Piotr Zieleń

Sprawozdanie 1

20 lipca 2022

Spis treści

Celem sprawozdania jest przedstawienie rozwiązań oraz wniosków, gdy jest to konieczne, z rozwiązywanych podczas zajęć labolatoryjnych list zadań.

1. Lista 1

Zaimportowałem biblioteki konieczne do rozwiązania zadań.

```
library(vcdExtra) # w tej bibliotece
          # znajdują się dane, z których korzystamy
library(tidyverse)
```

1.1. Zadanie 1

W tym zadaniu należało sporządzić tabele liczności dla zmiennych *Temperature* oraz *Preference* biorąc pod uwagę wszystkie dane, jak również w podgrupach ze względu na zmienną *Water softness*.

Kody do uzyskania wyników przedstawiłem dla zmiennej *Temperature* dla wszystkich danych i dla podgrupy *Soft*. W pozostałych przypadkach kody wyglądały bardzo podobnie, zmieniłem tylko zmienną, lub nazwę podgrupy.

Uzyskałem następujące wyniki:

— Dla zmiennej *Temperature*:

- Dla wszystkich podgrup:

```
apply(Detergent, "Temperature", sum)
## High  Low
##  369  639
```

- Dla podgrupy *Soft*:

```
apply(Detergent[, , , "Soft"], "Temperature", sum)
## High  Low
##  104  222
```

- Dla podgrupy *Medium*:

```
## High  Low
##  126  218
```

- Dla podgrupy *Hard*:

```
## High  Low
##  139  199
```

— Dla zmiennej *Preference*:

- Dla wszystkich podgrup:

```
## Brand X Brand M
##     508     500
```

- Dla podgrupy *Soft*:

```
## Brand X Brand M
##     168     158
```

- Dla podgrupy *Medium*:

```
## Brand X Brand M
##     169     175
```

- Dla podgrupy *Hard*:

```
## Brand X Brand M
##     171     167
```

1.2. Zadanie 2

W tym zadaniu należało sporządzić tabelę wielodzielczą, uwzględniającą zmienną *Temperature* i *Water Softness*.

Uzyskałem następującą tabelę:

```
structable(Temperature ~ Water_softness, Detergent)%>%
    addmargins()
```

```
##               Temperature
## Water_softness High Low  Sum
##         Soft    104 222  326
##         Medium  126 218  344
##         Hard    139 199  338
##         Sum     369 639 1008
```

1.3. Zadanie 3

W tym zadaniu należało sporządzić wykres kołowy i słupkowy dla zmiennej *Water Softness*

```
data <- apply(Detergent, "Water_softness", sum)
barplot(data, xlab="Water softness", ylab="Ilość osób",
        main="Wykres słupkowy dla zmiennej Water Softness",
        ylim=c(0, 400), col=c("red", "blue", "green"))
```

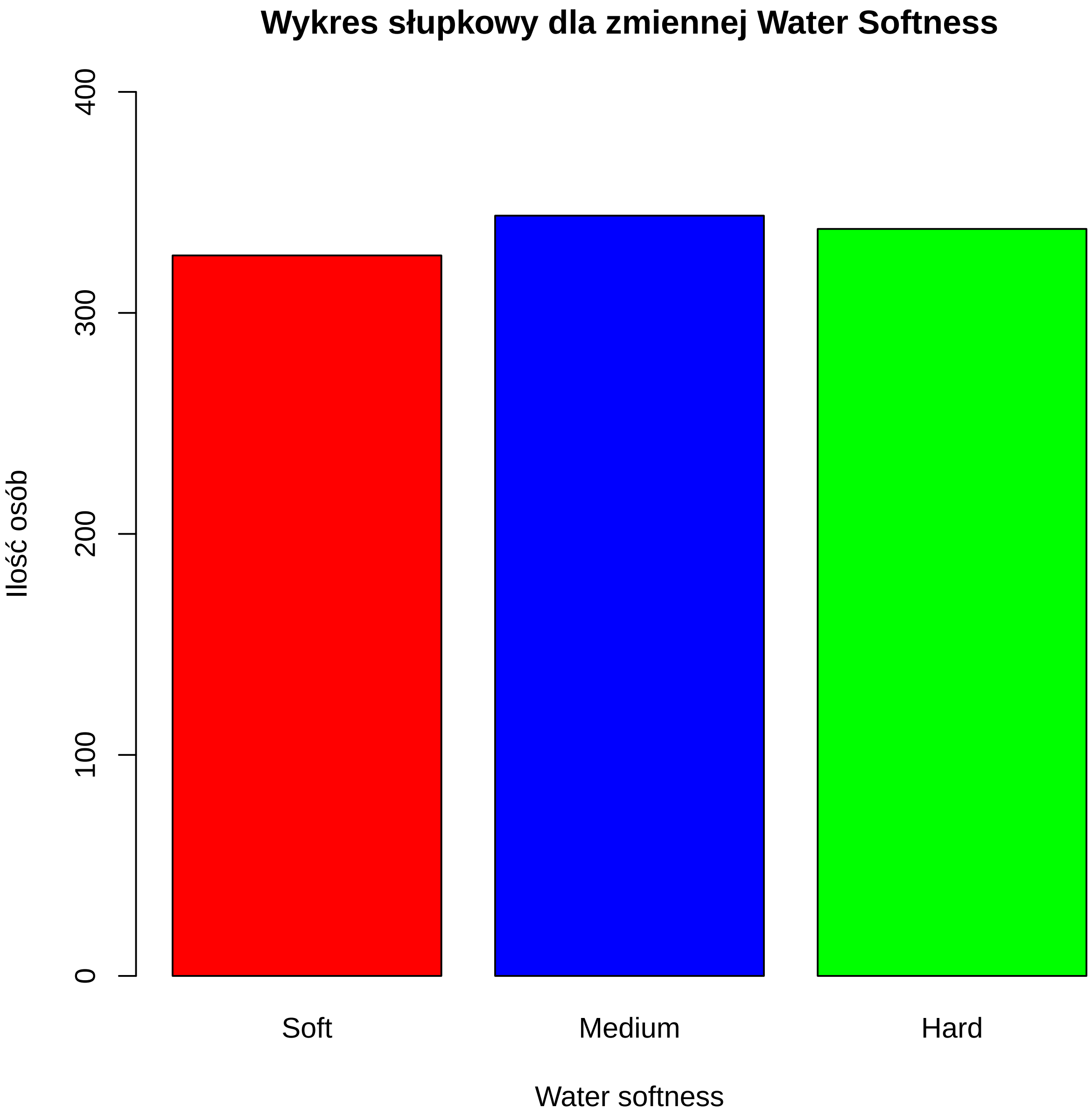
```
pie(data, main="Wykres kołowy dla zmiennej Water Softness",
    col=c("red", "blue", "green"))
```

1.4. Zadanie 4

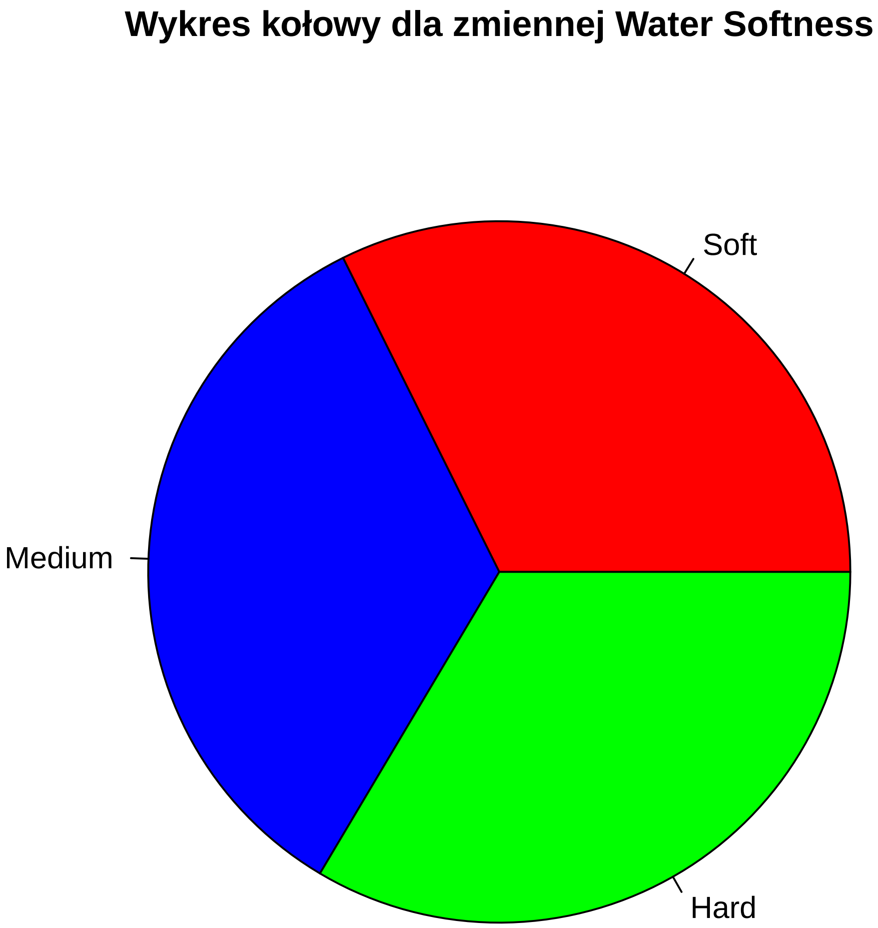
W tym zadaniu należało sporządzić wykresy mozaikowe, odpowiadające rozpatrywanym danym (*Water Softness* i *Preference*, *Preference* i *Temperature*, *Preference* i *User*) i podać krótkie wnioski do wykresu.

```
mosaicplot(~Water_softness + Preference, data=Detergent,
           main="Wykres mozaikowy zmiennych
           Water Softness i Preference")
```

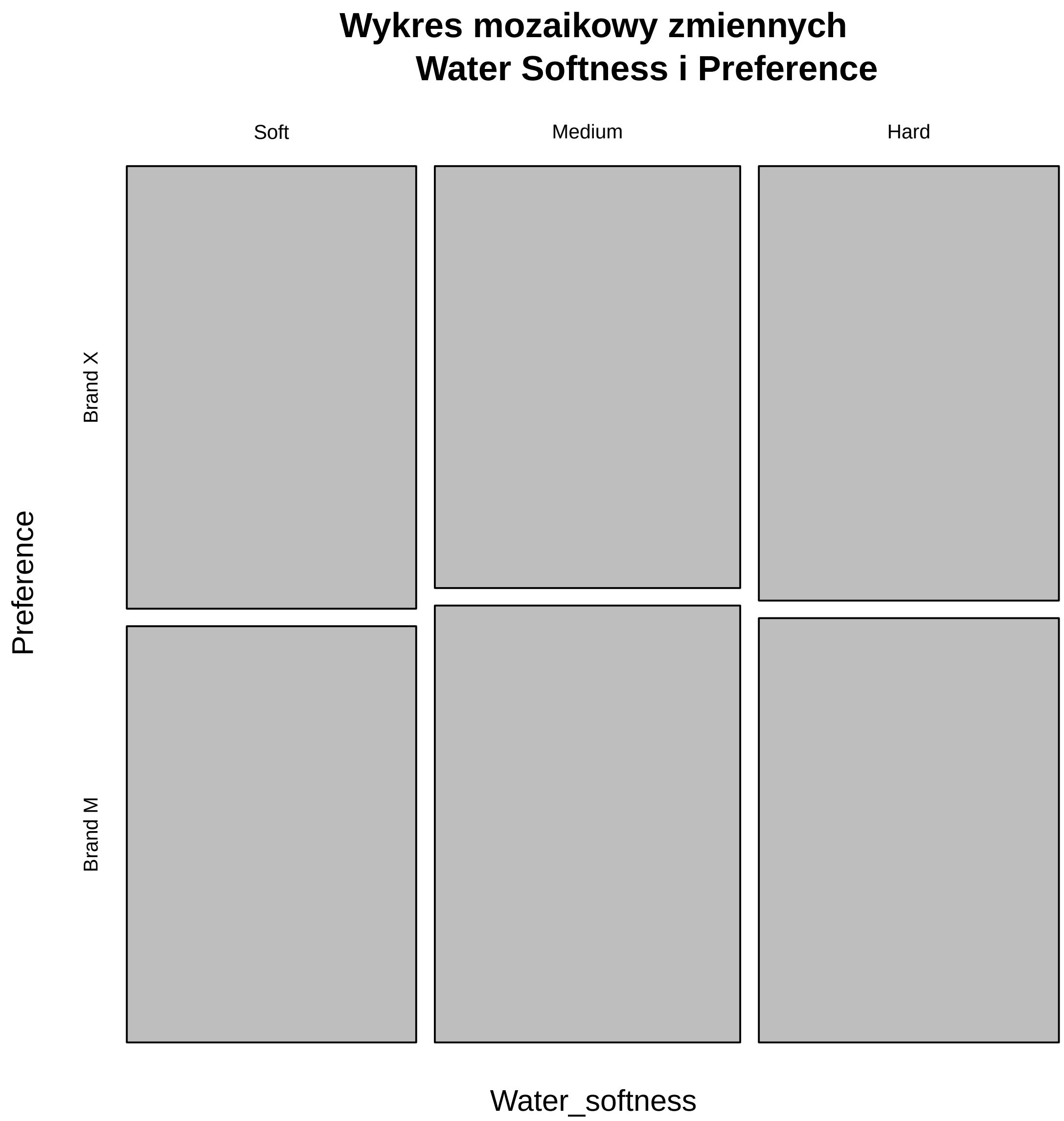
Na rysunku (3) przedstawiłem wykres mozaikowy zmiennych *Water Softness* i *Preference*. Zmienna *Water Softness* przyjmuje trzy wartości (*Soft*, *Medium*, *Hard*), a zmienna *Preference* — 2 (*Brand M*, *Brand X*), dlatego wykres mozaikowy jest podzielony na sześć prostokątów. Na rysunku widać, że możliwe przyjmowane wartości zmiennej *Water Softness* oraz zmiennej*Preference*, nie różnią się bardzo, pod względem wielkości pól na wykresie, co oznacza, że wśród tych zmiennych, nie ma podzbioru, który by się wyróżniał na tle całości. Można jednak



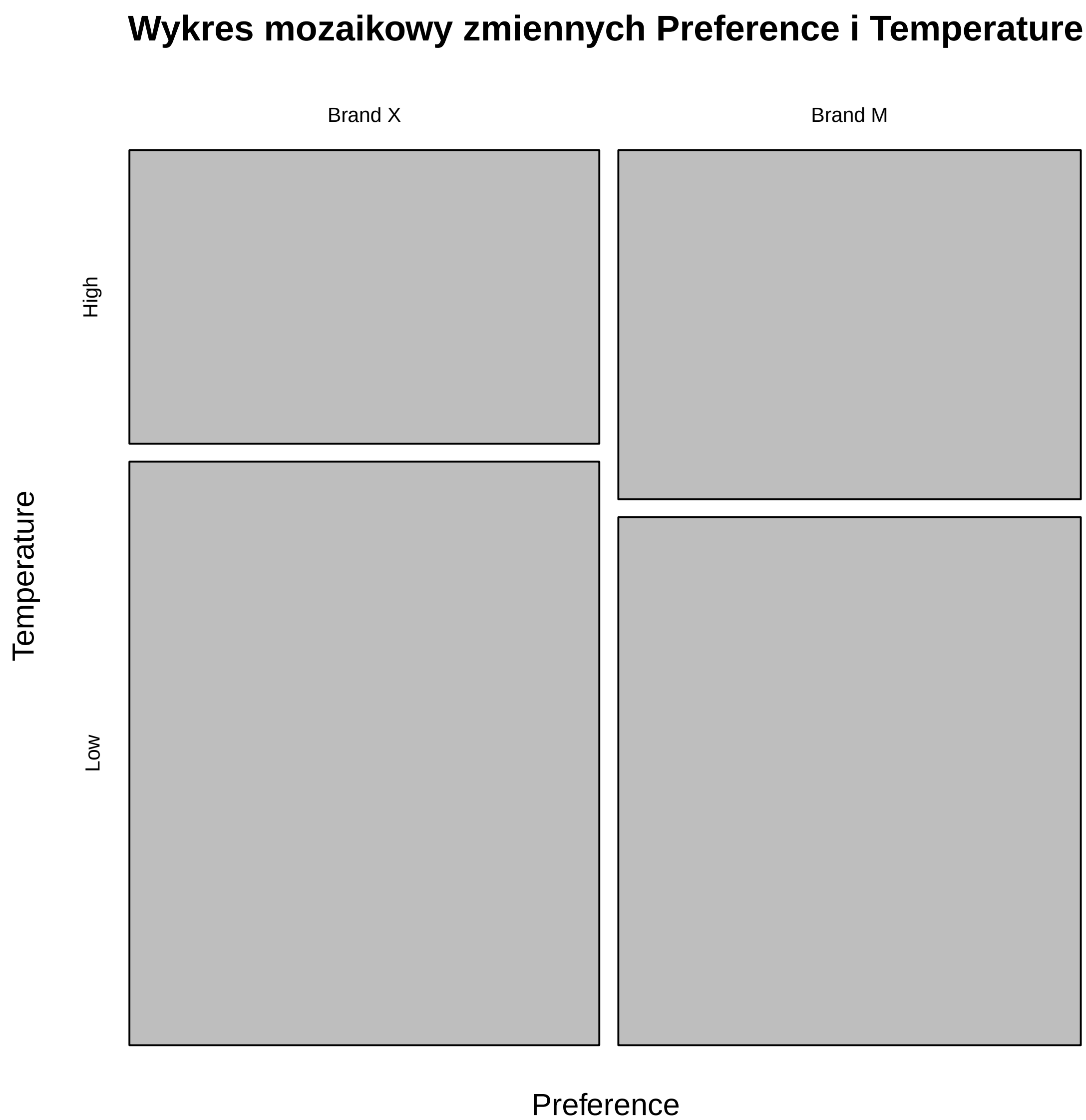
Rysunek 1. Wykres słupkowy zmiennej Water Softness



Rysunek 2. Wykres kołowy zmiennej Water Softness



Rysunek 3. Wykres mozaikowy Zmiennej Water Softnes i Preference



Rysunek 4. Wykres mozaikowy zmiennej Preference i Temperature

zauważyć, że w ankiecie najczęściej wybieranymi odpowiedziami dla rozważanych zmiennych, były: *Medium* i *Brand M*

Na rysunku (4) przedstawiłem wykres mozaikowy zmiennych *Preference* i *Temperature*. Zmienna *Preference* przyjmuje dwie wartości, a zmienna *Temperature* (*Low* i *High*), więc wykres jest podzielony na cztery prostokąty. Widać wyraźnie, że dużo częściej wybieraną w ankiecie wartością temperatury była: *Low*. Z wykresu można odczytać, że najczęściej wybieranymi odpowiedziami w tej ankiecie były: *Low* i *Brand X*, natomiast najrzadziej: *High* i *Brand X*.

Na rysunku (5) przedstawiłem wykres mozaikowy zmiennych *Preference* i *M_User*. Zmienna *Preference* przyjmuje dwie wartości, podobnie jak zmienna *M_User* (*Yes* i *No*), więc wykres jest podzielony na cztery prostokąty. Na wykresie widać, że wartości zmiennej *M_User* dość znacząco różnią się w zależności od przyjmowanych wartości zmiennej *Preference*. Najczęstszymi odpowiedziami, w tej ankiecie, wśród rozważanych zmiennych, były: *Brand X* oraz *No*, natomiast najrzadziej: *Brand X* i *Yes*.

2. Lista 2

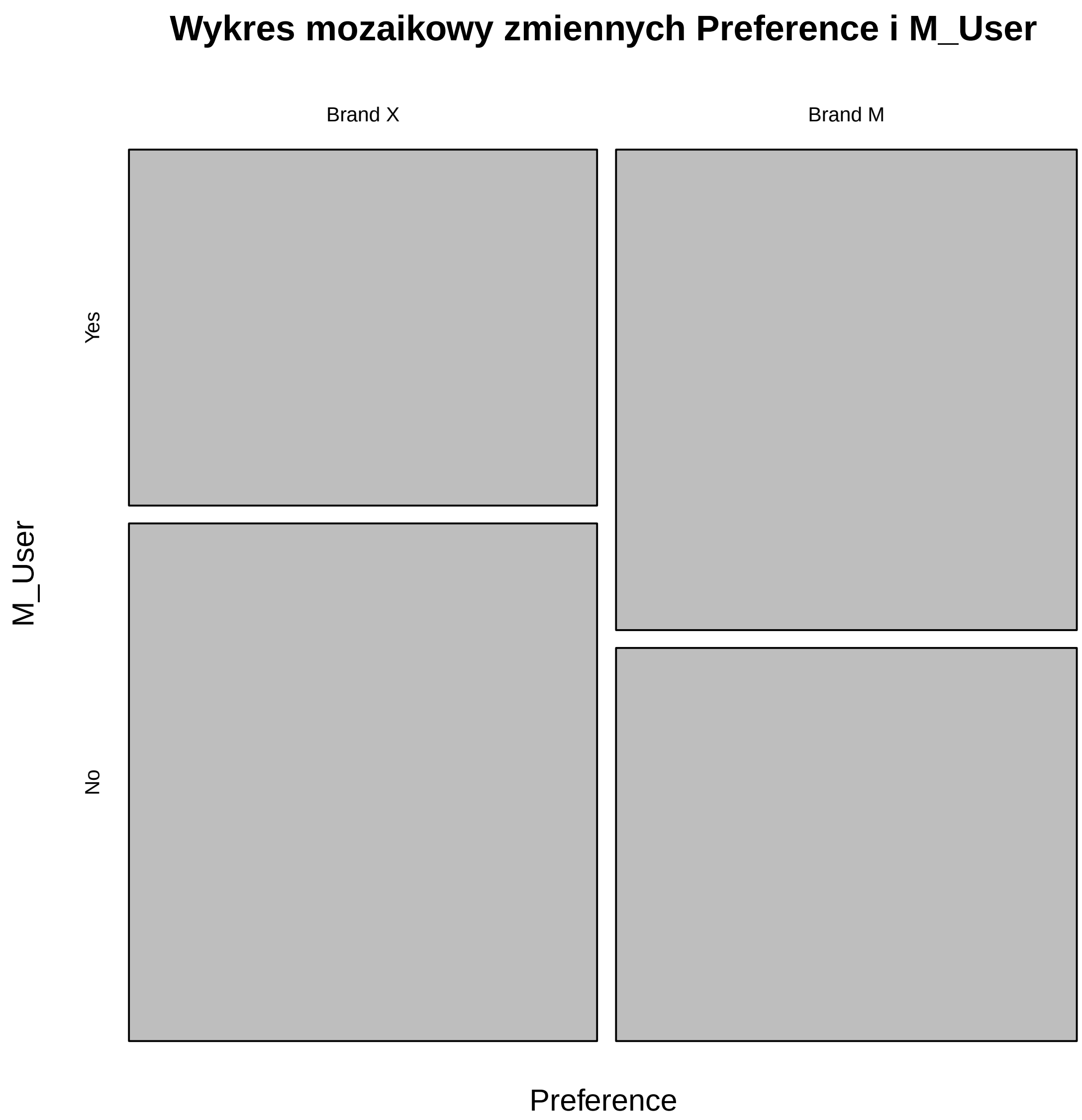
Zaimportowałem bibliotekę, która przydała się do sprawdzenia wariancji w zadaniu 3. — podrozdział (2.3)

```
library("matrixStats")
```

2.1. Zadanie 1

Aby zrobić to zadanie, skorzystałem z danych `mtcars`.

```
data <- mtcars
data

##                      mpg cyl  disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4           21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag       21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710          22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive      21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout   18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2
## Valiant             18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1
## Duster 360          14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4
## Merc 240D           24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2
## Merc 230            22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2
## Merc 280            19.2   6 167.6 123 3.92 3.440 18.30  1  0    4    4
## Merc 280C           17.8   6 167.6 123 3.92 3.440 18.90  1  0    4    4
## Merc 450SE          16.4   8 275.8 180 3.07 4.070 17.40  0  0    3    3
## Merc 450SL          17.3   8 275.8 180 3.07 3.730 17.60  0  0    3    3
## Merc 450SLC         15.2   8 275.8 180 3.07 3.780 18.00  0  0    3    3
## Cadillac Fleetwood  10.4   8 472.0 205 2.93 5.250 17.98  0  0    3    4
## Lincoln Continental 10.4   8 460.0 215 3.00 5.424 17.82  0  0    3    4
## Chrysler Imperial   14.7   8 440.0 230 3.23 5.345 17.42  0  0    3    4
## Fiat 128            32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1
## Honda Civic         30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2
## Toyota Corolla      33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1
## Toyota Corona       21.5   4 120.1  97 3.70 2.465 20.01  1  0    3    1
```



Rysunek 5. Wykres mozaikowy Zmiennej Preference i M User

```
## Dodge Challenger    15.5   8 318.0 150 2.76 3.520 16.87  0  0    3    2
## AMC Javelin         15.2   8 304.0 150 3.15 3.435 17.30  0  0    3    2
## Camaro Z28          13.3   8 350.0 245 3.73 3.840 15.41  0  0    3    4
## Pontiac Firebird    19.2   8 400.0 175 3.08 3.845 17.05  0  0    3    2
## Fiat X1-9           27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1
## Porsche 914-2       26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2
## Lotus Europa        30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2
## Ford Pantera L      15.8   8 351.0 264 4.22 3.170 14.50  0  1    5    4
## Ferrari Dino        19.7   6 145.0 175 3.62 2.770 15.50  0  1    5    6
## Maserati Bora       15.0   8 301.0 335 3.54 3.570 14.60  0  1    5    8
## Volvo 142E          21.4   4 121.0 109 4.11 2.780 18.60  1  1    4    2
```

Celem zadania jest napisanie fragmentu programu, którego celem jest wylosowanie próbki rozmiaru około 1/10 liczby przypadków z wybranej bazy danych ze zwracaniem i bez zwracania.

2.1.1. Dla przypadku ze zwracaniem

W pierwszym kroku wylosowałem numery wierszy dla których odczytamy dane:

```
ind <- sample(nrow(mtcars), 0.1*nrow(mtcars), replace=T)
ind

## [1]  6 30 31
```

Następnie odczytałem dane dla wylosowanych numerów wierszy

```
mtcars[ind, ]

##                mpg cyl disp  hp drat   wt  qsec vs am gear carb
## Valiant       18.1   6  225 105 2.76 3.46 20.22  1  0    3    1
## Ferrari Dino  19.7   6  145 175 3.62 2.77 15.50  0  1    5    6
## Maserati Bora 15.0   8  301 335 3.54 3.57 14.60  0  1    5    8
```

2.1.2. Dla przypadku bez zwracania

W celu wyznaczenia numerów wierszy bez zwracania, wystarczy zmienić argument `replace` funkcji `sample` z wartości `TRUE` na `FALSE`.

```
ind <- sample(nrow(mtcars), 0.1*nrow(mtcars), replace=F)
ind

## [1] 14 19  8
```

```
mtcars[ind, ]

##              mpg cyl  disp  hp drat    wt  qsec vs am gear carb
## Merc 450SLC 15.2   8 275.8 180 3.07 3.780 18.00  0  0    3    3
## Honda Civic 30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2
## Merc 240D   24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2
```

2.2. Zadanie 2

W tym zadaniu należało zaproponować algorytm generowania wektora z rozkładu dwumianowego i sprawdzić jego poprawność oraz napisać program do generowania tych liczb, zgodnie z zaproponowanym algorytmem i sprawdzić czy zaproponowany algorytm działa poprawnie, na podstawie porównania wartości oczekiwanej i wariancji wysymulowanych prób do teoretycznej wartości oczekiwanej i wariancji. Przypominając, jeśli $X \sim \mathcal{B}(n,p)$, to:

$$\mathrm{E}\, X = np \tag{1}$$
$$\mathrm{Var}\, X = np(1-p) \tag{2}$$

2.2.1. Pierwszy algorytm

Chcemy wygenerować zmienną losową X z rozkładu dwumianowego ($X \sim \mathcal{B}(n,p)$). Dla zmiennej losowej Y dyskretnej, zdefiniowanej następująco:

$$P(Y_i = 1) = p \tag{3}$$
$$P(Y_i = 0) = 1 - p \tag{4}$$
$$p \in [0,1] \tag{5}$$

Zmienna losowa X będzie równa co do rozkładu sumie niezależnych zmiennych losowych Y_i ($X \stackrel{\mathrm{d}}{=} \sum_{i=1}^{n} Y_i, Y_i$ — i.i.d.). Korzystając z tego faktu można zaproponować następujący algorytm:

1. Losujemy $U \sim \mathcal{U}(0,1)$
2. Jeżeli $U < p$, to Y=1, w przeciwnym wypadku $Y = 0$
3. Powtarzamy ktoki 1 i 2 n razy
4. Wstawiamy $X = \sum_{i=1}^{n} Y_i$

Zaproponowany program:

```
binom.rv <- function(n, p, N){
    sapply(1:N, function(...){
        sum(runif(n) < p)
    })
}
```

Sprawdźmy teraz poprawność zaproponowanego kodu:

```
Y1 <- binom.rv(10, 0.4, 1000) # Rozkład B(10, 4)
mean(Y1) # Wartość oczekiwana teoretyczna: 4

## [1] 3.938

var(Y1) # Wariancja teoretyczna: 2.4

## [1] 2.460617

Y2 <- binom.rv(50, 0.2, 1000) # Rozkład B(50, 0.2)
mean(Y2) # Wartość oczekiwana teoretyczna: 10

## [1] 9.941

var(Y2) # Wariancja teoretyczna: 8

## [1] 7.887406

Y3 <- binom.rv(100, 0.6, 1000) # Rozkład B(100, 0.6)
mean(Y3) # Wartość oczekiwana teoretyczna: 60
```

```
## [1] 60.154
var(Y3) # Wariancja teoretyczna 24
## [1] 21.96825
```

Widać, że zaproponowana metoda sprawdza się całkiem dobrze. Wartości oczekiwane i wariancje dla przykładowych trzech prób, są bliskie wartości oczekiwanej i wariancji teoretycznej.

2.2.2. Drugi algorytm

W drugim algorytmie skupiłem się na sposobie generowania zmiennej losowej dyskretnej, metodą akceptacji-odrzucenia. Celem jest wyznaczenie zmiennej losowej $X \sim \mathcal{B}(n,p)$ W tej metodzie mamy dwa założenia:

1. potrafimy efektywnie generować inną zmienną losową Y, o rozkładzie: $q_i = P(Y = i)$, tak aby zmienne losowe X i Y przyjjmowały wartości z tego samego zbioru
2. potrafimy wyznaczyć stałą $0 < c < \infty$, taką że $\max \frac{p_i}{q_i} \leqslant c$

Algorytm:

1. Generuj jedną realizację zmiennej losowej Y
2. Generuj $U \sim \mathcal{U}(0,1), \quad U \perp Y$
3. Jeśli $U \leqslant \frac{p_Y}{cq_Y}$, to zwróć $X = Y$, w przeciwnym wypadku wróć do 1.

W celu napisania programu, wprowadziłem funkcję pomocniczą `pY`, która wyznacza wartość p_Y:

```
pY <- function(Y, p){
    if (Y == 1){
        p
    } else {
        1-p
    }
}
```

Zaproponowany program:

```
binom.rv2 <- function(n, p, N){
    sapply(1:N, function(...){
        a <- c()
        for (i in 1:n){
            while (length(a) != n){
                Y <- round(runif(1))
                c <- 2
                U <- runif(1)
                if (U <= pY(Y, p)/(c*0.5)){
                    a <- append(a, Y)
                }
            }
        }
        sum(a)
    })
}
```

Sprawdźmy teraz poprawność zaproponowanego kodu:

```
Y4 <- binom.rv2(10, 0.4, 1000)
                               # Rozkład B(10, 4)
mean(Y4) # Wartość oczekiwana teoretyczna: 4

## [1] 4.013

var(Y4) # Wariancja teoretyczna: 2.4

## [1] 2.461292

Y5 <- binom.rv2(50, 0.2, 1000)
                               # Rozkład B(50, 0.2)
mean(Y5) # Wartość oczekiwana teoretyczna: 10

## [1] 10.083

var(Y5) # Wariancja teoretyczna: 8

## [1] 7.647759

Y6 <- binom.rv2(100, 0.6, 1000)
                               # Rozkład B(100, 0.6)
mean(Y6) # Wartość oczekiwana teoretyczna: 60

## [1] 59.95

var(Y6) # Wariancja teoretyczna 24

## [1] 25.55906
```

Na podstawie uzyskanych symulacyjnie wyników, możemy stwierdzić, że zaproponowany algorytm działa całkiem dobrze. Wartości $\mathrm{E}\,X$ i $\mathrm{Var}\,X$ wyznaczonych symulacyjnie prób, są bliskie teoretycznym wartością $\mathrm{E}\,X$ i $\mathrm{Var}\,X$.

2.3. Zadanie 3

W tym zadaniu należało zaproponować algorytm generowania wektora z rozkładu wielomianowego i udowodnić jego poprawność, na podstawie wysymulowanej wielokrotnie próby i porównania jej średniej wartości oczekiwanej i wariancji z teoretyczną wartością oczekiwaną i wariancją oraz napisać program do generowania tych wektorów, zgodnie z zaproponowanym algorytmem.

2.3.1. Pierwszy algorytm

Zaproponowany algorytm wygląda następująco:

1. Zadeklaruj pusty wektor X, o długości k (liczba odpowiedzi na pytanie w ankiecie), składający się z samych zer
2. Generuj indeks I, $P(I = i) = p_i, i = 1, \dots, k$
3. Wstaw $X[I] = X[I] + 1$
4. Powtórz kroki $1 - 3$, n razy

```
multinom.rv <- function(n, p, N=1) {
 sapply(1:N, function(...){
  k <- length(p)
  X <- numeric(k)
```

```
  I <- sample(x=1:k, size=n, replace = TRUE, prob = p)
   for (i in 1:n) {
   X[I[i]] = X[I[i]] + 1
   }
  X
  }
 )
}
```

Do sprawdzenia wariancji, będziemy potrzebować funkcji `rowVars` z zaimportowanej biblioteki '`matrixStats`' Zobaczmy teraz czy zaproponowany kod działa poprawnie:

```
Y7 <- multinom.rv(10, c(0.5,0.1,0.4), 1000)
                         # Rozkład B(10, 4)
rowMeans(Y7) # Wartość oczekiwana teoretyczna: (5, 1, 4)

## [1] 4.927 1.047 4.026

rowVars(Y7) # Wariancja teoretyczna: (2.5, 0.9, 2.4)

## [1] 2.4761471 0.9757668 2.2936176

Y8 <- multinom.rv(50, c(0.25,0.45,0.3), 1000)
                         # Rozkład B(50, 0.2)
rowMeans(Y8) # Wartość oczekiwana teoretyczna: (12.5, 22.5, 15)

## [1] 12.418 22.596 14.986

rowVars(Y8) # Wariancja teoretyczna: (9.375, 12,375, 10.5)

## [1]  9.809085 12.317101 10.165970

Y9 <- multinom.rv(100, c(0.3,0.4,0.3), 1000)
                         # Rozkład B(100, 0.6)
rowMeans(Y9) # Wartość oczekiwana teoretyczna: (30, 40, 30)

## [1] 29.847 40.097 30.056

rowVars(Y9) # Wariancja teoretyczna: (21, 24, 21)

## [1] 21.20480 23.45304 20.78565
```

2.3.2. Drugi algorytm

Drugie podejście jest podobne do pierwszego, natomiast jest kilka różnic. Prawdopodobieństwo uzyskania każdej odpowiedzi na pytanie w ankiecie możemy umieścić na pewnym przedziale, na odcinku $[0, 1]$. Wtedy na podstawie wygenerowanej zmiennej losowej z rozkładu jednostajnego na odcinku $[0, 1]$, możemy podać odpowiedź, która padła na pytanie.

Zaproponowany algorytm:

1. Zadeklaruj pusty wektor X, o długości k (liczba odpowiedzi na pytanie w ankiecie), składający się z samych zer
2. Generuj $U \sim U(0, 1)$
3. Wstaw $I = 1$ $P = p_1$ (prawdopodobieństwo uzyskania na przykład pierwszej odpowiedzi w ankiecie)
4. Jeśli $U > P$ to wstaw $I = I + 1$ i $P = P + p_I$, w przeciwnym wypadku powtórz krok.
5. Wstaw X[i] = X[i] + 1

6. Powtórz kroki $2-5$ n razy.

Kod do zaproponowanego algorytmu:

```
multinom.rv2 <- function(n, p, N){
  sapply(1:N, function(...){
    vect <- numeric(length(p))
    for (j in 1:n){
    U <- runif(1)
    P <- p[1]
    i <- 1
    while (U > P) {
      i <- i + 1
      P <- P + p[i]
    }
    vect[i] <- vect[i] + 1
    }
  vect
  })
}
```

Sprawdźmy poprawność działania kodu:

```
Y10 <- multinom.rv2(10, c(0.5,0.1,0.4), 1000)
                        # Rozkład B(10, 4)
rowMeans(Y10) # Wartość oczekiwana teoretyczna: (5, 1, 4)

## [1] 5.022 1.012 3.966

rowVars(Y10) # Wariancja teoretyczna: (2.5, 0.9, 2.4)

## [1] 2.4579740 0.9568128 2.3872312

Y11 <- multinom.rv2(50, c(0.25,0.45,0.3), 1000)
                        # Rozkład B(50, 0.2)
rowMeans(Y11) # Wartość oczekiwana teoretyczna: (12.5, 22.5, 15)

## [1] 12.487 22.564 14.949

rowVars(Y11) # Wariancja teoretyczna: (9.375, 12,375, 10.5)

## [1]  9.777609 12.742647 10.683082

Y12 <- multinom.rv2(100, c(0.3,0.4,0.3), 1000)
                        # Rozkład B(100, 0.6)
rowMeans(Y12) # Wartość oczekiwana teoretyczna: (30, 40, 30)

## [1] 30.018 40.014 29.968

rowVars(Y12) # Wariancja teoretyczna: (21, 24, 21)

## [1] 20.98666 26.10391 21.29627
```

Na podstawie uzyskanych wyników, można stwierdzić, że zaproponowany algorytm działa poprawnie.

2.4. Zadanie 4

Zadanie polega na stworzeniu propozycji badania ankietowego.

Cel badania: Czy zwiększyły się preferencje dotyczące dań wegeteriańskich i wegańskich wśród studentów i pracowników? Celem badania ankietowego jest sprawdzenie preferencji żywieniowych i stworzenie na tej podstawie odpowiedniego menu w stołówce studenckiej PWr.

Grupa docelowa: Studenci i pracownicy PWr.

Sposób zbierania danych: Ankieta zostałaby prowadzona online w celu łatwiejszego dostępu do analizowanych danych. Studenci przy zamawianiu obiadu proszeni byliby o zeskanowanie kodu QR z linkiem do ankiety, oprócz tego również zostałyby wysłane maile. Wykorzystanym schematem losowania byłoby losowanie warstwowe. Wyodrębnione warstwy to przede wszystkim osobno grupa studentów i grupa pracowników, ale można byłoby podzielić również bardziej dokładnie na mniejsze przedziały wiekowe oraz ze względu na płeć.

Ankieta zawierać będzie pytań z odpowiedziami w 5-stopniowej skali oraz metryczkę.

Propozycja kwestionariusza:

Metryczka:

1. Student/Pracownik:
2. Płeć:
3. Wiek:
4. Rok studiów/Rok pracy na uczelni:
5. Wydział:

Pytania:

1. Jak często korzystasz z posiłków w SKS-ie?
   a) bardzo rzadko
   b) rzadko
   c) czasem
   d) często
   e) bardzo często

2. Jak często wybierasz opcję bezmięsną w SKS-ie?
   a) bardzo rzadko
   b) rzadko
   c) czasem
   d) często
   e) bardzo często

3. Czy chciałbyś, chciałabyś, aby została zwiększona liczba dań wegetariańskich?
   a) zdecydowanie nie
   b) nie
   c) nie wiem
   d) tak
   e) zdecydowanie tak

4. Czy korzystałbyś z oferty posiłków wegańskich?
   a) zdecydowanie nie
   b) nie
   c) nie wiem
   d) tak
   e) zdecydowanie tak

Wyniki: Wyniki zostałyby przedstawione w tabelach oraz na wykresach słupkowych.

3. Lista 3

Zaimportowałem biblioteki, które przydadzą się do rozwiązywanych zadań:

```
library(binom)
library(stats)
```

3.1. Zadanie 1

W tym zadaniu należało przeprowadzić symulację, w celu porównania prawdopodobieństwa pokrycia i długości przedziałów ufności Cloppera-Pearsona (exact), Walda (asymptotic) i dowolnego wybranego typu przedziału ufności, zaimplementowanego w funkcji *binom.confint*. Należało przyjąć poziom ufności — 0.95, różne rozmiary próby i różne wartości prawdopodobieństwa p oraz wyniki umieścić na rysunkach i w tabelach i wyciągnąć wnioski.

Jako dowolny przedział ufności, wziąłęm przedziay Agrestiego-Coulla. Przedział ten jest przedziałem ufności punktowo asymptotycznym. Wprowadzając oznaczenia:

$$\kappa(\alpha) = z\left(1 - \frac{\alpha}{2}\right) \tag{6}$$

$$\tilde{X} = \sum_{i=1}^{n} X_i + \frac{\kappa^2(\alpha)}{2} \tag{7}$$

$$\tilde{n} = n + \kappa^2(\alpha) \tag{8}$$

$$\tilde{p} = \frac{\tilde{X}}{\tilde{n}} \tag{9}$$

$$\tilde{q} = 1 - \tilde{p} \tag{10}$$

gdzie z to dystrybuanta rozkładu $\mathcal{N}(0,1)$, przedziały ufności Agrestiego-Coulla są postaci $[T_L^{AC}, T_U^{AC}]$, gdzie:

$$T_L^{AC} = \tilde{p} - \kappa(\alpha)(\tilde{p}\tilde{q})^{\frac{1}{2}}\tilde{n}^{-\frac{1}{2}} \tag{11}$$

$$T_U^{AC} = \tilde{p} + \kappa(\alpha)(\tilde{p}\tilde{q})^{\frac{1}{2}}\tilde{n}^{-\frac{1}{2}} \tag{12}$$

Aby porównać prawdopodobieństwa pokrycia i długości rozważanych przedziałów ufności posłużyłem się symulacją Monte-Carlo. W tym celu napisałem funkcje:

```
funkcja <- function(N, s, p, method){
  X <- rbinom(N, s, p)
  pr_ufnosci <- binom.confint(X, s, methods=method)
  czy_p_w_pr_ufnosci <-ifelse(pr_ufnosci$lower < p
                              & pr_ufnosci$upper > p, 1, 0)
```

```r
  dl_p_uf <- pr_ufnosci$upper - pr_ufnosci$lower
  c(mean(czy_p_w_pr_ufnosci), mean(dl_p_uf))
}

dane_do_wykresow <- function(MC, n, pokrycie_dl_przedzialu){
  p <- seq(0, 1, by=0.01)
  metody <- c("asymptotic", "exact", "agresti-coull")
  wyniki <- matrix(0, 3, length(p))
  for (i in 1:3){
    wyniki[i, ] <- sapply(1:length(p), function(j){
      funkcja(MC, n, p[j], metody[i])[pokrycie_dl_przedzialu]})
  }
  wyniki
}
```

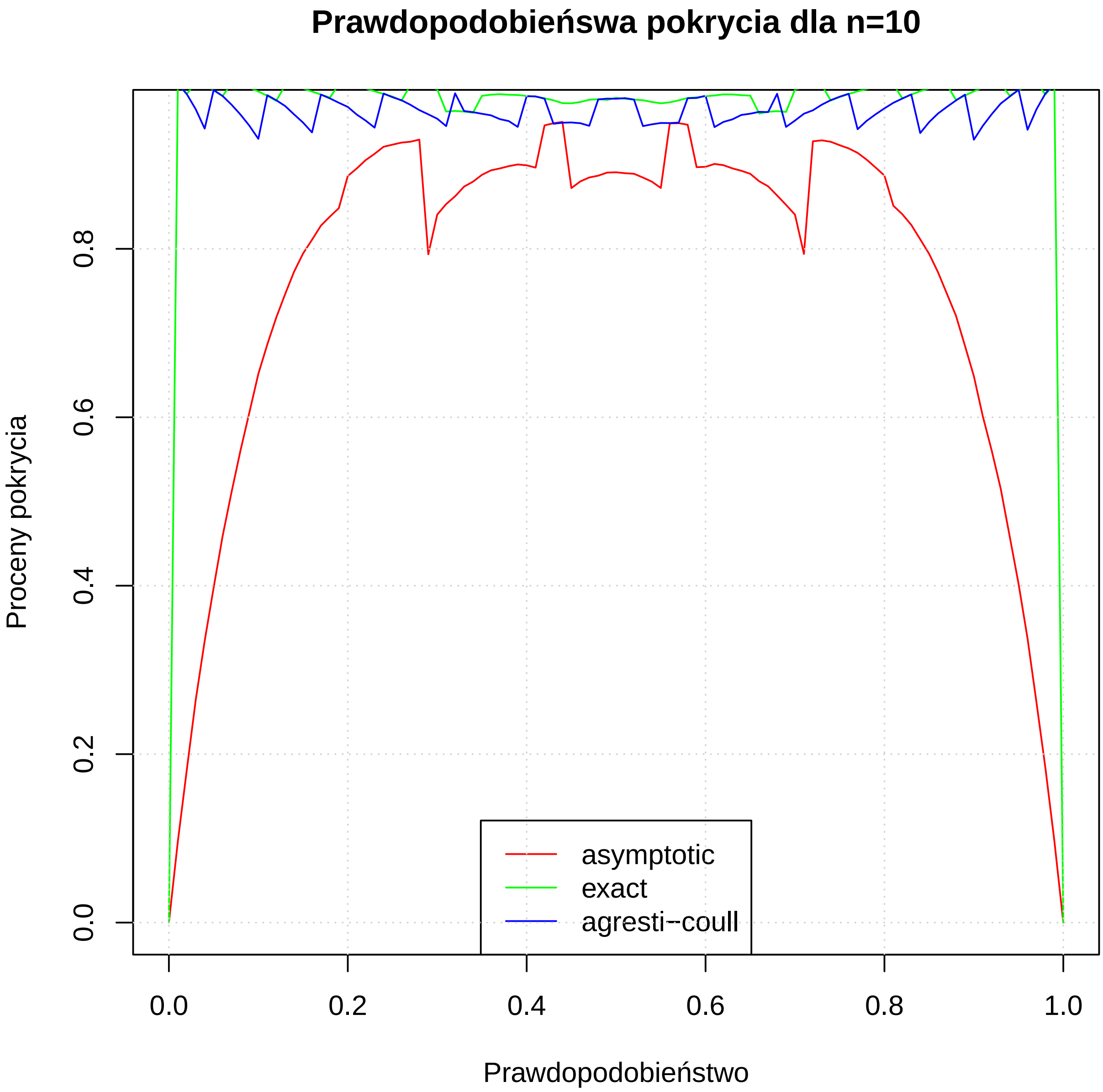
```r
p <- seq(0, 1, by=0.01)
rozmiary_proby <- c(10, 20, 50, 100)
MC <- 100000
```

```r
dane <- dane_do_wykresow(MC,rozmiary_proby[1], 1)

plot(p, dane[1, ], type="l", col="red",
   main="Prawdopodobieńswa pokrycia dla n=10",
   xlab = "Prawdopodobieństwo",
   ylab = "Proceny pokrycia", lwd=1)
lines(p, dane[2, ], type="l", col="green", lwd=1)
lines(p, dane[3, ], type="l", col="blue", lwd=1)
legend("bottom", legend = c("asymptotic",
                          "exact", "agresti-coull"),
     lwd=1,col=c("red", "green", "blue"))
grid()
```

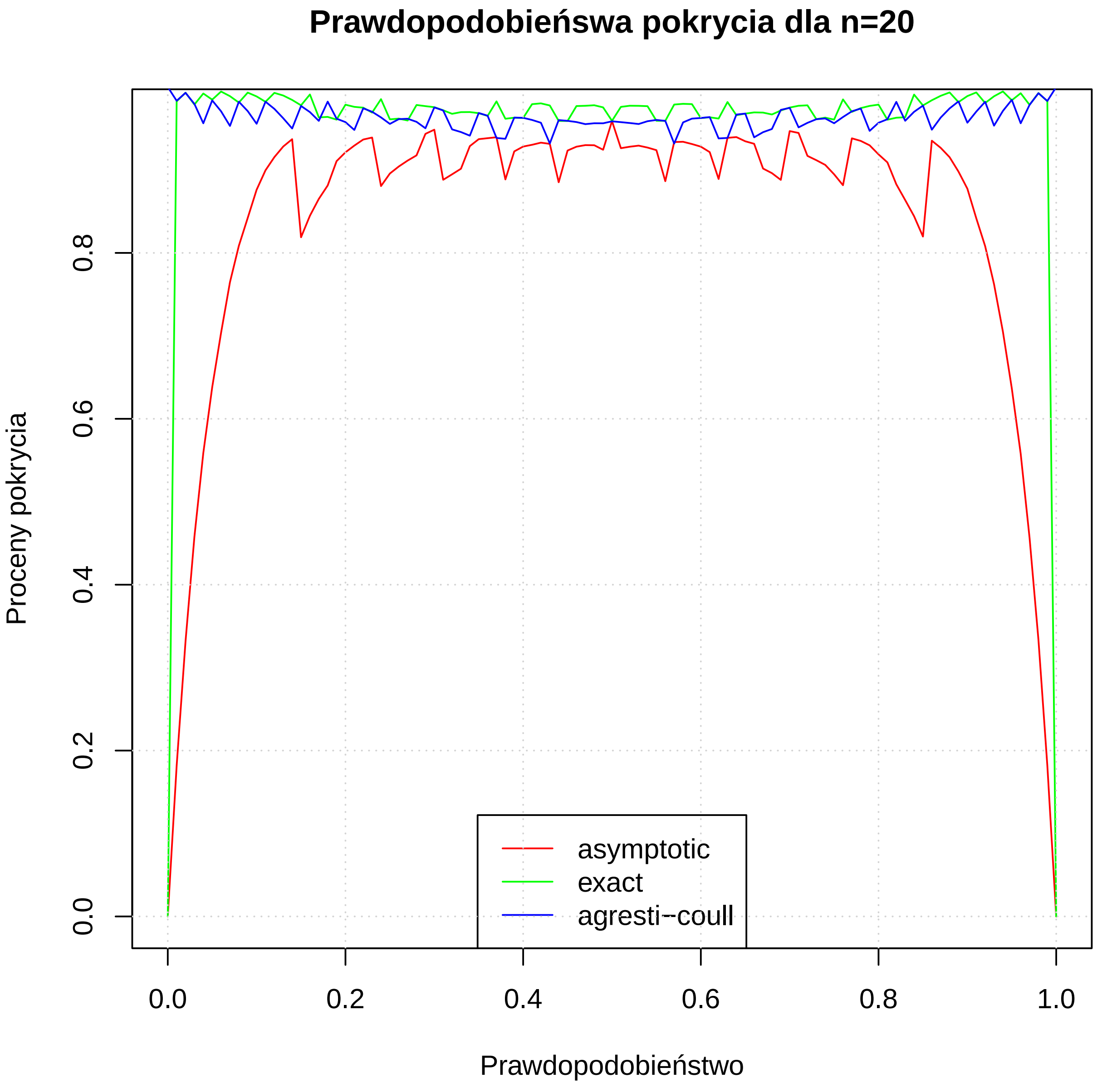
Na rysunkach (6), (7), (8) i (9) przedstawiłem wykresy prawdopodobieństwa pokrycia dla rozważanych przedziałów ufności, dla odpowiednio $n = 10$, $n = 20$, $n = 50$ i $n = 100$ rozmiarów próby. Wykresy powstały na podstawie symulacji Monte-Carlo, dla 10^5 powtórzeń.

Wnioski jakie można wyciągnąć na podstawie wykresów są takie, że im mniejsza próba, tym mniejszy procent pokrycia. Szczególnie taki wynik możemy zaobserwować dla skrajnych wartości. W zestawieniu trzech typów przedziałów ufności najsłabiej wypadają przedziały Walda, czyli asymptotyczne. Nawet przy zwiększaniu próby znacząco odstają od przedziałów Cloppera-Pearsona i Agrestiego-Coulla. Te dwa pozostałe typy przedziałów wypadają bardzo porównywalnie.

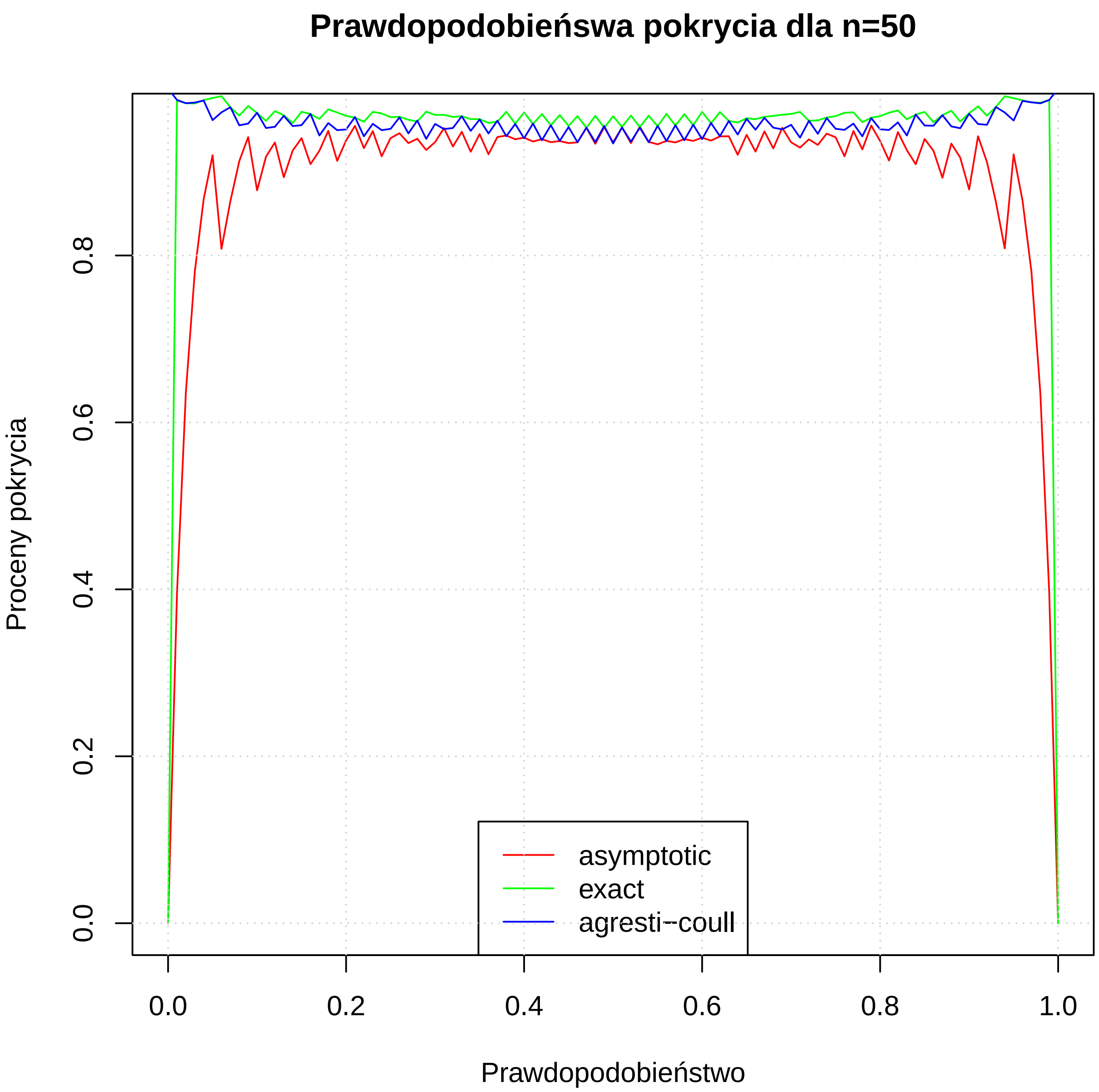
```r
dane <- dane_do_wykresow(MC, rozmiary_proby[1], 2)

plot(p, dane[1, ], type="l", col="red",
   main="Długości przedziałów dla n=10",
   xlab = "Prawdopodobieństwo",
   ylab = "Długość przedziału", lwd=1)
lines(p, dane[2, ], type="l", col="green", lwd=1)
```



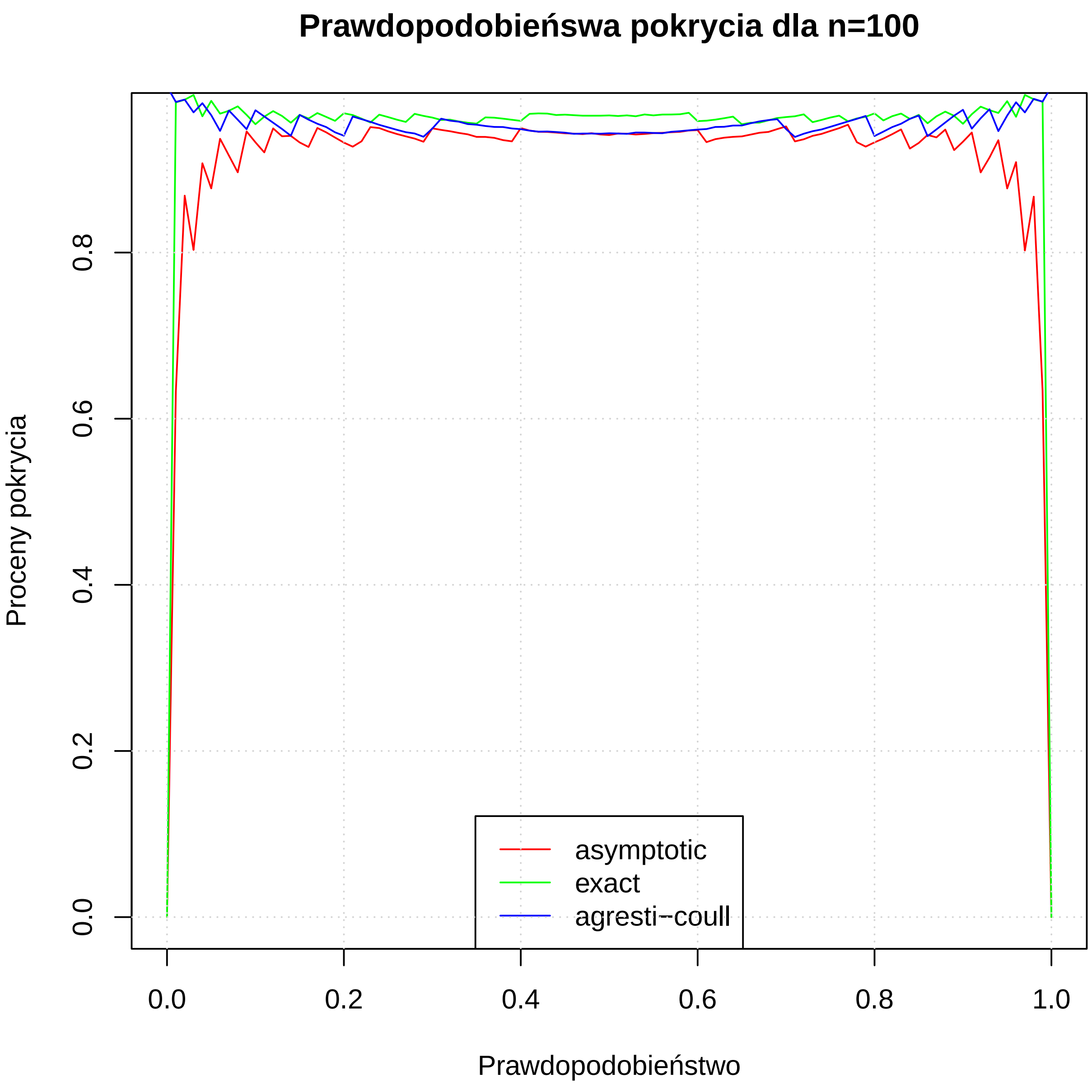
Rysunek 6. Prawdopodobieństwa pokrycia rozważanych przedziałów ufności, dla n=10



Rysunek 7. Prawdopodobieństwa pokrycia rozważanych przedziałów ufności, dla n=20



Rysunek 8. Prawdopodobieństwa pokrycia rozważanych przedziałów ufności, dla n=50



Rysunek 9. Prawdopodobieństwa pokrycia rozważanych przedziałów ufności, dla n=100

```
lines(p, dane[3, ], type="l", col="blue", lwd=1)
legend("bottom", legend = c("asymptotic",
                            "exact", "agresti-coull"),
     lwd=1,col=c("red", "green", "blue"))
grid()
```

Na rysunkach (10), (11), (12) i (13) przedstawiłem wykresy długości przedziałów dla odpowiednio $n = 10$,$n = 20$, $n = 50$ i $n = 100$ rozmiarów próby. Wykresy powstały na podstawie symulacji Monte-Carlo, dla 10^5 powtórzeń.

Im większa próba tym wyniki dla trzech typów przedziałów są bardziej zbliżone. Większe różnice można zaobserwować dla mniejszych prób, tam zdecydowanie lepiej od innych typów wypadają przedziały Agrestiego-Coulla, ponieważ są krótsze. Nie dotyczy to wartości skrajnych, ale wówczas, mimo że asymptotyczne przedziały są krótsze, to ich procent pokrycia nie jest wystarczający, by brać je pod uwagę.

3.2. Zadanie 2

W tym zadaniu należało wyznaczyć realizacje przedziałów ufności, na poziomie ufności $\alpha = 0.95$, dla czterech różnych zadanych prawdopodobieństw stosowania leków (w zadanym przedziale wiekowym, lub dla wszystkich osób biorących udział w ankiecie).

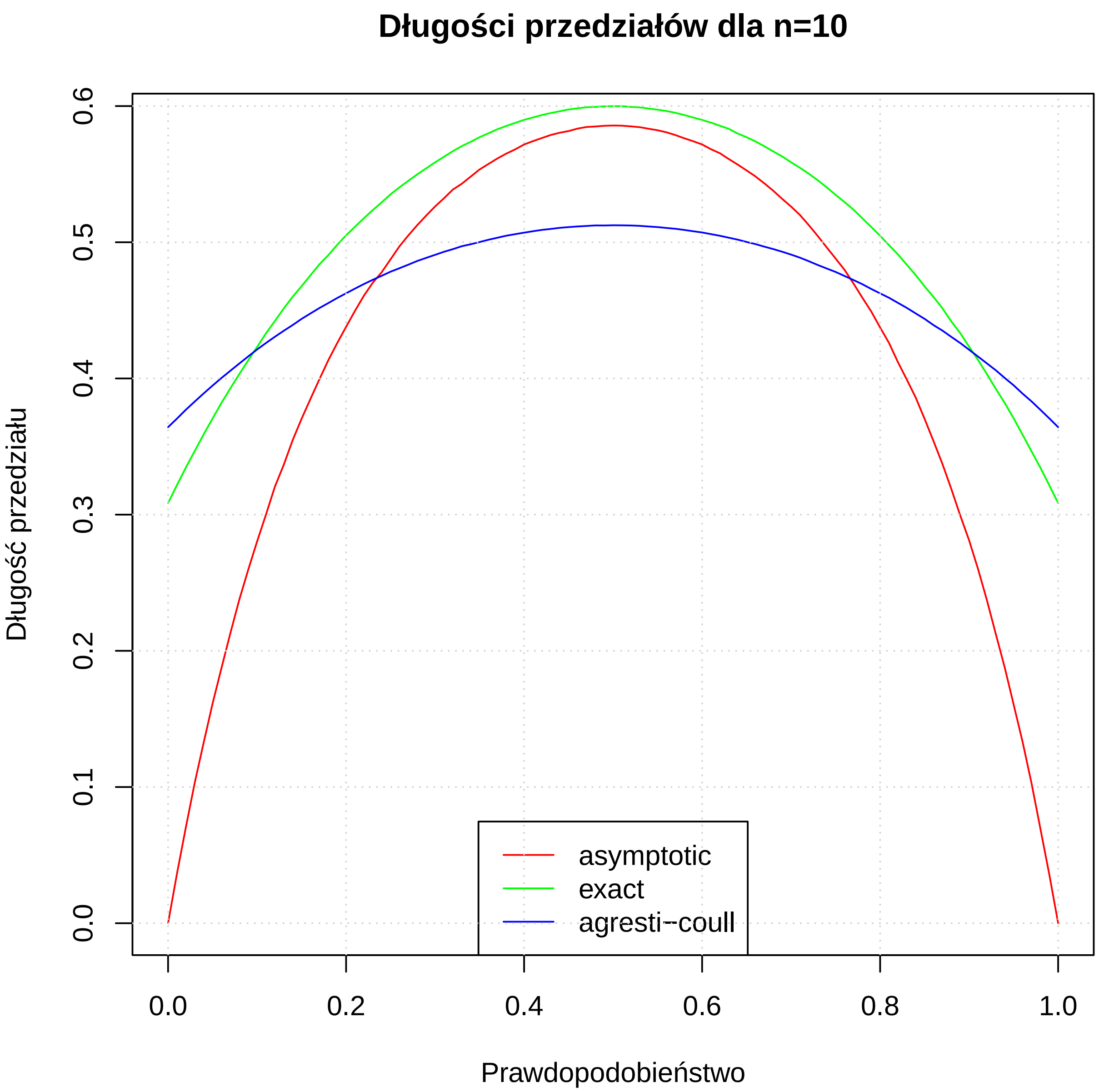
Dane do zadania przedstawia tabela (1)

	Wiek ankietowanych			
Lek	do lat 35	od 36 do 55	powyżej 55	Suma
Ibuprom	35	0	0	35
Apap	22	22	0	44
Paracetamol	15	15	15	45
Ibuprofen	0	40	10	50
Panadol	18	3	5	26
Suma	90	80	30	200

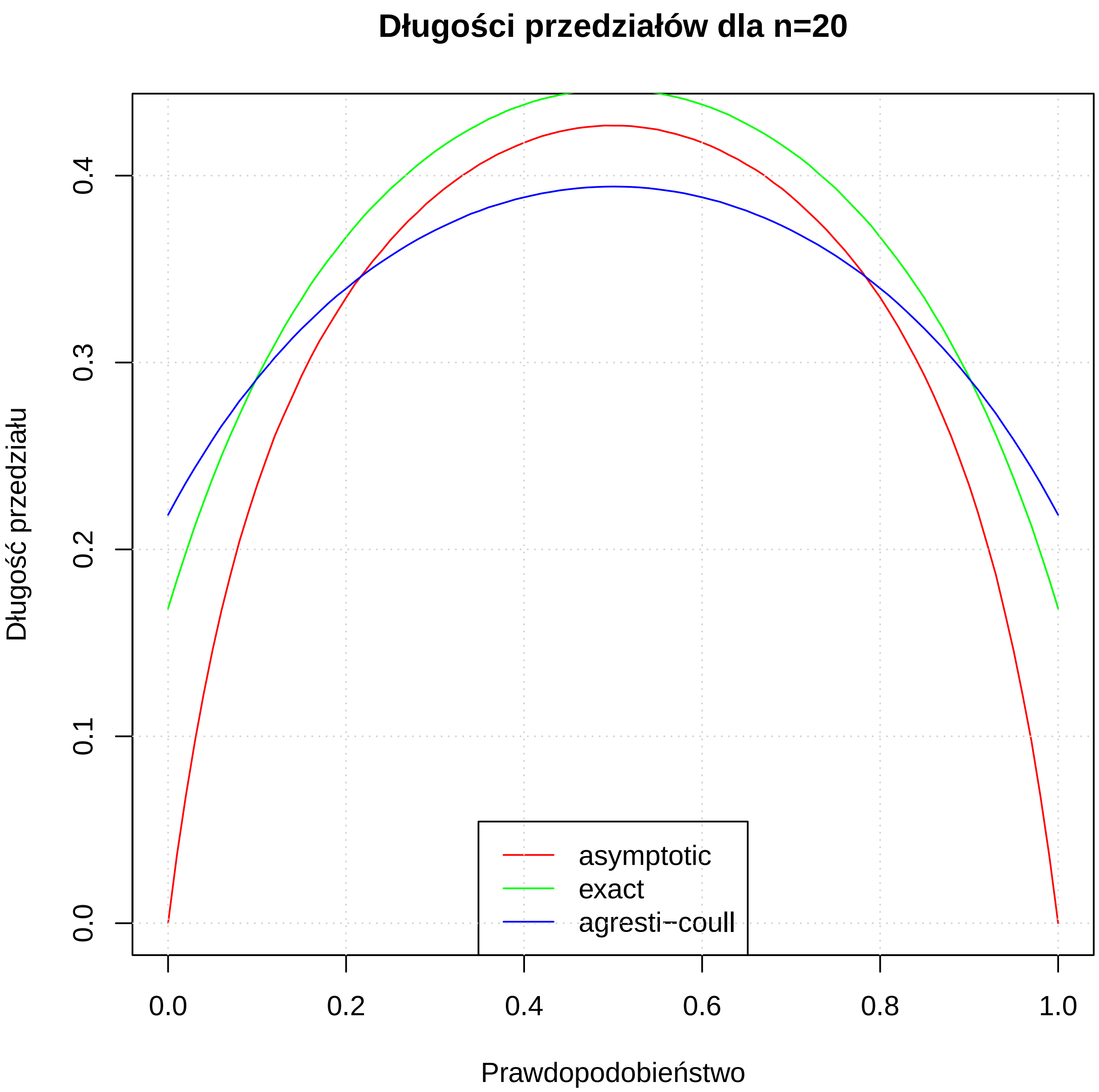
Tabela 1. Dane do zadania

- W pierwszym podpunkcie należało wyznaczyć przedziały ufności, dla prawdopodobieństwa stosowania leku ibuprofen (bez względu na grupę wiekową). Odczytując z tabeli (1) mamy: $x = 50$, $n = 200$.

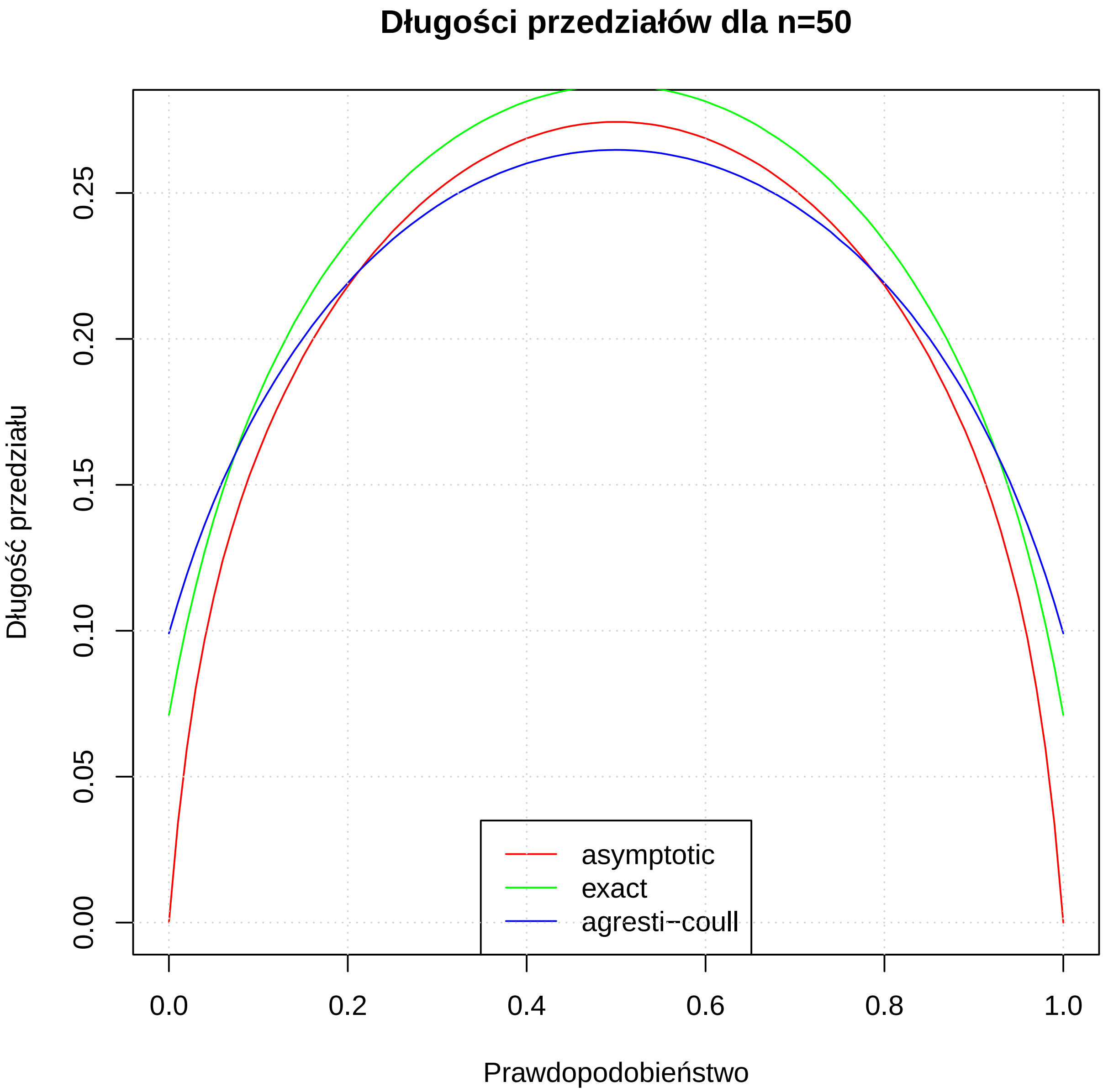
```
x1 <- 50
n1 <- 200
data1 <- binom.confint(x1, n1)
df <- data.frame(data1)
df

##            method  x   n      mean     lower     upper
## 1   agresti-coull 50 200 0.2500000 0.1948993 0.3145233
## 2      asymptotic 50 200 0.2500000 0.1899886 0.3100114
## 3           bayes 50 200 0.2512438 0.1923105 0.3115641
## 4         cloglog 50 200 0.2500000 0.1923621 0.3116476
## 5           exact 50 200 0.2500000 0.1916072 0.3159628
## 6           logit 50 200 0.2500000 0.1948697 0.3146322
```



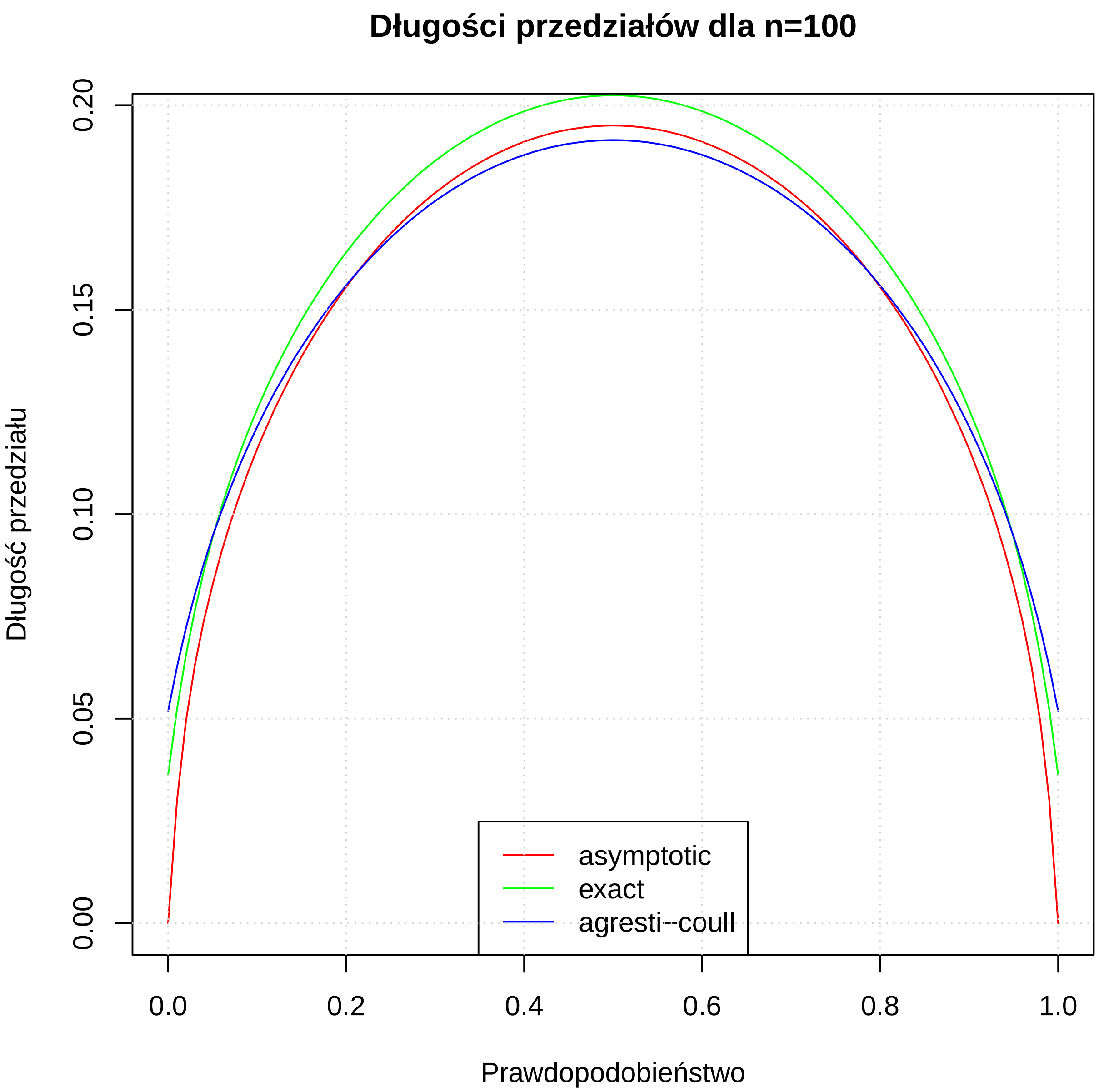
Rysunek 10. Długości przedziałów ufności rozważanych testów, dla n=10



Rysunek 11. Długości przedziałów ufności rozważanych testów, dla n=20



Rysunek 12. Długości przedziałów ufności rozważanych testów, dla n=50



Rysunek 13. Długości przedziałów ufności rozważanych testów, dla n=100

```
## 7          probit 50 200 0.2500000 0.1939760 0.3136105
## 8         profile 50 200 0.2500000 0.1934176 0.3129498
## 9             lrt 50 200 0.2500000 0.1934316 0.3129489
## 10      prop.test 50 200 0.2500000 0.1928239 0.3169864
## 11         wilson 50 200 0.2500000 0.1950817 0.3143410
```

- Następnie należało wyznaczyć przedziały ufności, również dla leku ibuprofen, dla klientów do 35 lat. Odczytując z tabeli (1) mamy: $x = 0$, $n = 90$

```
x2 <- 0
n2 <- 90
data2 <- binom.confint(x2, n2)
df <- data.frame(data2)
df
```

```
##           method x  n        mean        lower      upper
## 1  agresti-coull 0 90 0.000000000 -0.008180285 0.04911591
## 2     asymptotic 0 90 0.000000000  0.000000000 0.00000000
## 3          bayes 0 90 0.005494505  0.000000000 0.02105727
## 4        cloglog 0 90 0.000000000  0.000000000 0.04015892
## 5          exact 0 90 0.000000000  0.000000000 0.04015892
## 6          logit 0 90 0.000000000  0.000000000 0.04015892
## 7         probit 0 90 0.000000000  0.000000000 0.04015892
## 8        profile 0 90 0.000000000  0.000000000 0.03652208
## 9            lrt 0 90 0.000000000  0.000000000 0.02111561
## 10     prop.test 0 90 0.000000000  0.000000000 0.05101162
## 11        wilson 0 90 0.000000000  0.000000000 0.04093563
```

- W kolejnym podpunkcie należało wyznaczyć przedziału ufności dla prawdopodobieństwa stosowania leku apap, dla wszystkich biorących udział w ankiecie. Odczytując z tabeli (1) mamy: $x = 44$, $n = 200$

```
x3 <- 44
n3 <- 200
data3 <- binom.confint(x3, n3)
df <- data.frame(data3)
df
```

```
##           method  x   n     mean     lower     upper
## 1  agresti-coull 44 200 0.220000 0.1679267 0.2826267
## 2     asymptotic 44 200 0.220000 0.1625894 0.2774106
## 3          bayes 44 200 0.221393 0.1651366 0.2792052
## 4        cloglog 44 200 0.220000 0.1654772 0.2795930
## 5          exact 44 200 0.220000 0.1646361 0.2838612
## 6          logit 44 200 0.220000 0.1679499 0.2827004
## 7         probit 44 200 0.220000 0.1670005 0.2815308
## 8        profile 44 200 0.220000 0.1663740 0.2807561
## 9            lrt 44 200 0.220000 0.1663832 0.2807552
## 10     prop.test 44 200 0.220000 0.1659406 0.2850661
## 11        wilson 44 200 0.220000 0.1681654 0.2823880
```

- W ostatnim podpunkcie należało wyznaczyć przedziały ufności, podobnie jak w podpunkcie powyżej, dla prawdopodobieństwa stosowania leku apap, ale dla podgrupy do 35 lat. Odczytując z tabeli (1) mamy: $x = 22$, $n = 90$

```
x4 <- 22
n4 <- 90
data4 <- binom.confint(x4, n4)
df <- data.frame(data4)
df
```

```
##           method  x  n      mean     lower     upper
## 1  agresti-coull 22 90 0.2444444 0.1667306 0.3430809
## 2     asymptotic 22 90 0.2444444 0.1556573 0.3332316
## 3          bayes 22 90 0.2472527 0.1612799 0.3363365
## 4        cloglog 22 90 0.2444444 0.1615228 0.3366897
## 5          exact 22 90 0.2444444 0.1599693 0.3463767
## 6          logit 22 90 0.2444444 0.1667000 0.3435007
## 7         probit 22 90 0.2444444 0.1648158 0.3411605
## 8        profile 22 90 0.2444444 0.1636309 0.3396167
## 9            lrt 22 90 0.2444444 0.1636231 0.3396152
## 10     prop.test 22 90 0.2444444 0.1626454 0.3484391
## 11        wilson 22 90 0.2444444 0.1673278 0.3424837
```

W tabeli (2) umieściłem wyniki dla testów, których użyłem w zadaniu 1 (3.1), a w tabelach (3) i (4) pozostałe przedziały ufności, które były dostępne w pakiecie `binom.confint`:

Przedziały ufności Agrestiego-Coulla				
Lek	Liczba sukcesów	Liczba prób	Przedział ufności	Długość przedziału
Ibuprofen	50	200	[0.1948993, 0.3145233]	0.119624
Ibuprofen (do lat 35)	0	90	[-0.0081803, 0.0491159]	0.0572962
Apap	44	200	[0.1679267, 0.2826267]	0.1147
Apap (do lat 35)	22	90	[0.1667306, 0.3430809]	0.1763503
Przedziały ufności Walda				
Ibuprofen	50	200	[0.1899886, 0.3100114]	0.1200228
Ibuprofen (do lat 35)	0	90	[0, 0]	0
Apap	44	200	[0.1625894, 0.2774106]	0.1148211
Apap (do lat 35)	22	90	[0.1556573, 0.3332316]	0.1775743
Przedziały ufności Cloppera-Pearsona				
Ibuprofen	50	200	[0.1916072, 0.3159628]	0.1243557
Ibuprofen (do lat 35)	0	90	[0, 0.0401589]	0.0401589
Apap	44	200	[0.1646361, 0.2838612]	0.1192252
Apap (do lat 35)	22	90	[0.1599693, 0.3463767]	0.1864074

Tabela 2. Uzyskane wyniki dla przedziałów wykorzystanych w zadaniu 1.

Przedziały ufności Bayesa				
Lek	Liczba sukcesów	Liczba prób	Przedział ufności	Długość przedziału
Ibuprofen	50	200	[0.1923105, 0.3115641]	0.1192536
Ibuprofen (do lat 35)	0	90	[0, 0.0210573]	0.0210573
Apap	44	200	[0.1651366, 0.2792052]	0.1140686
Apap (do lat 35)	22	90	[0.1612799, 0.3363365]	0.1750565
Przedziały ufności Cloglog				
Ibuprofen	50	200	[0.1923621, 0.3116476]	0.1192856
Ibuprofen (do lat 35)	0	90	[0, 0.0401589]	0.0401589
Apap	44	200	[0.1654772, 0.279593]	0.1141158
Apap (do lat 35)	22	90	[0.1615228, 0.3366897]	0.1751669
Przedziały ufności Logit				
Ibuprofen	50	200	[0.1948697, 0.3146322]	0.1197625
Ibuprofen (do lat 35)	0	90	[0, 0.0401589]	0.0401589
Apap	44	200	[0.1679499, 0.2827004]	0.1147504
Apap (do lat 35)	22	90	[0.1667, 0.3435007]	0.1768006
Przedziały ufności Probit				
Ibuprofen	50	200	[0.193976, 0.3136105]	0.1196346
Ibuprofen (do lat 35)	0	90	[0, 0.0401589]	0.0401589
Apap	44	200	[0.1670005, 0.2815308]	0.1135809
Apap (do lat 35)	22	90	[0.1648158, 0.3411605]	0.1763447

Tabela 3. Uzyskane wyniki

Przedziały ufności Profile				
Lek	Liczba sukcesów	Liczba prób	Przedział ufności	Długość przedziału
Ibuprofen	50	200	[0.1934176, 0.3129498]	0.1195322
Ibuprofen (do lat 35)	0	90	[0, 0.0365221]	0.0365221
Apap	44	200	[0.166374, 0.2807561]	0.1143821
Apap (do lat 35)	22	90	[0.1636309, 0.3396167]	0.1759858
Przedziały ufności Lrt				
Ibuprofen	50	200	[0.1934316, 0.3129489]	0.1195173
Ibuprofen (do lat 35)	0	90	[0, 0.0211156]	0.0211156
Apap	44	200	[0.1663832, 0.2807552]	0.114372
Apap (do lat 35)	22	90	[0.1636231, 0.3396152]	0.1759921
Przedziały ufności prop.test				
Ibuprofen	50	200	[0.1928239, 0.3169864]	0.1241625
Ibuprofen (do lat 35)	0	90	[0, 0.0510116]	0.0510116
Apap	44	200	[0.1659406, 0.2850661]	0.1191255
Apap (do lat 35)	22	90	[0.1626454, 0.3484391]	0.1857937
Przedziały ufności Wilsona				
Ibuprofen	50	200	[0.1950817, 0.314341]	0.1194416
Ibuprofen (do lat 35)	0	90	[0, 0.0409356]	0.0409356
Apap	44	200	[0.1681654, 0.282388]	0.1142226
Apap (do lat 35)	22	90	[0.1673278, 0.3424837]	0.1751559

Tabela 4. Uzyskane wyniki

»»»¿ d4ea8b7876c9f168be9eb784204a7e33196481cf

Po porównaniu danych przedziałów ufności wyznaczonych różnymi metodami, widać, że przedziały ufności Agrestiego-Coulla i Walda, gdy liczba sukcesów jest równa zero, wskazują niepokące wyniki. Pierwsza metoda wyznacza przedział ufności obejmujący wartości ujemne, zaś w drugim przypadku przedział ufności ma zerową długość.

Przypuszczam, że w przypadku zerowej liczby sukcesów, metody Agrestiego-Coulla i Walda nie działają poprawnie. Przedziały ufności uzyskane pozostałymi metodami mają w ogólności zbliżoną długość, co wynika ze stosunkowo duzych prób n. Przedziały ufności wyznaczone metodą Cloppera-Pearsona oraz prop.test mają największą długość. Najkrótsze długości przedziałów uzyskujemy stosując metodę Bayesa. Natomiast spośród trzech wyżej analizowanych metod, najlepszym wyborem wydaje się być test Agrestiego-Coulla, który ma najkrótsze przedziały ufności. Jednak nie sprawdza się on przy skrajnych małej liczbie sukcesów, dając ujemną dolną granicę prawdopodobieństwa p, co potwierdza wcześniejsze wnioski. Dla skrajnych przypadków, spośród trzech badanych metod, najlepiej działa metoda Cloppera-Pearsona.

3.3. Zadanie dodatkowe

W tym zadaniu należało wyznaczyć granice punktowo asymptotycznego przedziału ufności, dla prawdopodobieństwa sukcesu, bazując na przekształceniu *logit*, *probit* albo *cloglog*. Następnie dla wysymulowanych danych wyznaczyć realizację wyznaczonego przedziału i porównać z odpowiednią realizacją uzyskaną z funkcji *binom.confint*. Wybrałem metodę delta, bazującą na przedziale typu *logit*, gdzie funkcja g jest równa $g(p) = \ln \frac{p}{1-p}$.

Korzystając z wykładu, wiemy że w przypadku estymacji prawdopodobieństwa sukcesu p, estymator największej wiarogodności $\hat{p}$ jest postaci:

$$\hat{p}(\mathbf{X}) = \frac{1}{n}\sum_{i=1}^{n} X_i = \overline{X} \tag{13}$$

oraz

$$\sqrt{n}\frac{\overline{X} - p}{\sqrt{p(1-p)}} \tag{14}$$

dąży, gdy $n \to \infty$ według rozkładu do $\mathcal{N}(0, 1)$. Wiemy na tej podstawie, że:

$$\sqrt{n}(\hat{p}(\mathbf{X}) - p) \tag{15}$$

dąży do rozkładu $\mathcal{N}(0, p(1-p))$. Na tej podstawie, korzystając z wykładu, wiemy że:

$$\sqrt{n}[g(\hat{p}(\mathbf{X})) - g(p)] \tag{16}$$

dąży do rozkładu $\mathcal{N}(0, [g'(p)]^2 p(1-p))$, gdzie funkcja g, w typie *logit*, to $g(p) = \ln \frac{p}{1-p}$.

W następnym kroku, na podstawie funkcji centralnej asymptotycznie:

$$Q_n(\mathbf{X}, p) = \frac{\sqrt{n}[g(\hat{p}(\mathbf{X})) - g(p)]}{g'(p)\sqrt{p(1-p)}} \tag{17}$$

konstruujemy przedział ufności dla parametru $g(p)$:

$$z\left(\frac{\alpha}{2}\right) < \frac{\sqrt{n}[g(\hat{p}(\mathbf{X})) - g(p)]}{g'(p)\sqrt{p(1-p)}} < z\left(1 - \frac{\alpha}{2}\right) \tag{18}$$

Wyznaczamy przedział dla $g(p)$:

$$\begin{aligned} -\frac{z\left(\frac{\alpha}{2}\right)g'(p)\sqrt{p(1-p)}}{\sqrt{n}} + g(\hat{p}(\mathbf{X})) > \\ > g(p) > \\ > -\frac{z\left(1-\frac{\alpha}{2}\right)g'(p)\sqrt{p(1-p)}}{\sqrt{n}} + g(\hat{p}(\mathbf{X})) \end{aligned} \tag{19}$$

Teraz "odwracamy" aby uzyskać przedział ufności dla p. Dostajemy ostatecznie:

$$\begin{aligned} g^{-1}\left(-\frac{z\left(1-\frac{\alpha}{2}\right)g'(p)\sqrt{p(1-p)}}{\sqrt{n}} + g(\hat{p}(\mathbf{X}))\right) < \\ < p < \\ < g^{-1}\left(-\frac{z\left(\frac{\alpha}{2}\right)g'(p)\sqrt{p(1-p)}}{\sqrt{n}} + g(\hat{p}(\mathbf{X}))\right) \end{aligned} \tag{20}$$

Przed sprawdzeniem poprawności wyznaczonego przedziału, musiałem wyznaczyć $g'(p)$ oraz $g^{-1}(p)$:

$$g'(p) = \frac{1}{p - p^2} \tag{21}$$

$$g^{-1}(p) = \frac{\exp(x)}{\exp(x) + 1} \tag{22}$$

Sprawdźmy teraz poprawność naszych przekształceń, przeprowadzając symulacje. Jako prawdopodobieństwo sukcesu p przyjąłem $p = 0.3$, liczbę prób $n = 50$ oraz poziom istotności $\alpha = 0.05$. Na początku wprowadźmy funkcję g, funkcję liczącą pochodną funkcji g i funkcję liczącą funkcję odwrotną g.

```
g <- function(x){
log(x/(1-x))
}
g.derivative <- function(x){
1/(x-x^2)
}
g.inv <- function(x){
exp(x)/(exp(x) + 1)
}
```

Teraz wysymulujmy próbę i wyznaczmy przedział ufności, korzystając z funkcji `binom.confint`:

```
p <- 0.3
n <- 50
alfa <- 0.05
X <- rbinom(n, 1, p)

przedzial <- binom.confint(sum(X), n, methods="logit")
c(mean(przedzial$lower), mean(przedzial$upper))

## [1] 0.2062019 0.4601918
```

A korzystając z naszych obliczeń, uzyskałem następujące wyniki:

```
Tl <- g.inv(- (qnorm(1-alfa/2)*g.derivative(p)
                *sqrt(p*(1-p)))/(sqrt(n)) + g(mean(X)))

Tu <- g.inv(- (qnorm(alfa/2)*g.derivative(p)
                *sqrt(p*(1-p)))/(sqrt(n)) + g(mean(X)))
c(Tl, Tu)

## [1] 0.2044631 0.4628402
```

Porównując uzyskane wyniki wyznaczone z funkcji `binom.confint` i uzyskane z naszych przekształceń, możemy powiedzieć, że uzyskany przez nas przedział został wyznaczony poprawnie.