Wojskowa Akademia Techniczna

Analiza i wizualizacja danych

Dostosowywanie wykresów

Prowadzący: xxxx
Wykonał: Konrad Prusaczyk
Grupa: WCY18IJ3S1
Data wykonania: 29/04/2021

Zadanie 1

Omówić różnice pomiędzy programami a i b i c

 # Program a

```
par(mai = c(1, 1, 1, 1), omi = c(0, 0, 0, 0))
xx <- c(9.20, 6.00, 6.00, 11.25, 11.00, 7.25, 9.7, 13.25, 14.00, 8.00)
hist(xx, breaks = c(6, 8, 10, 12, 14))
```

 # Program b

```
par(mai = c(1, 1, 1, 1), omi = c(0, 0, 0, 0))
xx <- c(9.20, 6.00, 6.00, 11.25, 11.00, 7.25, 9.7, 13.25, 14.00, 8.00)
hist(xx, breaks = c(6, 8, 10, 12, 14), right = F)
```

 # Program c

```
par(mai = c(1, 1, 1, 1), omi = c(0, 0, 0, 0))
xx <- c(9.20, 6.00, 6.00, 11.25, 11.00, 7.25, 9.7, 13.25, 14.00, 8.00)
 br1 <-  c(6, 8, 10, 12, 14, 16)
 bw1 <- br1[2] - br1[1]
 xxh <- floor(xx/bw1) * bw1 + 0.1 * bw1
hist(xxh, breaks = br1, right = F)
```

Różnica pomiędzy programem a I b:
Brak przekazania argumentu „right" defaultowo ustawiana jest na wartość „True". Stąd wynika różnica, ponieważ w programie b komórki histogramu nie są prawostronnie zamknięte (lewostronnie otwarte).

Różnica pomiędzy programem b i c:
W programie c jako argument „breaks" zostaje przekazany inny wektor niż w przypadku programu b. Dzięki czemu w programie c występuje więcej „breakpoint'ów" czyli kolumn przez co zostaje przedłużona oś X (xxh) i wykres dzięki temu jest czytelniejszy.

Różnice pomiędzy programem a i c wynikają z powyższych opisów różnic pomiędzy programem a i b oraz b i c.

Zadanie 2

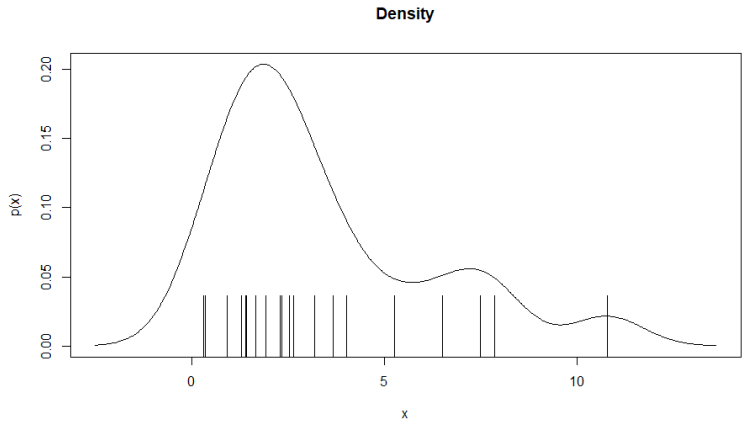
Co przedstawia wykres generowany poniższym kodem ? Dodać tytuł do wykresu.

```
par(mai = c(1, 1, 1, 1), omi = c(0, 0, 0, 0))
xx <- c(1.92, 4.01, 6.51, 1.40, 1.67, 5.27, 1.42, 0.36,
  3.18, 3.67, 7.48, 2.65, 7.86, 10.78, 2.30, 1.29, 0.31, 0.93,
  2.34, 2.53)

d1 <- density(xx, bw = "Sj-ste")
ex <- d1$x
ey <- d1$y

plot(ex, ey, type = "n", xlab = "x", ylab = "p(x)", main = "Density") lines(ex, ey)
```

```
rug(xx, ticksize = 0.2, lwd = 1)
```



Powyższy wykres przedstawia nieparametryczny estymator jądrowy gęstości.
Przy pomocy funkcji density() obliczany jest estymator jądrowy gęstości przy jednoczesnym „wygładzeniu" pasma (bandwidth) metodą „SJ-ste".
Przy pomocy funkcji plot() rysujemy osie wraz z ich labelami, pole na wykres, tytuł wykresu ale samego wykresu nie ze względu na przekazanie argumentu type="n". Dopiero przy pomocy funkcji lines() ukazuje nam się wykres, który możemy już w jakiś sposób analizować. Dopełnieniem jest funkcja rug(), która rysuje wykres danych dla pojedynczej zmiennej ilościowej w tym przypadku dla zmiennej „xx", wyświetlany jako znaczniki wzdłuż osi. Służy do wizualizacji dystrybucji danych.

Zadanie 3

Co prezentuje wykres generowany poniższym kodem ? Omówić znaczenie strzałek.
Dodać tytuł do wykresu.

```
# (1)
 par(mai = c(1, 1, 1, 1), omi = c(0, 0, 0, 0))
# (2)
 set.seed(591)
 xx1 <- rnorm(20, mean = 3, sd = 3.6)
 xx2 <- rpois(40, lambda = 3.5)
 xx3 <- rchisq(31, df = 5)
# (3)
 mean1 <- c(mean(xx1), mean(xx2), mean(xx3))
 sd1 <- c(sd(xx1), sd(xx2), sd(xx3))
 data1 <- list(xx1, xx2, xx3)
# (4)
 xmin1 <- min(xx1, xx2, xx3) - 1
 xmax1 <- max(xx1, xx2, xx3) + 1
# (5)
 stripchart(data1, method = "jitter", jit = 0.3, vert = T,
  pch = 1, cex = 0.4, ylim = c(xmin1, xmax1),
  group.names = c("Group-1", "Group-2", "Group-3"), main="Rozkłady normalny, poissona, chi-kwadrat")
# (6)
```

```
arrows(1:3, mean1 + sd1, 1:3, mean1 - sd1, angle = 45,
 code = 3, length = 0.07)
arrows(1:3, mean1 + 2 * sd1, 1:3, mean1 - 2 * sd1,
 angle = 30, code = 3, length = 0.07)
arrows(1:3, mean1 + 0.01 * sd1, 1:3, mean1 - 0.01 * sd1,
 angle = 90, code = 3, length = 0.12)
```

#1

Ustawienie parametrów graficznych dla rozmiaru marginesu oraz rozmiaru marginesów zewnętrznych.

#2

Z 591 wygenerowanych zmiennych losowych

- **Generujemy 20 wielowymiarowych normalnych zmiennych losowych. Przy średniej=3 oraz odchyleniu standardowym 3.6.**
- **Generujemy 40 losowych wartości dla rozkładu Poissona z parametrem lambda 3.5.**
- **Generujemy 31 losowych wartości dla rozkładu chi-kwadrat ze stopniem swobody 5.**

#3

Obliczanie średniej ze zmiennych xx1, xx2, xx3 i przypisanie ich do zmiennej mean1.
Wyznaczenie odchylenia standardowego z trzech powyższych zmiennych i przypisanie ich do zmiennej sd1.
Konstrukcja listy z trzech powyższych zmiennych.

#4

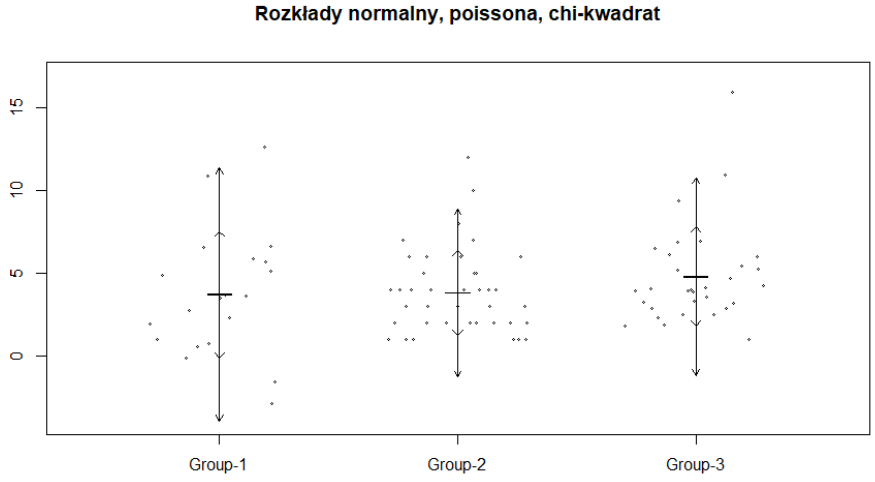
Przypisanie wartości minimalnej pomniejszonej o 1 z trzech w.w zmiennych do zmiennej xmin1. Podobnie w przypadku zmiennej xmax1.

#5

Utworzenie pionowego wykresu paskowego dla listy data1 przy zastosowaniu metody „jitter" w celu uwidocznienia nakładających się na siebie punktów.

#6

Dla każdej grupy rysowane są strzałki.
Pierwsza strzałka pod kątem 45° o koordynatach mean1 + sd1 i mean1 - sd1, długości 0.07 i stylu 3.
Druga strzałka pod kątem 30° o koordynatach mean1 + 2 * sd1 i mean1 - 2 * sd1, długości 0.07 i stylu 3.
Trzecia strzałka pod kątem 90° o koordynatach mean1 + 0.01 * sd1 i mean1 - 0.01 * sd1, długości 0.07 i stylu



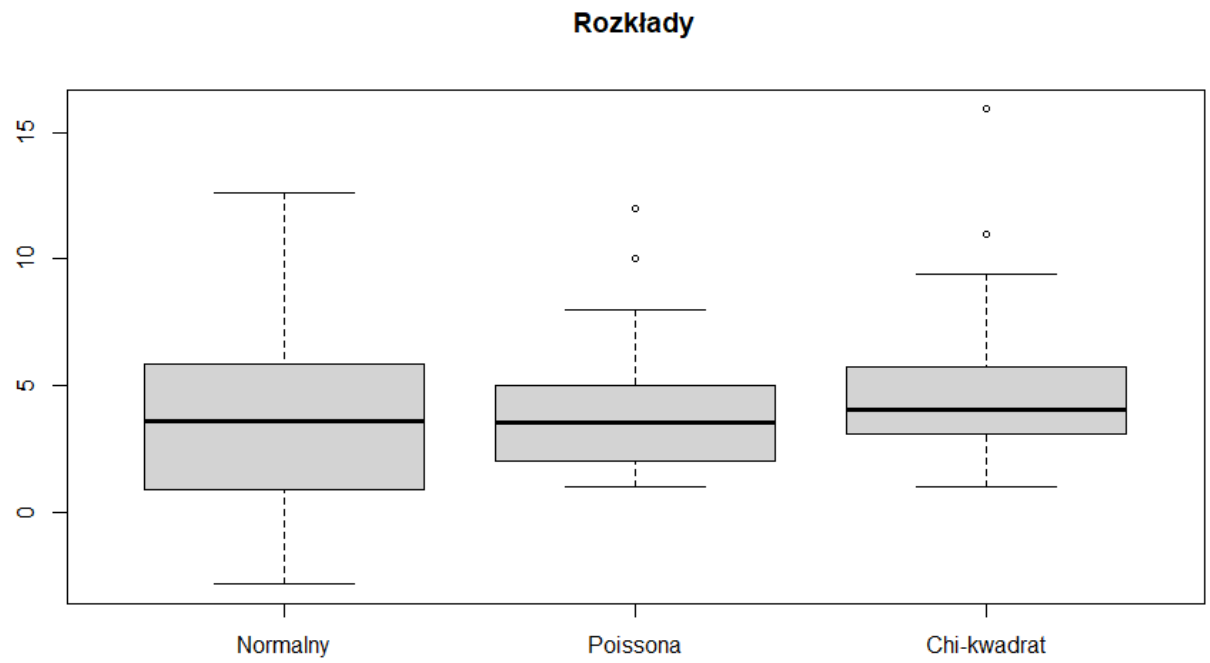
Zadanie 4

Co prezentuje wykres generowany poniższym kodem ?
Dodać tytuł i oznaczenia do wykresu tak aby był on bardziej czytelny.

```
# (1)
 par(mai = c(1, 1, 1, 1), omi = c(0, 0, 0, 0))
# (2)
 set.seed(591)
 xx1 <- rnorm(20, mean = 3, sd = 3.6)
 xx2 <- rpois(40, lambda = 3.5)
 xx3 <- rchisq(31, df = 5, ncp = 0)
# (3)
 box1 <- boxplot(xx1, xx2, xx3, names = c("Normalny", "Poissona",
  "Chi-kwadrat"), cex = 0.7, main="Rozkłady")

# (4)
 print("box1$stats")
 print(box1$stats)
```

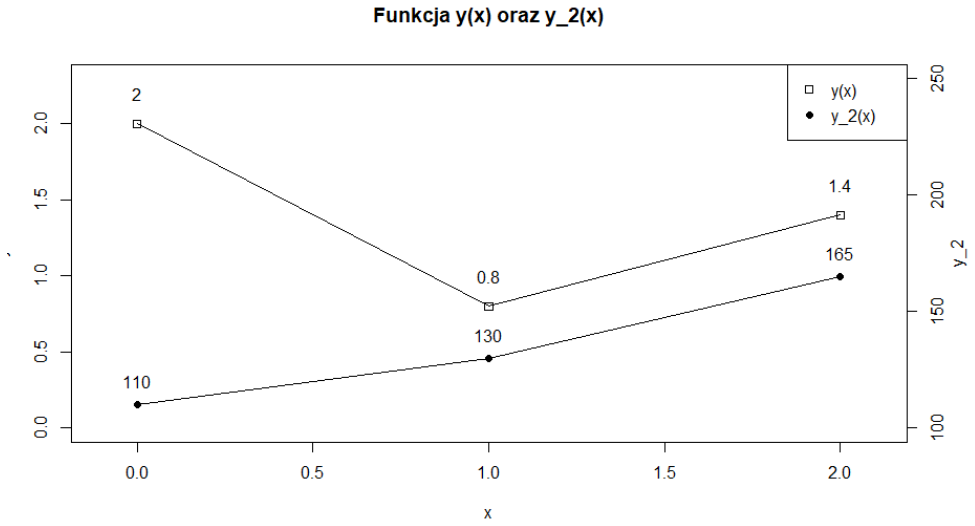
Działania na zmiennych są podobne jak w poprzednim zadaniu, tym razem przedstawiamy wartości zmiennych przy pomocy wykresu pudełkowego. W odróżnieniu od wykresu w poprzednim zadaniu, przy pomocy wykresu pudełkowego otrzymujemy również informacje o medianie, miarach tendencji centralnej czyli pierwszym oraz trzecim kwartylu i rozstępie ćwiartkowym.



Zadanie 5

Co prezentuje wykres generowany poniższym kodem ?
Dodać tytuł i legendę do wykresu tak aby był on bardziej czytelny.

```
# (1)
 par(mai = c(1, 1, 1, 1), omi = c(0, 0, 0, 0))
# (2)
 plot(c(-0.1, 2.1), c(0, 2.3), type = "n", xlab = "x",
  ylab = "y", main="Funkcja  y(x) oraz y_2(x)")
 xx <- c(0, 1,2)
 yy <- c(2, 0.8, 1.4)
 lines(xx, yy)
 points(xx, yy, pch = 0)
# (3)
 text(xx, yy + 0.2, labels = as.character(yy))
# (4)
 par(new = T)
# (5)
 plot(c(-0.1, 2.1), c(100, 250), type = "n", xlab = "",
  ylab = "", axes = F)
# (6)
 axis(4)
 mtext("y_2",  side = 4, line = 2)
# (7)
 xx <- c(0, 1, 2)
 yy <- c(110, 130, 165)
 lines(xx, yy)
 points(xx, yy, pch = 16)
# (8)
 text(xx, yy + 10, labels = as.character(yy))
 legend("topright", legend = c("y(x)", "y_2(x)"), pch=c(0,16))
```



#1
Ustawienie parametrów graficznych dla rozmiaru marginesu oraz rozmiaru marginesów zewnętrznych.

#2
Tak jak w zadaniu 2, linie funkcji rysujemy dopiero przy pomocy funkcji lines() oraz jej punkty funkcją points(). Przez to, że do funkcji plot() przekazujemy argument type="n" wykres nie jest w pełni uzupełniony.

#3
Dodajemy etykiety punktów funkcji y(x).

#4
Deklarujemy, że kolejny wykres ma być nałożony na poprzedni.

#5
Tak jak wyżej, z powodu przekazaniu argumentu „type="n" po wywołaniu funkcji plot() nie widać żadnych zmian gołym okiem.

#6
Dodajemy drugą oś y_2 z prawej strony.

#7
Rysujemy wykres funkcji y_2 wraz z jej punktami.

#8
Dodajemy etykiety punktów funkcji y_2(x).

Zadanie 6

Przeanalizować poniższy program. Opisać znaczenie poszczególnych sekcji.

```
# (1)
 par(mai = c(1, 1, 1, 1), omi = c(0, 0, 0, 0))
# (2)
 yy <- c(50,30,40)
# (3)
 name1 <- c("data-a", "data-b", "data-c")
# (4)
 pie(yy, labels = name1, col = c("red","green","skyblue"))
# (5)
 par(new = T)
# (6)
 par(mai=c(2, 2, 2, 2))
# (7)
 yy2 <- c(50, 20, 10, 20, 20)
# (8)
 name2 <- c("data-a1", "data-b1", "data-b2", "data-c1",
  "data-c2")
# (9)
```

```
pie(yy2, labels = name2, col = c("pink", "gold", "blue",
 "gold", "blue"))
```

#1
Ustawienie parametrów graficznych dla rozmiaru marginesu oraz rozmiaru marginesów zewnętrznych.

#2
Przypisanie do zmiennej yy wektora z trzema wartościami typu integer 50, 30, 40.

#3
Przypisanie do zmiennej name1 wektora z trzema zmiennymi typu string „data-a", „data-b", „data-c".

#4
Tworzymy diagram kołowy dla zmiennych yy, podpisując je etykietami ze zmiennej name1. Dodatkowo ustawiamy kolory kolejnych zmiennych na czerwony, zielony i podniebny.

#5
Deklarujemy, że kolejny wykres ma być nałożony na poprzedni.

#6
Zmieniamy parametr graficzny dla rozmiaru marginesu.

#7
Przypisanie do zmiennej yy2 wektora z pięcioma wartościami typu integer 50, 20, 10, 20, 20.

#8
Przypisanie do zmiennej name2 wartości z pięcioma zmiennymi typu string "data-a1", "data-b1", "data-b2", "data-c1", "data-c2".

#9
Nakładamy na poprzedni diagram nowy diagram kołowy dla zmiennych yy2, podpisując je etykietami ze zmiennej name2. Dodatkowo ustawiamy kolory kolejnych zmiennych różowy, złoty, niebieski, złoty, niebieski.