1

Tabla 13: Resultados de la prueba por parejas de Wilcox para la regla 2 en población.

seg	0	1	2	3
1	1	-	-	-
2	1	1	-	-
3	1	1	1	-
4	1	1	1	1

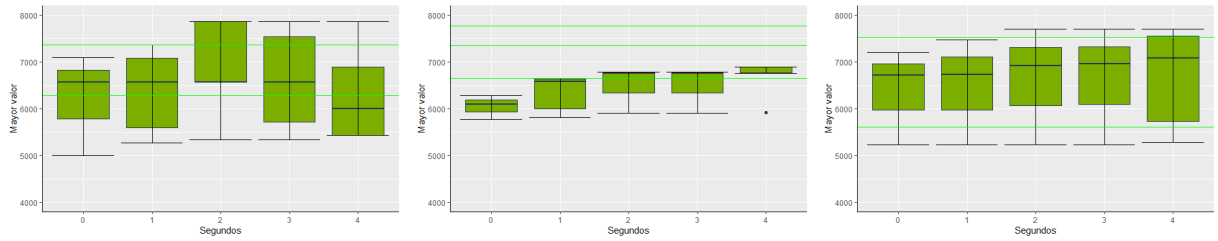
Y de igual forma para la regla 3.

Figura 7: Variación de probabilidad de mutación para la regla 3.



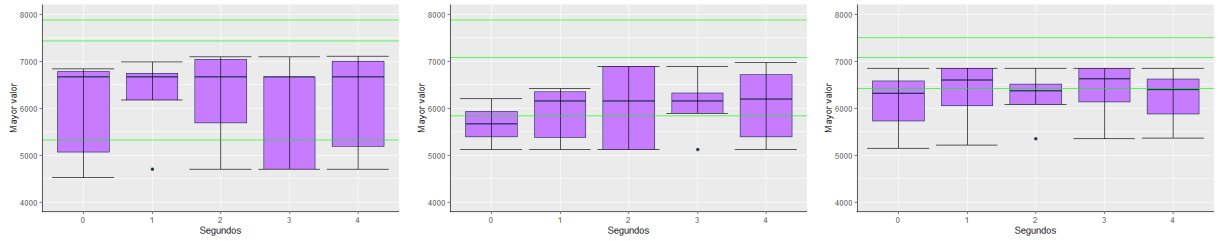
(a) 0.05 (b) 0.1 (c) 0.5

Figura 8: Variación de cruzamientos para la regla 3.



(a) 10 (b) 20 (c) 30

Figura 9: Variación de población para regla 3.



(a) 20 (b) 30 (c) 40

Tabla 14: Resultados de la prueba Shapiro–Wilk para la regla 3 en probabilidad de mutaciones.

seg	W	P
0	0.9335	0.5156
1	0.8953	0.0476
2	0.9286	0.2916
3	0.9095	0.1143
4	0.9409	0.5107

Tabla 15: Resultados de la prueba Shapiro–Wilk para la regla 3 en cruzamiento.

seg	W	P
0	0.9504	0.6947
1	0.8964	0.0838
2	0.9165	0.2907
3	0.9128	0.2321
4	0.8991	0.0922

Tabla 16: Resultados de la prueba Shapiro–Wilk para la regla 3 en población.

seg	W	P
0	0.8532	0.0402
1	0.8689	0.0405
2	0.8432	0.0108
3	0.8627	0.0418
4	0.8647	0.0282

Tabla 17: Resultados de la prueba por parejas de Wilcox para la regla 3 en probabilidad de mutaciones.

seg	0	1	2	3
1	1	-	-	-
2	1	1	-	-
3	1	1	1	-
4	1	1	1	1

Tabla 18: Resultados de la prueba por parejas de Wilcox para la regla 3 en cruzamientos.

seg	0	1	2	3
1	1	-	-	-
2	1	1	-	-
3	1	1	1	-
4	1	1	1	1

Tabla 19: Resultados de la prueba por parejas de Wilcox para la regla 3 en población.

seg	0	1	2	3
1	1	-	-	-
2	1	1	-	-
3	1	1	1	-
4	1	1	1	1

Todos los códigos para la generación del presente trabajo se encuentran en mi github [2].

3. Conclusiones

De la regla 1 podemos concluir que tiende a llegar al óptimo conforme más tiempo pase para las 3 instancias, la que menos valor mayor alcanzó fué con los cruzamientos junto con valores óptimos bajos, que también fué la que no presentó normalidad. De la prueba de Wilcox para la regla 1 se tienen diferencias entre el primer y último segundo para la probabilidad de mutaciones y fué el que presentó mayores diferencias entre grupos. La que mantuvo valores mayores fué la variación de población.

De la regla 2 podemos concluir que igual tiende a llegar al óptimo conforme más tiempo pase. La que menos valor mayor alcanzó en general fué con la población. Siendo la de cruzamientos a 30 la menor de todas y la de 20 la mayor de todas. En su mayoría presentaron normalidad y no se observaron diferencias entre grupos.

De la regla 3 podemos concluir que igual se observa una tendencia para la probabilidad de mutación a alcanzar un mayor valor. Siendo que los otros se quedan relativamente igual entre ellos. El mayor valor lo alcanzó la variación de cruzamientos. La variación de población fué la que no presentó normalidad y obtuvo valores bajos. En los 3 no se apreciaron diferencias entre grupos.

En general el mayor valor alcanzado regla 1 a 20 de población, menor valor alcanzado regla 2 a 30 de cruzamientos, más óptimos cercanos para la variación de cruzamiento.

Referencias

[1] dww. Is there a way to incorporate a countdown timer into an R function? https://stackoverflow.com/questions/56857150/is-there-a-way-to-incorporate-a-countdown-timer-into-an-r-function/, 2021. [Online; accessed 2-November-2021].

[2] eduartilon. tarea10, 2021. URL https://github.com/eduartilon/Simulacion/tree/main/tarea10.

[3] María Montemayor. Algoritmo genetico, 2021. URL https://github.com/MariaMontemayor/Simul/tree/main/tarea10.

[4] Thomas Pernet. 9 Non Parametric tests, 2020. URL https://bookdown.org/thomas_pernet/Tuto/non-parametric-tests.html.

[5] Elisa Schaeffer. Práctica 10: algoritmo genético. https://elisa.dyndns-web.com/teaching/comp/par/p10.html/, 2021. [Online; accessed 2-November-2021].

[6] Elisa Schaeffer. Simulación: algoritmo genético (p10, ad21), 2021. URL https://www.twitch.tv/videos/1188685632.

[7] El Tío Estadístico. Cómo hacer la Prueba de Normalidad en R, 2020. URL https://www.youtube.com/watch?v=LAzSb6jCFbs.