Statystyka dla przyrodników w R

Idzi Siatkowski

Joanna Zyprych-Walczak

21 kwietnia 2017

Spis treści

Rozdział 1

Wprowadzenie

1.1 Cel książki

Prezentowana książka przeznaczona jest dla wszystkich początkujących, nie znających R a chcących poznać podstawowe możliwości obliczeniowe i graficzne oprogramowania R w zakresie zastosowań statystyki. Zawiera przede wszystkim przykłady wraz z programami (ko-dami, skryptami) napisanymi w R. Teoria przedstawiona jest w dużym skrócie. Przykłady dotyczą przede wszystkim zagadnień przyrodniczych i zawierają informacje o książkach w których znajduje się właściwa teoria. Po wykonaniu przykładów czytelnik będzie potrafił sa-modzielnie rozwiązywać problemy statystyczne takie jak testowanie, regresja oraz wykony-wać wysokiej jakości rysunki związane ze statystyką.

1.2 Co to jest R

R jest narzędziem (programem, pakietem, środowiskiem) przeznaczonym m.in. do wykonywania prostych, złożonych oraz bardzo skomplikowanych obliczeń i analiz staty-stycznych, a także do tworzenia grafiki wysokiej jakości. Oznacza to, że możemy wykonywać proste obliczenia takie jak np. na kalkulatorze, możemy stosować zaawansowane metody sta-tystyczne, czy obliczenia symulacyjne oraz optymalizacyjne. Ponadto możemy w łatwy spo-sób tworzyć wykresy oraz innego rodzaju grafiki.

1.3 Zalety R

- Darmowy do wszelkich zastosowań (licencja GPL GNU)
- Możliwość korzystania z ok. 9859 pakietów (styczeń 2017)
- Możliwość tworzenia wysokiej jakości wykresów
- Wykonywanie funkcji z bibliotek napisanych w różnych językach programowania (Fortran, C, C++, S)
- Pozwala na tworzenie i używanie własnych programów
- R jest wykorzystywany w uczelniach, instytutach badawczych, bankach, małych i dużych firmach analizujących różne typy danych
- Działa w różnych systemach operacyjnych (np. Windows, Linux, Mac)
- R jest elastyczny, nie jest "czarną skrzynką" tzn. na każdym etapie dostępny jest kod wykonywanych poleceń

1.4 Instalacja R i RStudio

Instalacja R

W pierwszej kolejności należy skopiować na swój komputer plik instalacyjny R, np. plik "R-3.3.2-win.exe" ze strony internetowej

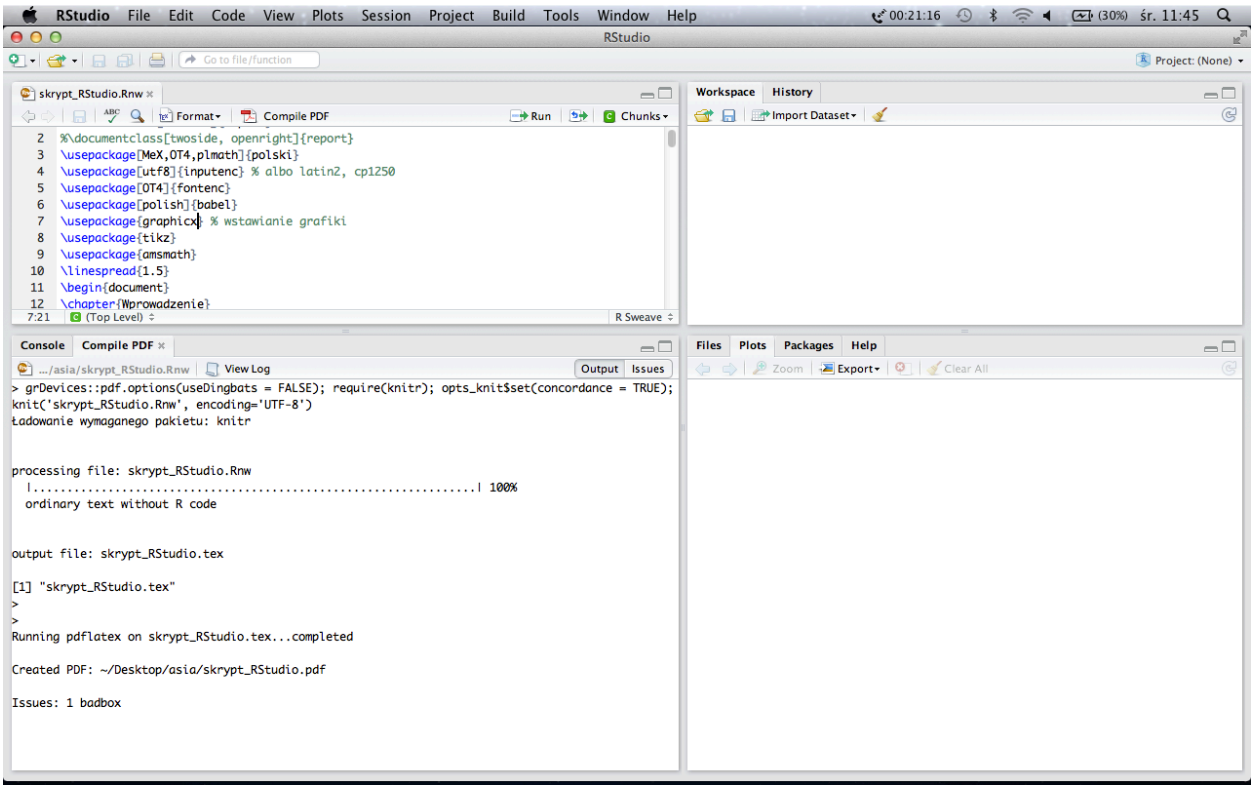
www.r-project.org

czyli:

1. uruchamiamy stronę internetową "www.r-project.org"
2. wybieramy "download R"
3. wybieramy np. "https://cloud.r-project.org/"
4. wybieramy "Download R for Windows" (działamy pod windows'em)
5. wybieramy "install R for the first time"
6. wybieramy "Download R 3.3.2 for Windows".
7. zapisujemy plik instalacyjny "R-3.3.2-win.exe" na swoim komputerze.

Następnie należy uruchomić skopiowany plik instalacyjny i postępować zgodnie w sugestiami.

Instalacja RStudio



Rysunek 1.1: Here is a nice figure!

Po instalacji R proponujemy zainstalować edytor (interfejs) RStudio dla łatwiejszego korzystania z R. Należy skopiować na swój komputer darmową wersję programu instalacyj-nego RStudio ze strony internetowej

www.rstudio.com

czyli np. plik “RStudio-1.0.136.exe”. Uruchamiając ten plik dokonujemy instalacji edytora RStudio. Po zainstalowaniu uruchamiamy RStudio i mamy ekran komputera np. tak jak na rysunku 1.

Interfejs RStudio składa się z czterech okienek. Lewe dolne okienko jest konsolą. Po znaku zachęty “>” możemy napisać polecenie (komendę, skrypt) i po naciśnięciu klawisza “enter” polecenie to zostanie wykonane, a wynik zostanie wyświetlony poniżej. Okienko lewe górne (okno edycji) służy do edycji skryptów, które można tworzyć, zmieniać, zapisywać oraz wykonywać klikając na polecenie “run”. Wyniki realizacji poleceń wyświetlane są w lewym dolnym okienku, czyli okienku konsoli. Okienko prawe górne jest okienkiem zawiera-jącym historię działania w RStudio oraz przedstawiającym informacje o wprowadzonych da-nych. Natomiast w prawym dolnym okienku znajdują się informacje o pakietach, plikach, wyświetlane są rysunki oraz pomoc.

Uwaga: Należy najpierw zainstalować R, a następnie RStudio. Uruchamiamy tylko RStudio.

1.5 Pakiety

Podczas instalacji R, instalowane są także bazowe pakiety obliczeniowe. W każdym momencie możemy zainstalować dowolny pakiet korzystając z prawego dolnego okienka RStudio. Należy w zakładce "Packages" uruchomić polecenie "install" i wpisać nazwę pakietu. Informacje dotyczące pakietów, czyli opisy oraz kody pakietów, można znaleźć uruchamiając kolejno:

1. www.r-project.org
2. CRAN
3. wybierając np. "https://cloud.r-project.org/"
4. wybierając "Packages"

Po zainstalowaniu pakietu, można z niego korzystać (czyli korzystać z poleceń w nim zawartych) dopiero po "uruchomieniu" pakietu poleceniem library(), gdzie w nawiasach wpisana jest nazwa pakietu.

1.6 Dokumentacja i szukanie pomocy

Materiały dotyczące R, dla początkujących, a także zaawansowanych użytkowników, wykorzystanie R w podstawowej oraz zaawansowanej statystyce, a także zastosowanie R w tworzeniu wykresów znajdują się przede wszystkim "w internecie", szczególnie na stronie "www.r-project.org". Są to artykuły, raporty oraz książki - także w języku polskim. Natomiast pomoc najłatwiej można uzyskać wpisując w okienku konsoli hasło poprzedzone zna-kiem zapytania lub wpisując polecenie help(), gdzie w nawiasach wpisana jest nazwa hasła. Treść pomocy wyświetlona zostanie w prawym dolnym okienku.

Rozdział 2

Obliczenia w R

Polecenia w R można realizować na kilka sposobów. Dwa najprostsze są następujące:

1. w lewym górnym oknie RStudio (okno edycji) piszemy polecenie (kod, skrypt) i następnie wykonujemy polecenie "Run" (kursor wskazuje, który wiersz poleceń będzie wykonany, natomiast zaznaczony obszar wskazuje które polecenia będą wykonane).
2. w lewym dolnym oknie RStudio (okno konsoli) po znaku zachęty (">") piszemy polecenie (kod, skrypt) i wykonujemy to polecenie naciskając klawisz "enter".

Uwagi:

1. Realizacja wykonanych poleceń przedstawiana jest w lewym dolnym oknie RStudio (okno konsoli).
2. Po znaku "#" występuje komentarz, który nie jest wykonywany,
3. Liczba rzeczywista przedstawiana jest za pomocą kropki, a nie przecinka.

2.1 Proste obliczenia matematyczne

Przykład 1. W lewym górnym oknie RStudio (okno edycji) piszemy:

```
6+8
```

i wykonujemy polecenie "Run". Wówczas w lewym dolnym oknie RStudio (okno konsoli) pojawi się:

```
> 6+8
```

```
[1] 14
```

gdzie znak ">" jest tzw. znakiem zachęty, "6+8" jest wykonanym poleceniem, "[1]" jest liczbą elementów wyjściowych, natomiast "14" jest wynikiem realizacji polecenie wejściowego.

Uwaga. W prezentowanym manuskrypcie wszystkie polecenia, kody oraz skrypty oznaczane będą czcionką koloru bordowego i nazwane "Kod w R:". Najlepiej polecenia takie umieścić w lewym górnym oknie RStudio (okno edycji). Natomiast wynik wykonania skryptu (po uruchomieniu poleceniem "Run"), przedstawiony będzie w lewym dolnym oknie RStudio (okno konsoli) i oznaczony w manuskrypcie kolorem niebieskim oraz nazwany "Realizacje w R:".

Kod w R

```
3+5 # dodawanie
4-6 # odejmowanie
8*7 # mnożenie
21/5 # dzielenie
5^3 # 5 do potęgi 3
sqrt(49) # pierwiastek kwadratowy z 49
49^(1/2) # pierwiastek kwadratowy z 49
(-8)^(1/3) # pierwiastek trzeciego stopnia z -8
log(7) # logarytm naturalny z 7
log10(6) # logarytm przy podstawie 10 z 6
log2(5) # logarytm przy podstawie 2 z 5
log(4, 5) # logarytm przy podstawie 5 z 4
exp(3) # e do potęgi 3
sin(6.28) # sinus kąta 6.28, gdzie 6.28 jest kątem w radia-nach, czyli kąta 360 stopni
cos(pi/2) # cosinus kąta pi/2, gdzie pi/2 jest kątem w radia-nach, czyli kąta 90 stopni
```

Realizacja w R

```
> 3+5 # dodawanie
```

```
[1] 8
```

Tablica 2.1: **Podstawowe funkcje i operatory w R**

Funkcje (operatory) w R	**Opis**
$+, -, *, /$	dodawanie, odejmowanie, mnożenie, dzielenie
sqrt(x),$x\hat{}y$	pierwiastkowanie, potęgowanie
$>, <,$	większe, mniejsze
$>=, <=$	większe lub równe, mniejsze lub równe
x%%y	reszta z dzielenia x przez y
log(x), log2(x), exp(x)	logarytm z x, eksponenta z x
round(x)	zaokrąglenie liczby x
abs(x)	wartość bezwzględna z x
choose(x)	symbol Newtona
factorial(x)	silnia z x
%in%	czy znajduje się w wektorze
%*%	iloczyn macierzowy
&	iloczyn logiczny
|	suma logiczna
!	zaprzeczenie
==	równość
! =	nierówność
all	sprawdza, czy wszystkie elementy spełniają warunek
any	sprawdza, czy którykolwiek z elementów spełnia warunek
which	zwraca indeksy obiektu spełniające warunek

```
> 4-6 # odejmowanie
```

```
[1] -2
```

```
> 8*7 # mnożenie
```

```
[1] 56
```

```
> 21/5 # dzielenie
```

```
[1] 4.2
```

```
> 5^3 # 5 do potęgi 3
```

```
[1] 125
```

```
> sqrt(49) # pierwiastek kwadratowy z 49
```

```
[1] 7
```

```
> 49^(1/2) # pierwiastek kwadratowy z 49
```

```
[1] 7
```

```
> (-8)^(1/3) # pierwiastek trzeciego stopnia z -8
```

```
[1] NaN
```

```
> log(7) # logarytm naturalny z 7
```

```
[1] 1.94591
```

```
> log10(6) # logarytm przy podstawie 10 z 6
```

```
[1] 0.7781513
```

```
> log2(5) # logarytm przy podstawie 2 z 5
```

```
[1] 2.321928
```

```
> log(4, 5) # logarytm przy podstawie 5 z 4
```

```
[1] 0.8613531
```

```
> exp(3) # e do potęgi 3
```

```
[1] 20.08554
```

```
> sin(6.28) # sinus kąta 6.28, gdzie 6.28 jest kątem w radia-nach, czyli kąta 360 stopni
```

```
[1] -0.003185302
```

```
> cos(pi/2) # cosinus kąta pi/2, gdzie pi/2 jest kątem w radia-nach, czyli kąta 90 stopni
```

```
[1] 6.123234e-17
```

Przykład 2

Kod w R

```
2+3; 1-2;4/2;4*3
```

Realizacja w R

```
> 2+3; 1-2;4/2;4*3
```

```
[1] 5
```

```
[1] -1
```

```
[1] 2
```

```
[1] 12
```

2.2 Zmienne

W R operatorem przypisania jest znak "=" lub "<-". W manuskrypcie stosujemy znak "=".

Kod w R: # zmienne

```
x=4  # przypisanie zmiennej x wartości 4
x  # wyświetlenie wartości zmiennej x, czyli 4
```

Realizacja w R

```
> x=4  # przypisanie zmiennej x wartości 4
> x  # wyświetlenie wartości zmiennej x, czyli 4
```

```
[1] 4
```

Kod w R:

```
imie = "Joasia"
imie
nazwisko="Nowak"
nazwisko
```

Realizacja w R

```
> imie = "Joasia"
> imie
```

```
[1] "Joasia"
```

```
> nazwisko="Nowak"
> nazwisko
```

```
[1] "Nowak"
```

2.3 Wektory, macierze oraz ramki danych

Podstawowa funkcja wykorzystywana w R w celu utworzenia wektora to c(). Przykładowo, gdy chcemy utworzyć wektor o nazwie a z elementami 3 i 1 piszemy: a=c(3,1). Wektor musi posiadać elementy tylko jednego typu. Rozróżniamy następujące wektory: wektor numerycz-ny, wektor znakowy oraz wektor logiczny.

WEKTORY

Kod w R:

```
a = c(3, 5, 7, 9, 11)  # wektor numeryczny
a
dni = c("wtorek", "czwartek", "sobota", "niedziela")  #  wektor znakowy
dni
c = c(TRUE,TRUE,TRUE,FALSE,TRUE,FALSE) # wektor logiczny
c
```

Realizacja w R

```
> a = c(3, 5, 7, 9, 11)  # wektor numeryczny
> a
```

```
[1]  3  5  7  9 11
```

```
> dni = c("wtorek", "czwartek", "sobota", "niedziela")  #  wektor znakowy
> dni
```

```
[1] "wtorek"    "czwartek"  "sobota"    "niedziela"
```

```
> c = c(TRUE,TRUE,TRUE,FALSE,TRUE,FALSE) # wektor logiczny
> c
```

```
[1]  TRUE  TRUE  TRUE FALSE  TRUE FALSE
```

Tablica 2.2: **Przykładowe funkcje tworzenia wektorów.**

Nazwa.funkcji	Przykład	Opis
:	1:3	tworzy sekwencje od : do
seq(from=x,to=y,by=z)	seq(from=0,to=8,by=2)	tworzy regularne sekwencje od 0 do 8 co 2
seq(from=x,to=y, length.out=z)	seq(from=0,to=10, length.out=3)	tworzy regularne sekwencje od 0 do 10 o 3 liczbach
rep(x),rep(x,y)	rep(3),rep(3,4)	x oznacza co ma być powtórzone, y ile razy
rep(x,length.out=y)	rep(1:2,length.out=4)	Powtórzona sekwencja liczb 1 i 2 o 4 liczbach
rep(x,each=y)	rep(3:1,each=2)	Każda cyfra z sekwencji 3:1 powtórzona 2 razy

Kod w R:

```
b=c( 3:14 )
b
```

Realizacja w R

```
> b=c( 3:14 )
> b
```

```
 [1]  3  4  5  6  7  8  9 10 11 12 13 14
```

Kod w R:

```
# działania na wektorach
a = c( 1, 3, 5 )
a
```

```
b = c( 8, 10 )
b
ab = c( a, b )
ab
```

Realizacja w R

```
> # działania na wektorach
> a = c( 1, 3, 5 )
> a
```

```
[1] 1 3 5
```

```
> b = c( 8, 10 )
> b
```

```
[1]  8 10
```

```
> ab = c( a, b )
> ab
```

```
[1]  1  3  5  8 10
```

Kod w R:

```
# dołączamy kolejne wartości
ab[6:10] <- c( 0, -6, -3, -1, -5)
ab
```

Realizacja w R

```
> # dołączamy kolejne wartości
> ab[6:10] <- c( 0, -6, -3, -1, -5)
> ab
```

```
 [1]  1  3  5  8 10  0 -6 -3 -1 -5
```

Podstawowe operacje na wektorach:

min(x), max(x), range(x) # minimum, maximum, rozstęp

sum(x), prod(x) # suma i iloczyn elementów

mean(x), median(x) # średnia arytmetyczna i mediana

var(x), sd (x) # wariancja i odchylenie standardowe

IQR(x) # zakres międzykwartylowy

sort(x) # posortowane elementy w kolejności rosnącej

summary(x) # statystyki: min, max, średnia, mediana, kwartyle

Przykład 1. (Kala 2005, s. 26)

Obserwowano plonowanie 30 krzaków pomidorów "New Yorker" i otrzymano następujące wielkości plonów (w kg): 1.52, 1.57, 1.30, 1.62, 1.55, 1.70, 2.05, 1.64, 1.95, 1.80, 1.76, 1.40, 1.92, 2.20, 1.57, 1.59, 1.27, 1.79, 1.29, 1.84, 1.77, 1.72, 1.53, 1.32, 1.69, 1.95, 1.75, 1.08, 1.70, 1.45.

Wyznaczyć wartość minimalną i maksymalną, rozstęp, sumę i iloczyn elementów, średnią arytmetyczną i medianę, wariancję i odchylenie standardowe. Następnie rosnąco posortować wszystkie elementy oraz wykonać polecenie "summary".

Kod w R:

```
# Przykład. R Kala: Statystyka dla przyrodników - 2005, s.26
# Plonowanie krzaków pomidorów odmiany "New Yorker"
# Przygotowanie danych
y = c(1.52, 1.57, 1.30, 1.62, 1.55, 1.70, 2.05, 1.64, 1.95, 1.80, 1.76, 1.40, 1.92, 2.20, 1.57, 1.59
# wyświetlanie zawartości y
y
# wykonanie obliczen
min(y)  # wartość minimalna
max(y)  # wartość maksymalna
range(y)  # rozstęp
sum(y)  # suma elementów
prod(y)  # iloczyn elementów
var(y)  # wariancja
sd(y)  # odchylenie standardowe
sort(y)
summary(y) # wartości wybranych statystyk
```

Realizacja w R

```
> # Przykład. R Kala: Statystyka dla przyrodników - 2005, s.26
> # Plonowanie krzaków pomidorów odmiany "New Yorker"
> # Przygotowanie danych
> y = c(1.52, 1.57, 1.30, 1.62, 1.55, 1.70, 2.05, 1.64, 1.95, 1.80, 1.76, 1.40, 1.92, 2.20, 1.5
> # wyświetlanie zawartości y
> y

 [1] 1.52 1.57 1.30 1.62 1.55 1.70 2.05 1.64 1.95 1.80 1.76 1.40 1.92 2.20
[15] 1.57 1.59 1.27 1.79 1.29 1.84 1.77 1.72 1.53 1.32 1.69 1.95 1.75 1.08
[29] 1.70 1.45
> # wykonanie obliczen
> min(y)  # wartość minimalna

[1] 1.08
> max(y)  # wartość maksymalna

[1] 2.2
> range(y)  # rozstęp

[1] 1.08 2.20
> sum(y)  # suma elementów

[1] 49.29
> prod(y)  # iloczyn elementów

[1] 2068140
> var(y)  # wariancja

[1] 0.06315966
> sd(y)  # odchylenie standardowe

[1] 0.2513158
```

```
> sort(y)
```

```
 [1] 1.08 1.27 1.29 1.30 1.32 1.40 1.45 1.52 1.53 1.55 1.57 1.57 1.59 1.62
[15] 1.64 1.69 1.70 1.70 1.72 1.75 1.76 1.77 1.79 1.80 1.84 1.92 1.95 1.95
[29] 2.05 2.20
```

```
> summary(y) # wartości wybranych statystyk
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  1.080   1.523   1.665   1.643   1.785   2.200
```

MACIERZE

Macierz - zbiór elementów tego samego typu o strukturze wierszy i kolumn

Przykład macierzy o 3 wierszach i 5 kolumnach

2 3 7 5 1

7 9 1 4 0

8 2 6 3 7

Funkcją tworzącą macierz jest np.:

```
matrix(data, nrow, ncol, byrow)
```

gdzie:

`data` - dane, które chcemy przedstawić w formie macierzy,

`nrow` - liczba wierszy,

`ncol` - liczba kolumn,

`byrow` - jeśli byrow=TRUE, to macierz tworzona jest wierszami (domyślnie byrow=FALSE)

Kod w R:

```
mat1 = matrix(c(1,3,5,7,9,11,13,15,18,21,23,25),  nrow = 3)
mat1
```

Realizacja w R

```
> mat1 = matrix(c(1,3,5,7,9,11,13,15,18,21,23,25),  nrow = 3)
> mat1
```

```
     [,1] [,2] [,3] [,4]
[1,]    1    7   13   21
[2,]    3    9   15   23
[3,]    5   11   18   25
```

Kod w R:

```
mat2 = matrix(c(1,3,5,7,9,11,13,15,18,21,23,25),  ncol = 3)
mat2
```

Realizacja w R

```
> mat2 = matrix(c(1,3,5,7,9,11,13,15,18,21,23,25),  ncol = 3)
> mat2
     [,1] [,2] [,3]
[1,]    1    9   18
[2,]    3   11   21
[3,]    5   13   23
[4,]    7   15   25
```

Kod w R:

```
macierz1 = matrix(seq(1:24), nrow = 8)
macierz1
```

Realizacja w R

```
> macierz1 = matrix(seq(1:24), nrow = 8)
> macierz1
     [,1] [,2] [,3]
[1,]    1    9   17
[2,]    2   10   18
[3,]    3   11   19
[4,]    4   12   20
[5,]    5   13   21
[6,]    6   14   22
[7,]    7   15   23
```

```
[8,]    8   16   24
```

Kod w R:

```
macierz2 = matrix(seq(1:24),  nrow = 8, byrow=TRUE)
macierz2
```

Realizacja w R

```
> macierz2 = matrix(seq(1:24),  nrow = 8, byrow=TRUE)
> macierz2
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9
[4,]   10   11   12
[5,]   13   14   15
[6,]   16   17   18
[7,]   19   20   21
[8,]   22   23   24
```

Kod w R:

```
macdiag1=diag(4)
macdiag1
```

Realizacja w R

```
> macdiag1=diag(4)
> macdiag1
     [,1] [,2] [,3] [,4]
[1,]    1    0    0    0
[2,]    0    1    0    0
[3,]    0    0    1    0
[4,]    0    0    0    1
```

Kod w R:

```
dim(macierz1)
```

```
dim(macierz2)
dim(macdiag1)
```

Realizacja w R

```
> dim(macierz1)
```

```
[1] 8 3
```

```
> dim(macierz2)
```

```
[1] 8 3
```

```
> dim(macdiag1)
```

```
[1] 4 4
```

Kod w R:

```
t(macierz1)
t(macierz2)
t(macdiag1)
```

Realizacja w R

```
> t(macierz1)
```

```
     [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
[1,]    1    2    3    4    5    6    7    8
[2,]    9   10   11   12   13   14   15   16
[3,]   17   18   19   20   21   22   23   24
```

```
> t(macierz2)
```

```
     [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
[1,]    1    4    7   10   13   16   19   22
[2,]    2    5    8   11   14   17   20   23
[3,]    3    6    9   12   15   18   21   24
```

```
> t(macdiag1)
```

```
     [,1] [,2] [,3] [,4]
[1,]    1    0    0    0
```

```
[2,]    0    1    0    0
[3,]    0    0    1    0
[4,]    0    0    0    1
```

Odwoływanie się do elementów macierzy:

a) A[2, 3] # element z drugiego wiersza i trzeciej kolumny macierzy A

b) A[2,] # drugi wiersz macierzy A

c) A[, 3] # trzecia kolumna macierzy A

d) A[, c(1,3)] # pierwsza i trzecia kolumna macierzy A

Kod w R:

```
# macierze
dane1 = matrix(seq(1:24),  nrow = 8)
dane1
dane1[7, 2]  # element z siódmego wiersza i drugiej kolumny macierzy dane1
dane1[5,]  # piąty wiersz macierzy dane1
dane1[,3]  # trzecia kolumna macierzy dane1
dane1[ , c(1,3)]  # pierwsza i trzecia kolumna macierzy dane1
dane1[c(4,6),]  # czwarty i szósty wiersz macierzy dane1
```

Realizacja w R

```
> dane1 = matrix(seq(1:24),  nrow = 8)
> dane1
```

```
     [,1] [,2] [,3]
[1,]    1    9   17
[2,]    2   10   18
[3,]    3   11   19
[4,]    4   12   20
[5,]    5   13   21
[6,]    6   14   22
[7,]    7   15   23
[8,]    8   16   24
```

```
> dane1[7, 2]  # element z siódmego wiersza i drugiej kolumny macierzy dane1
```

```
[1] 15
```

```
> dane1[5,]  # piąty wiersz macierzy dane1
```

```
[1]  5 13 21
```

```
> dane1[,3]  # trzecia kolumna macierzy dane1
```

```
[1] 17 18 19 20 21 22 23 24
```

```
> dane1[ , c(1,3)]  # pierwsza i trzecia kolumna macierzy dane1
```

```
     [,1] [,2]
[1,]    1   17
[2,]    2   18
[3,]    3   19
[4,]    4   20
[5,]    5   21
[6,]    6   22
[7,]    7   23
[8,]    8   24
```

```
> dane1[c(4,6),]  # czwarty i szósty wiersz macierzy dane1
```

```
     [,1] [,2] [,3]
[1,]    4   12   20
[2,]    6   14   22
```

Kod w R:

```
# mnożenie macierzy
A=matrix(c(1,2,3,4,5,6), nrow=2)
A
B=matrix(c(9,8,7,6,5,4,3,2,1), nrow=3)
B
C=A%*%B  # mnozenie macierzy A i B
C
```

Realizacja w R

```
> # mnożenie macierzy
> A=matrix(c(1,2,3,4,5,6), nrow=2)
> A
```

```
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
```

```
> B=matrix(c(9,8,7,6,5,4,3,2,1), nrow=3)
> B
```

```
     [,1] [,2] [,3]
[1,]    9    6    3
[2,]    8    5    2
[3,]    7    4    1
```

```
> C=A%*%B  # mnozenie macierzy A i B
> C
```

```
     [,1] [,2] [,3]
[1,]   68   41   14
[2,]   92   56   20
```

Kod w R:

```
# Operacje na macierzach
rowSums(B) # sumy dla wierszy macierzy B z poprzedniego przykładu
rowMeans(B) # średnie arytmetyczne dla wierszy macierzy B
colSums(A) # sumy dla kolumn macierzy A
colMeans(A) # średnie arytmetyczne dla kolumn macierzy A
```

Realizacja w R

```
> # Operacje na macierzach
> rowSums(B) # sumy dla wierszy macierzy B z poprzedniego przykładu
```

```
[1] 18 15 12
```

```
> rowMeans(B) # średnie arytmetyczne dla wierszy macierzy B
```

```
[1] 6 5 4
```

```
> colSums(A) # sumy dla kolumn macierzy A
```

```
[1]  3  7 11
```

```
> colMeans(A) # średnie arytmetyczne dla kolumn macierzy A
```

```
[1] 1.5 3.5 5.5
```

Kod w R:

```
# Operacje na macierzach
D=matrix(c(1,3,5,1,2,3,7,8,1), nrow=3)  # tworzenie macierzy D
D
wyznacznik=det(D)  # wyznacznik macierzy D
wyznacznik
D1=solve(D)  # macierz odwrotna do macierzy D
D1
```

Realizacja w R

```
> # Operacje na macierzach
> D=matrix(c(1,3,5,1,2,3,7,8,1), nrow=3)  # tworzenie macierzy D
> D
```

```
     [,1] [,2] [,3]
[1,]    1    1    7
[2,]    3    2    8
[3,]    5    3    1
```

```
> wyznacznik=det(D)  # wyznacznik macierzy D
> wyznacznik
```

```
[1] 8
```

```
> D1=solve(D)  # macierz odwrotna do macierzy D
> D1
```

```
       [,1]  [,2]   [,3]
[1,] -2.750  2.50 -0.750
[2,]  4.625 -4.25  1.625
[3,] -0.125  0.25 -0.125
```

Kod w R:

```
# Rozwiązywanie układu równań
w=c(3,1,5)  # wektor wyrazów wolnych
w
roz=solve(D,w)  # rozwiązanie układu równań postaci Dx=w
roz
```

Realizacja w R

```
> # Rozwiązywanie układu równań
> w=c(3,1,5)  # wektor wyrazów wolnych
> w
```

```
[1] 3 1 5
```

```
> roz=solve(D,w)  # rozwiązanie układu równań postaci Dx=w
> roz
```

```
[1] -9.50 17.75 -0.75
```

RAMKA DANYCH

Ramka danych - zbiór elementów o strukturze wierszy i kolumn, gdzie kolumny mogą być różnego typu

Kod w R:

```
# ramka danych
dawki=c("d0", "d20", "d50", "d100")
odmiany=c("K", "M", "P", "S")
plon=c(6.1, 5.4, 6.5, 6.3)
roslina=data.frame(Odmiany=odmiany, Dawki=dawki, Plon=plon)
roslina
```

Realizacja w R

```
> # ramka danych
> dawki=c("d0", "d20", "d50", "d100")
> odmiany=c("K", "M", "P", "S")
> plon=c(6.1, 5.4, 6.5, 6.3)
> roslina=data.frame(Odmiany=odmiany, Dawki=dawki, Plon=plon)
> roslina
```

```
  Odmiany Dawki Plon
1       K    d0  6.1
2       M   d20  5.4
3       P   d50  6.5
4       S  d100  6.3
```

Rozdział 3

Przygotowanie danych

W R dane można przygotować na wiele sposobów. Przy czym pisząc przygotowanie danych mamy na myśli następującą sytuację: R "potrafi" przeczytać, otworzyć i wyświetlić dane na ekranie. Najprostszą metodą "tworzenia danych", omówioną w poprzednim rozdziale, jest zastosowanie polecenia "c()", czyli utworzenie wektora elementów wskazanych w "()". Jeśli np. mamy liczby 3, 5, 2 oraz 8 i wykonamy polecenie x=c(3, 5, 2, 8), to zmienna x bę-dzie wektorem powyższych liczb. Natomiast, jeśli mamy dane zapisane na dysku w formie pliku tekstowego lub pliku utworzonego w excelu, to należy zastosować odpowiednie pole-cenia do czytania takiego pliku.

Przykład 1. (Greń 1975, s.161)

Wylosowano po 12 pędów żyta trzech różnych gatunków i otrzymano dla nich następujące

długości kłosów żyta (w cm) - patrz Tablica 3.1:

Tablica 3.1: Dane przykład 1

Gatunek		
A	B	C
6,7	7,5	5,9
7,3	7,7	6,9
8,0	7,7	7,0
8,0	8,2	7,0
7,9	8,9	9,5
9,2	8,9	9,6
10,1	10,6	9,6
9,2	10,2	10,3
8,3	9,4	8,1
8,4	9,4	8,5
8,0	8,2	8,6
7,9	7,8	8,8

Przygotować dane w formie: a) ramki danych, b) pliku tekstowego, c) pliku excelowskiego.

Sposób 1 - ramka danych

Kod w R:

```
# Sposób 1 - ramka danych
A = c(6.7,7.3,8.0,8.0,7.9,9.2,10.1,9.2,8.3,8.4,8.0,7.9)
B = c(7.5,7.7,7.7,8.2,8.9,8.9,10.6,10.2,9.4,9.4,8.2,7.8)
C = c(5.9,6.9,7.0,7.0,9.5,9.6,9.6,10.3,8.1,8.5,8.6,8.8)
dane=data.frame(A, B, C)  # tworzenie ramki danych o nazwie "dane"
dane
```

Realizacja w R

```
> # Sposób 1 - ramka danych
> A = c(6.7,7.3,8.0,8.0,7.9,9.2,10.1,9.2,8.3,8.4,8.0,7.9)
> B = c(7.5,7.7,7.7,8.2,8.9,8.9,10.6,10.2,9.4,9.4,8.2,7.8)
> C = c(5.9,6.9,7.0,7.0,9.5,9.6,9.6,10.3,8.1,8.5,8.6,8.8)
```

```
> dane=data.frame(A, B, C)  # tworzenie ramki danych o nazwie "dane"
> dane
```

```
      A    B    C
1   6.7  7.5  5.9
2   7.3  7.7  6.9
3   8.0  7.7  7.0
4   8.0  8.2  7.0
5   7.9  8.9  9.5
6   9.2  8.9  9.6
7  10.1 10.6  9.6
8   9.2 10.2 10.3
9   8.3  9.4  8.1
10  8.4  9.4  8.5
11  8.0  8.2  8.6
12  7.9  7.8  8.8
```

Sposób 2 - plik tekstowy

W folderze “D://abc” pod nazwą “kwiaty.txt” zapisujemy plik tekstowy postaci:

Tablica 3.2: Dane przykład 1

A	B	C
6.7	7.5	5.9
7.3	7.7	6.9
8.0	7.7	7.0
8.0	8.2	7.0
7.9	8.9	9.5
9.2	8.9	9.6
10.1	10.6	9.6
9.2	10.2	10.3
8.3	9.4	8.1
8.4	9.4	8.5
8.0	8.2	8.6
7.9	7.8	8.8

Następnie wykonujemy polecenia czytania pliku tekstowego, podstawienia przeczytanych wartości pod nazwę "dane1" (druga linia poniższego kodu) oraz wyświetlenie zawartości zmiennej "dane1" (trzecia linia kodu).

Kod w R:

```
# czytanie pliku tekstowego
dane1 = read.table("~/Desktop/kwiaty.txt", header=TRUE)
dane1
```

Realizacja w R

```
> # czytanie pliku tekstowego
> dane1 = read.table("~/Desktop/kwiaty.txt", header=TRUE)
> dane1
```

```
   A   B   C
1  6.7 7.5 5.9
2  7.3 7.7 6.9
3  8.0 7.7 7.0
```

```
4   8.0  8.2  7.0
5   7.9  8.9  9.5
6   9.2  8.9  9.6
7  10.1 10.6  9.6
8   9.2 10.2 10.3
9   8.3  9.4  8.1
10  8.4  9.4  8.5
11  8.0  8.2  8.6
12  7.9  7.8  8.8
```

Sposób 3 - plik excelowski

W folderze "D://abc" pod nazwami "kwiaty.xlsx" oraz "kwiaty.xls" zapisujemy plik z treścią jak plik tekstowy "kwiaty.txt". Następnie wykonujemy poniższe polecenia.

Kod w R:

```
# czytanie pliku typu xlsx
library(readxl)  # otwarcie pakietu "openxlsx"
dane2 <- read_excel("~/Desktop/kwiaty.xlsx", sheet = 1)
dane2
```

Realizacja w R

```
> # czytanie pliku typu xlsx
> library(readxl)  # otwarcie pakietu "openxlsx"
> dane2 <- read_excel("~/Desktop/kwiaty.xlsx", sheet = 1)
> dane2
```

```
      A    B    C
1   6.7  7.5  5.9
2   7.3  7.7  6.9
3   8.0  7.7  7.0
4   8.0  8.2  7.0
5   7.9  8.9  9.5
6   9.2  8.9  9.6
7  10.1 10.6  9.6
```

```
8   9.2 10.2 10.3
9   8.3  9.4  8.1
10  8.4  9.4  8.5
11  8.0  8.2  8.6
12  7.9  7.8  8.8
```

Przydatne funkcje:

`rm(list=ls())` - usuwanie wszystkich obiektów z pamięci

`setwd("D://abc")` - ustanowienie aktualnej ścieżki dostępu do folderu "abc" znajdującego się na dysku D. Oznacza to, że jeśli zastosujemy tą funkcje, to zamiast funkcji

```
read.table("D://abc/kwiaty.txt", header=TRUE)
```

możemy wykorzystać funkcję postaci

```
read.table("kwiaty.txt", header=TRUE)
```

Rozdział 4

Wizualizacje

W rozdziale tym przedstawione zostaną podstawowe informacje dotyczące graficznych pre-zentacji danych oraz wykresów dla przykładowych funkcji.

4.1 Graficzna prezentacja danych

Przykład 1. (Kala 2005, s. 26)

Obserwowano plonowanie 30 krzaków pomidorów "New Yorker" i otrzymano następujące wielkości plonów (w kg): 1.52, 1.57, 1.30, 1.62, 1.55, 1.70, 2.05, 1.64, 1.95, 1.80, 1.76, 1.40, 1.92, 2.20, 1.57, 1.59, 1.27, 1.79, 1.29, 1.84, 1.77, 1.72, 1.53, 1.32, 1.69, 1.95, 1.75, 1.08, 1.70, 1.45.

Wykonać polecenie "summary" oraz przedstawić graficznie dane w postaci: barplot, plot, histogram oraz boxplot.

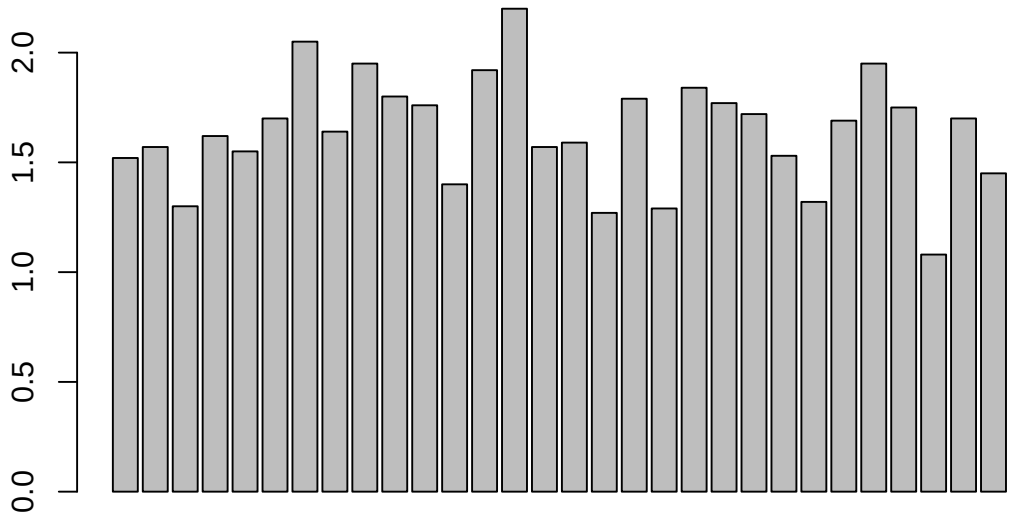
Kod w R:

```
# R Kala: Statystyka dla przyrodnikow - 2005, s.26
#  Plonowanie krzakow pomidorow odmiany "New Yorker"
y=c(1.52,1.57,1.30,1.62,1.55,1.70,2.05,1.64,1.95,1.80,1.76,1.40,1.92,2.20,1.57,
1.59,1.27,1.79,1.29,1.84,1.77,1.72,1.53,1.32,1.69,1.95,1.75,1.08,1.70,1.45)
summary(y)  # wyznaczenie wybranych statystyk punktowych
barplot(y)  # rys. 1
plot(y)  # rys. 2
plot(y,xlab="numery krzakow",ylab="wartosci y w kg", main="Wielkosci
```

```
     plonów pomidorow") # rys. 3
hist(y)  # rys. 4
hist(y, main="Plonowanie pomidorow")  # rys. 5
hist(y, col=rainbow(20), xlab="przedzialy", ylab="liczebnosci",
     main="Plonowanie pomidorow") # rys. 6
boxplot(y) # rys. 7
```

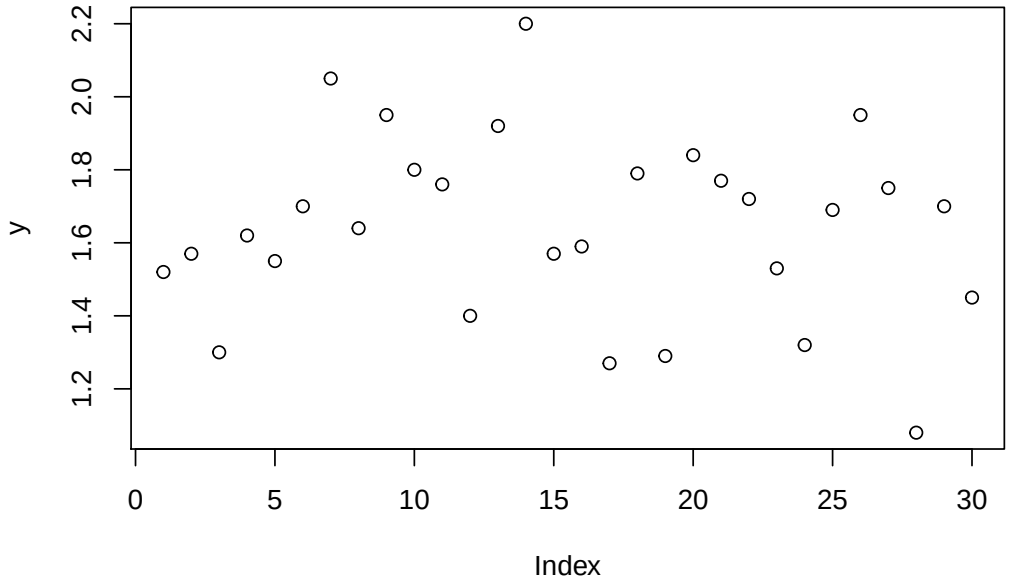
Realizacja w R

```
barplot(y)  # rys. 1
```



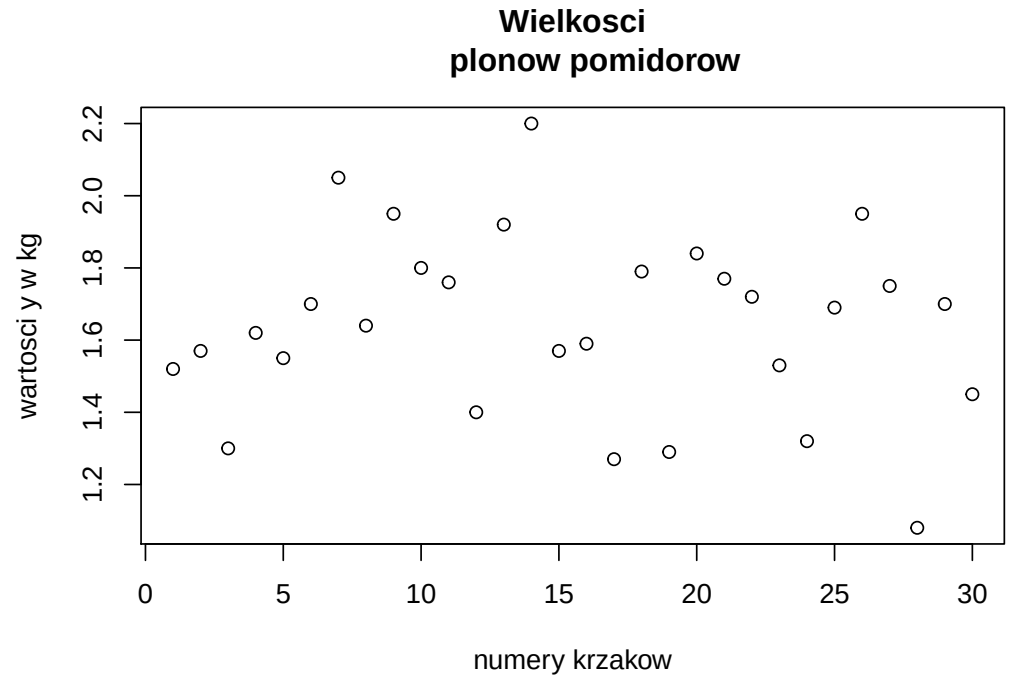
Rysunek 4.1: Barplot

```
plot(y)  # rys. 2
```



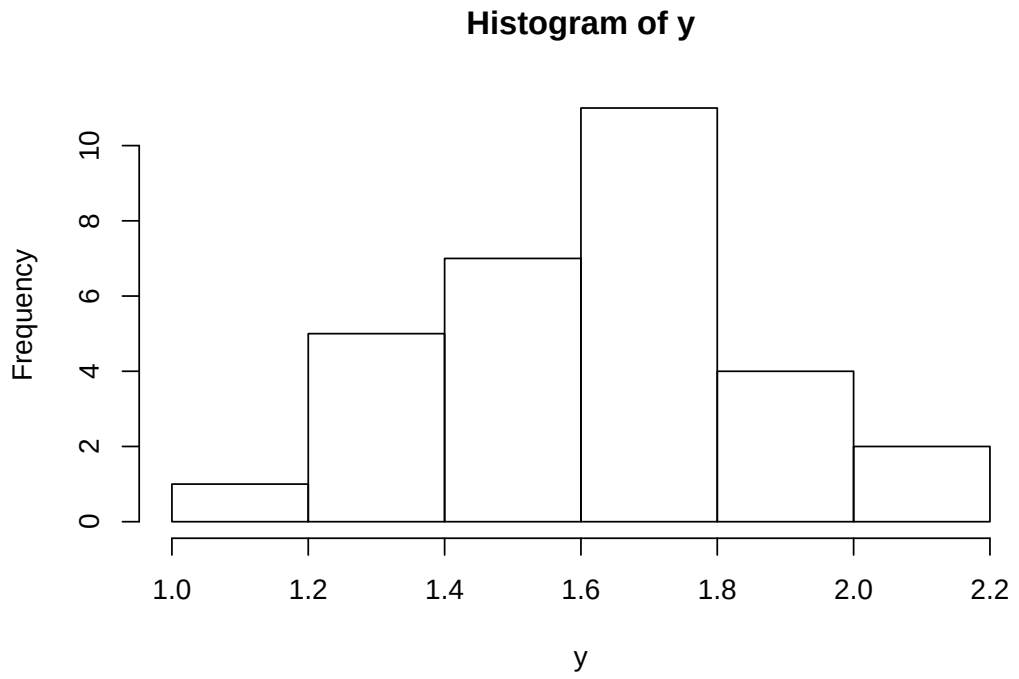
Rysunek 4.2: plot1

```
plot(y,xlab="numery krzakow",ylab="wartosci y w kg", main="Wielkosci
     plonow pomidorow") # rys. 3
```



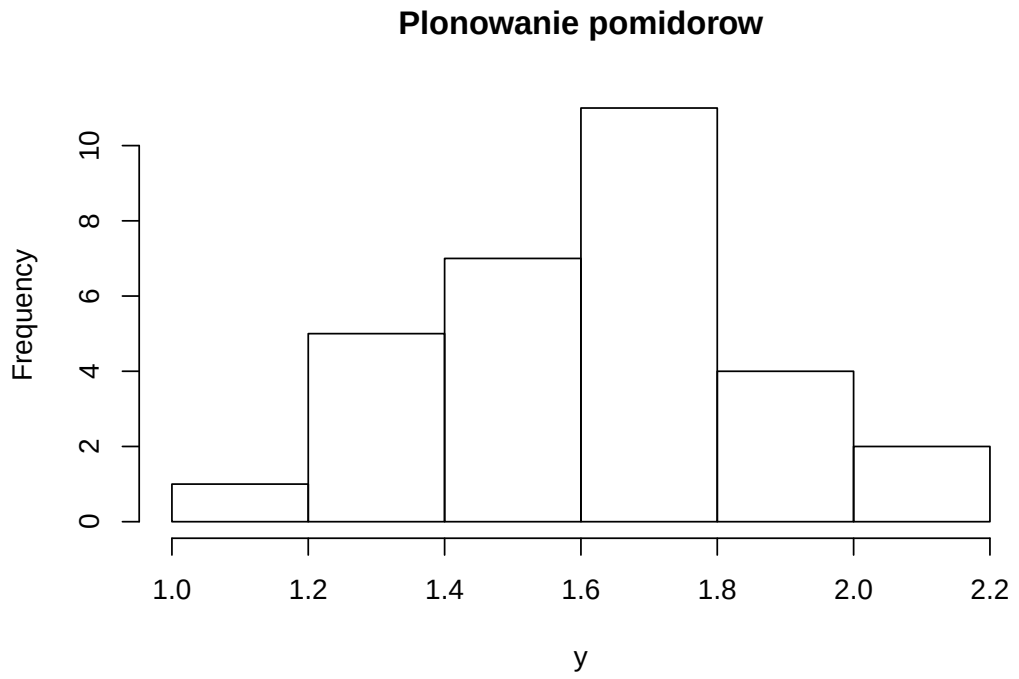
Rysunek 4.3: plot2

```r
hist(y)  # rys. 4
```



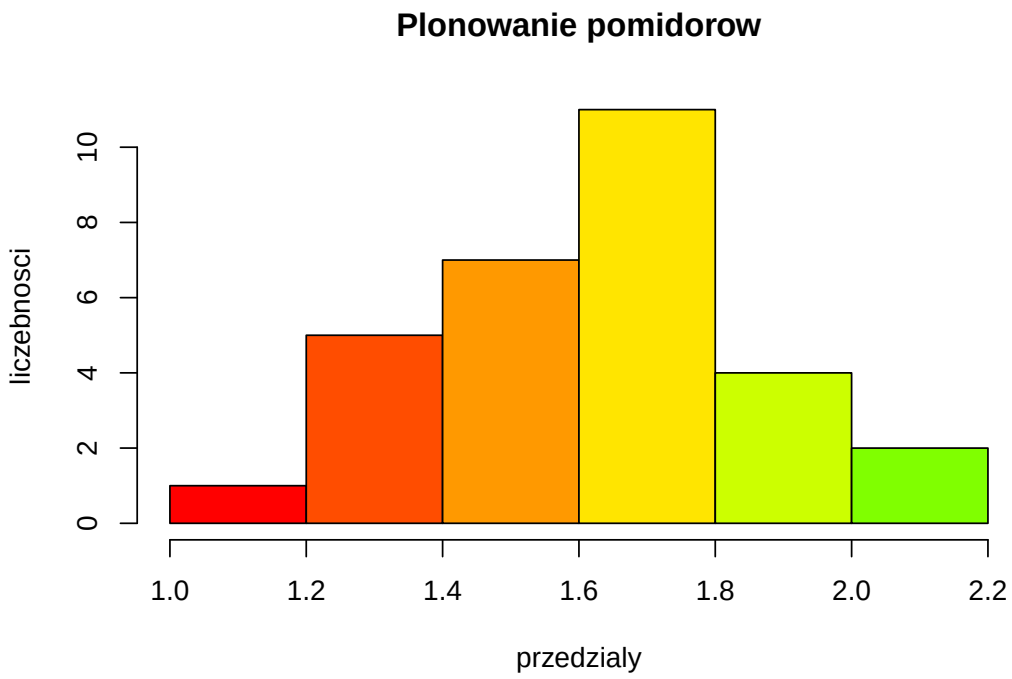
Rysunek 4.4: plot3

```r
hist(y, main="Plonowanie pomidorow")  # rys. 5
```



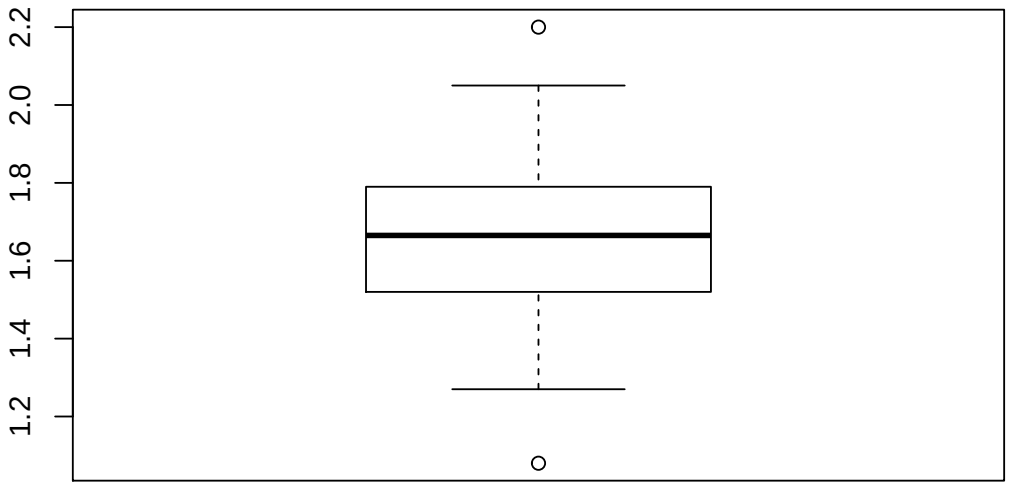
Rysunek 4.5: plot4

```
hist(y, col=rainbow(20), xlab="przedzialy", ylab="liczebnosci",
     main="Plonowanie pomidorow") # rys. 6
```



Rysunek 4.6: plot5

```
boxplot(y) # rys. 7
```



Rysunek 4.7: plot6

Przykład 2. (Greń 1975, s.161) – patrz s. xxx

Dla danych z przykładu xxx wykonać wykres typu boxplot.

Kod w R:

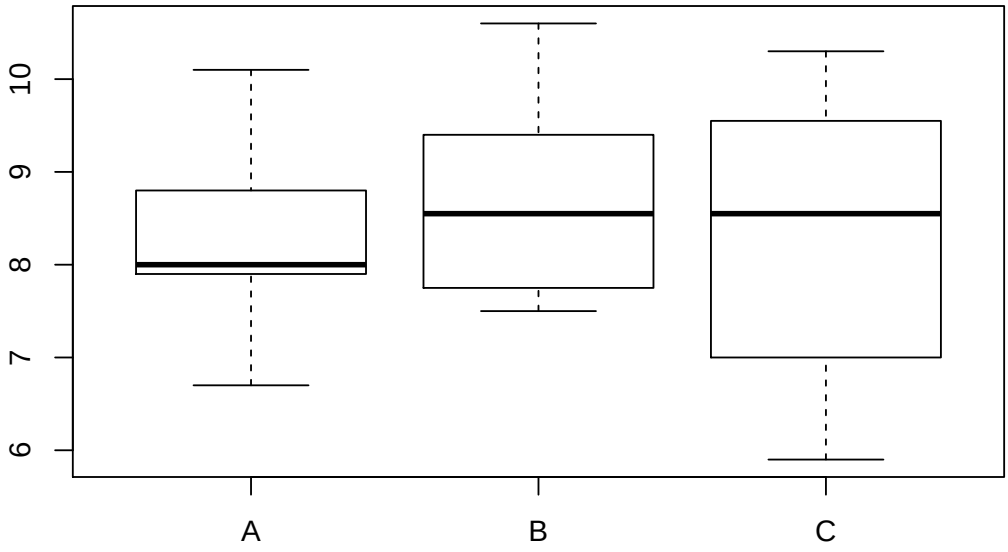
```
# przygotowanie danych
A = c(6.7,7.3,8.0,8.0,7.9,9.2,10.1,9.2,8.3,8.4,8.0,7.9)
B = c(7.5,7.7,7.7,8.2,8.9,8.9,10.6,10.2,9.4,9.4,8.2,7.8)
C = c(5.9,6.9,7.0,7.0,9.5,9.6,9.6,10.3,8.1,8.5,8.6,8.8)
dane=data.frame(A, B, C)  # tworzenie ramki danych o nazwie "dane"
dane
# wykresy typu boxplot
boxplot(dane)
boxplot(dane, main="ABC")
```

Realizacja w R

```
> # przygotowanie danych
> A = c(6.7,7.3,8.0,8.0,7.9,9.2,10.1,9.2,8.3,8.4,8.0,7.9)
> B = c(7.5,7.7,7.7,8.2,8.9,8.9,10.6,10.2,9.4,9.4,8.2,7.8)
> C = c(5.9,6.9,7.0,7.0,9.5,9.6,9.6,10.3,8.1,8.5,8.6,8.8)
> dane=data.frame(A, B, C)  # tworzenie ramki danych o nazwie "dane"
> dane
```

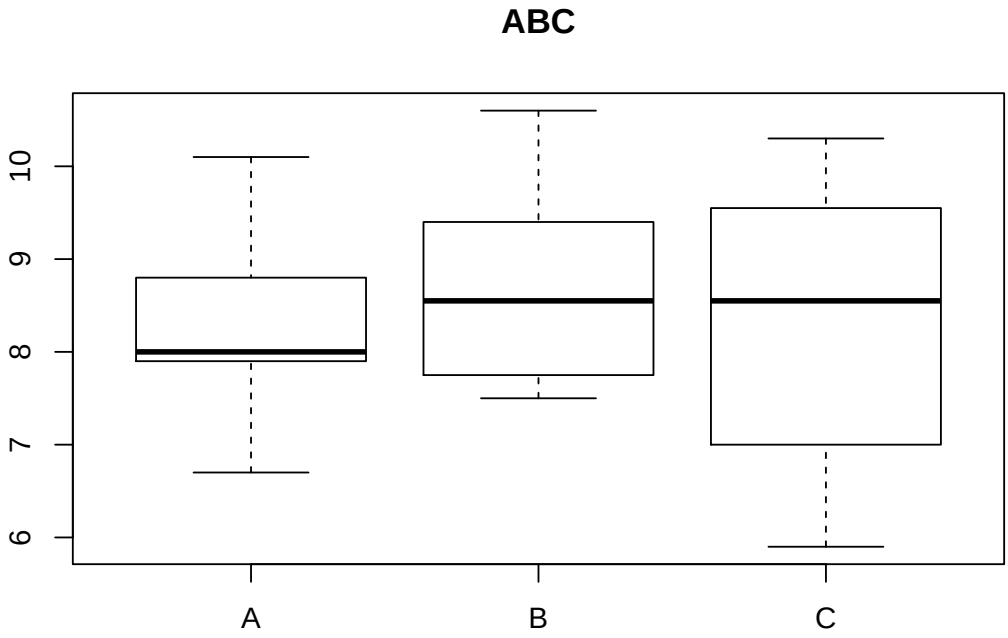
```
      A    B    C
1   6.7  7.5  5.9
2   7.3  7.7  6.9
3   8.0  7.7  7.0
4   8.0  8.2  7.0
5   7.9  8.9  9.5
6   9.2  8.9  9.6
7  10.1 10.6  9.6
8   9.2 10.2 10.3
9   8.3  9.4  8.1
10  8.4  9.4  8.5
11  8.0  8.2  8.6
12  7.9  7.8  8.8
```

```
boxplot(dane) # rys.
```



Rysunek 4.8: plot7

```
boxplot(dane,main="ABC") # rys.
```



Rysunek 4.9: plot8

4.2 Wykresy dla przykładowych funkcji

Przykład 3 Narysować wykres funkcji $y = x^3$ dla dla $x < -10; 10 >$ z osiami współrzędnych. **Kod w R:**

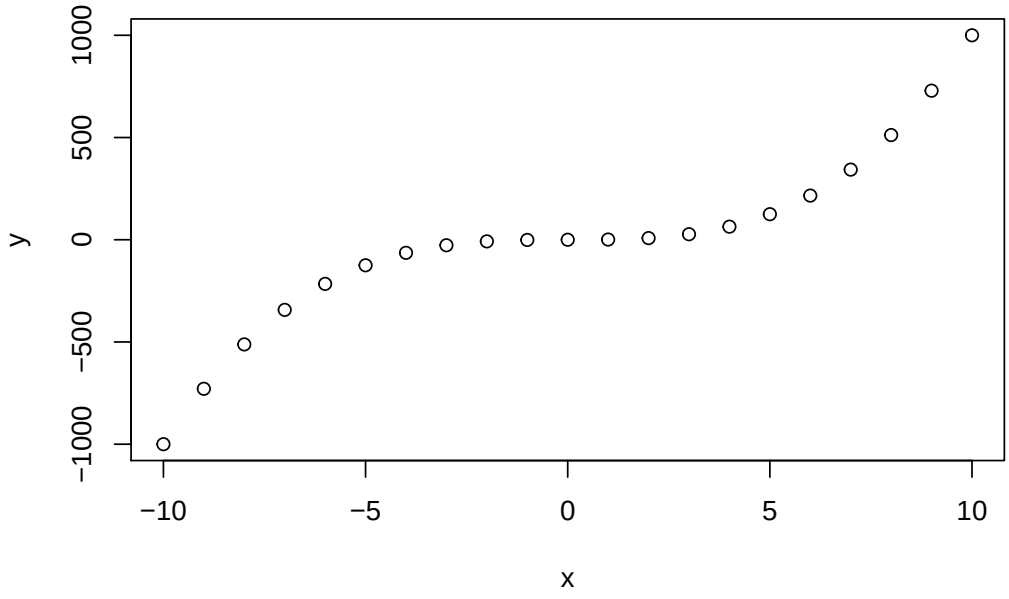
```
x = -10:10  # ustalenie wartości x
x
y=x^3  # ustalenie funkcji y
plot(x, y)  # Plot x i y
lines(x,y)  # Dodanie linii łączących x i y.
abline(h=0)  # Dodanie linii poziomej y=0
abline(v=0, col="red") # Dodanie czerwonej linii pionowej x=0
```

Realizacja w R

```
> x = -10:10  # ustalenie wartości x
> x
```

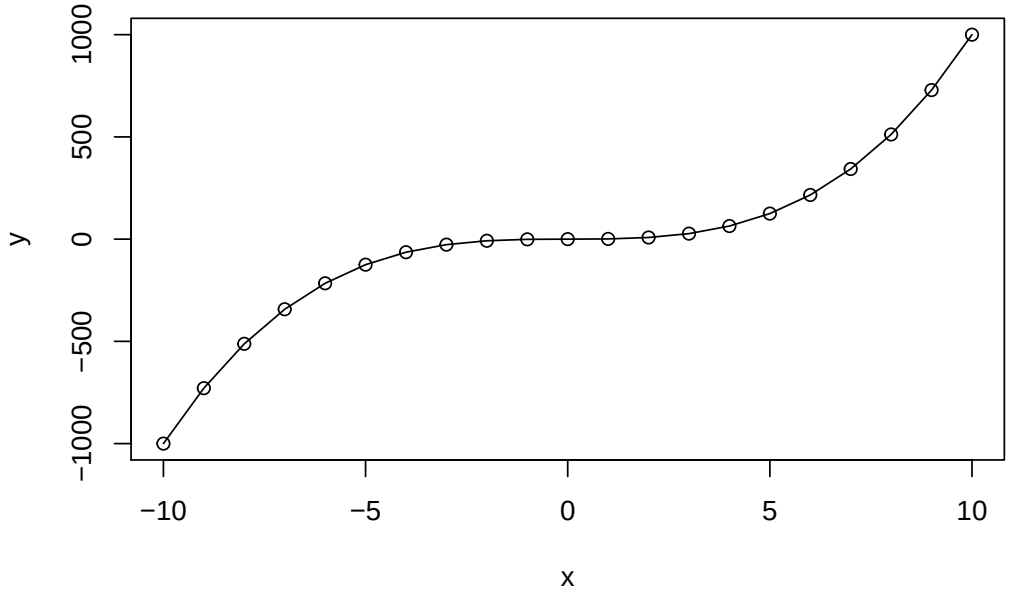
```
 [1] -10  -9  -8  -7  -6  -5  -4  -3  -2  -1   0   1   2   3   4   5   6
[18]   7   8   9  10
```

```
> y=x^3  # ustalenie funkcji y
plot(x, y)  # Plot x i y
```



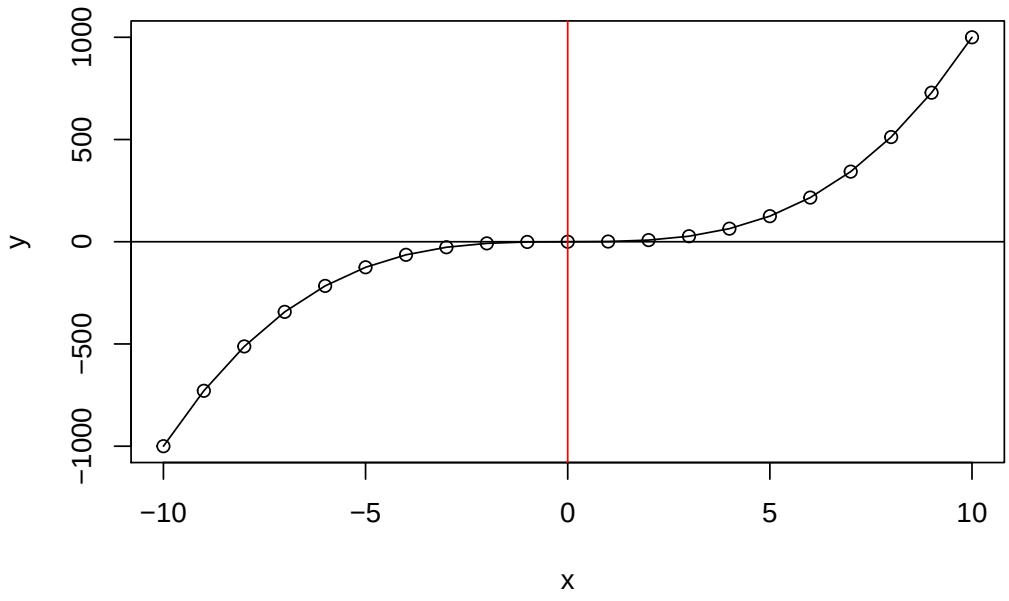
Rysunek 4.10: plot9

```r
lines(x,y)  # Dodanie linii łączących x i y.
```



Rysunek 4.11: plot10

```r
abline(h=0)  # Dodanie linii poziomej y=0
abline(v=0, col="red") # Dodanie czerwonej linii pionowej x=0
```



Rysunek 4.12: plot11

Przykład 4

Wykonać w jednym „oknie” wykresy funkcji $y = sin(x)$ oraz $y = cos(x)$ dla $x < -3; 3 >$.

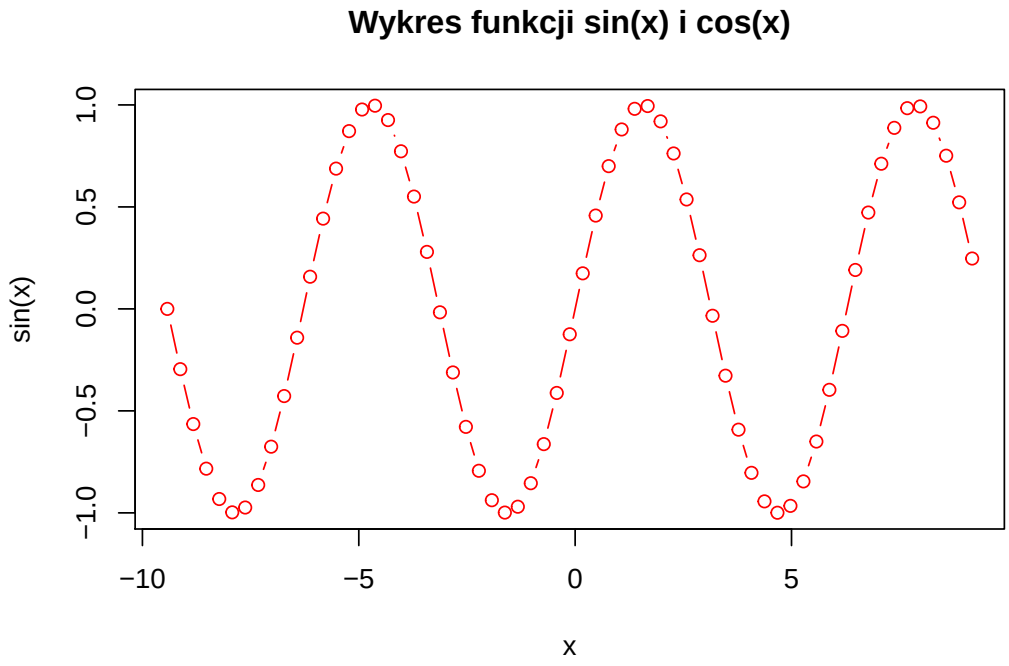
Kod w R:

```
# przygotowujemy siatke punktów
x = seq(-3*pi, 3*pi, by=0.3)
# rysujemy funkcje sin(x)
plot(x, sin(x), type="b", main="Wykres funkcji sin(x) i cos(x)", col="red")
# następnie dorysowujemy do niej funkcję cos(x)
lines(x, cos(x),  col="blue", type="l")
```

Realizacja w R

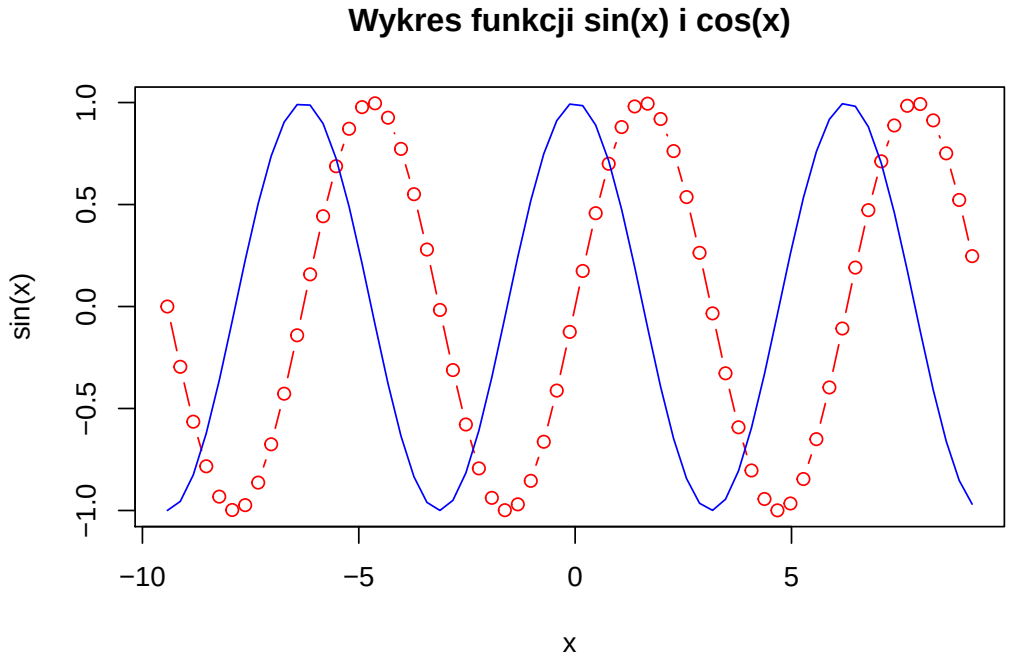
```
> # przygotowujemy siatke punktów
> x = seq(-3*pi, 3*pi, by=0.3)
```

```
# rysujemy funkcje sin(x)
plot(x, sin(x), type="b", main="Wykres funkcji sin(x) i cos(x)", col="red")
```



Rysunek 4.13: plot12

```
# następnie dorysowujemy do niej funkcję cos(x)
lines(x, cos(x),  col="blue", type="l")
```



Rysunek 4.14: plot14

Rozdział 5

Testowanie

5.1 Wprowadzenie

Mamy dane populacje w ramach których chcemy zweryfikować interesujące nas przypusz-czenie. Na przykład, dane jest 200 ha pole z pszenżytem odmiany A oraz 150 ha pole z pszenżytem odmiany B. Chcemy porównać ciężar nasion w kłosie. Oczywiście najlepszym sposobem postępowania jest zważenie nasion wszystkich kłosów z obu pól. Jak wiadomo, taka czynność nie jest wykonywana. Powinniśmy losowo wybrać kilkanaście lub kilkadziesiąt kłosów z pierwszego pola (próba A) i drugiego pola (próba B). Tak więc mamy populacje oraz mamy próby, gdzie najczęściej stosowane oznaczenia wybranych parametrów przedstawia Tabela 5.1.

Tablica 5.1: Podstawowe parametry dla populacji oraz próby.

populacja	próba
μ – średnia cechy w populacji	$\bar{x}$ średnia cechy w próbie
σ^2 – wariancja cechy w populacji	s^2 – wariancja cechy w próbie
σ – odchylenie standardowe cechy w populacji	s – odchylenie standardowe cechy w próbie

Testowanie jest to weryfikacja przypuszczeń. W opracowaniu tym rozpatrujemy testy parametryczne, czyli testy dotyczące parametrów populacji (np. średnia, wariancja). Postać przypuszczeń składa się z dwóch hipotez: hipotezy zerowej H_0 oraz hipotezy alternatywnej H_1. Po wybraniu właściwej statystyki, wyliczamy wartość tej statystyki dla wylosowanych prób oraz tzw. p-wartość (p-value) i podejmujemy decyzję: albo odrzucamy hipotezę zerową i przyjmujemy hipotezę alterna-

tywną, albo stwierdzamy brak podstaw do odrzucenia hipotezy zerowej (w praktyce przyjmujemy hipotezę zerową). Porównując rzeczywistość z naszą decyzją możemy mieć sytuacje przedstawione w 5.2. Prawdopodobieństwo odrzucenia hipote-zy prawdziwej jest błędem pierwszego rodzaju oznaczanym przez oraz nazywanym pozio-mem istotności. Natomiast prawdopodobieństwo przyjęcia hipotezy nie prawdziwej jest błędem drugiego rodzaju oznaczanym przez β oraz nazywanym mocą testu.

Tablica 5.2: Możliwe decyzji podczas testowania.

		Decyzja	
		H przyjmujemy	H odrzucamy
Rzeczywistość	H prawdziwa	OK	α
	H nie jest prawdziwa	β	OK

Reguły postępowania podczas testowania hipotez:

1. Mamy dane populacje w ramach których chcemy wykonać testowanie.
2. Formułujemy problem badawczy.
3. Ustalamy poziom istotności α, np. w tym manuskrypcie $\alpha = 0.05$.
4. Formułujemy hipotezę zerową H0 oraz hipotezę alternatywną H_1.
5. Losowo wybieramy próby.
6. Ustalamy właściwą statystykę.
7. Wyznaczamy parametry ustalonej statystki, m.in. p-wartość.
8. Podejmujemy decyzje:

   8.1. jeśli p-wartość <0.05 (poziom istotności), to odrzucamy hipotezę zerową i przyjmu-jemy hipotezę alternatywną,

   8.2. jeśli p-wartość 0.05, to przyjmujemy hipotezę zerową.
9. Dokonujemy interpretacji problemu badawczego.

Tablica 5.3: Wybrane testy statystyczne dla wartości średnich

		Dane z rozkładu normalnego	Dane nie są z rozkładu normalnego
Próby niezależne	2 grupy	Test t dla grup niezależnych (t.test)*	Test Wilcoxona (Wilcox.test)
	>2 grupy	ANOVA (aov)	Test Kruskala-Wallisa (kruskal.test)
Próby zależne (związane)	2 grupy	Test t związany (t.test)	Test Wilcoxona związany (wilcox.test)
	>2grupy	ANOVA (aov)	Test Friedmana (friedman.test)

* w nawiasach podane są nazwy funkcji w R.

Tabela 5.3 wskazuje, że wybór testu zależy od trzech charakterystyk:

1. czy dane podlegają rozkładowi normalnemu czy nie podlegają,
2. czy próby są niezależne, czy są zależne,
3. czy rozpatrujemy dwie próby (grupy), czy więcej niż dwie .

5.2 Testy istotności dwóch wartości średnich dla prób z rozkładów normalnych

Założenie:

Mamy dwie próby odpowiednio o liczebności n_1 z rozkładu $N(\mu_1, \sigma_1 2)$ oraz o liczebności n_2 z rozkładu $N(\mu_2, \sigma_2 2)$.

Możemy rozpatrywać hipotezę dwustronną lub hipotezę lewostronną lub hipotezę prawostronną.

Hipotezy

a) hipoteza dwustronna (test obustronny)

$$H_0 : \mu_1 = \mu_2$$

$$H_1 : \mu_1 \neq \mu_2$$

b) hipoteza lewostronna (test lewostronny)

$$H_0 : \mu_1 = \mu_2$$
$$H_1 : \mu_1 < \mu_2$$

c) hipoteza prawostronna (test prawostronny)

$$H_0 : \mu_1 = \mu_2$$
$$H_1 : \mu_1 > \mu_2$$

Rozpatrujemy dwie sytuacje: próby są niezależne lub próby są zależne (związane, sprzężone).

5.2.1 Próby niezależne

Przykład (Elandt 1964, s. 102)

Dany jest ciężar w gramach 1000 nasion dla dwóch rodów seradeli:

Tablica 5.4: Dane przykład Elandt 1964, s. 102

Ród A		Ród B	
1	3,8	1	3,7
2	3,7	2	4,6
3	2,9	3	5,4
4	3,5	4	6,2
5	2,6	5	4,2
6	3,3	6	3,5
		7	5,3
		8	5,5

Zweryfikować przypuszczenie, że średnie ciężary tych rodów różnią się istotnie.

Rozwiązanie

Niech μ_1 oznacza średni ciężar 1000 nasion rodu A, natomiast μ_2 oznacza średni ciężar 1000 nasion rodu B. Rozpatrujemy hipotezę obustronną postaci:

$$H_0 : \mu_1 = \mu_2$$

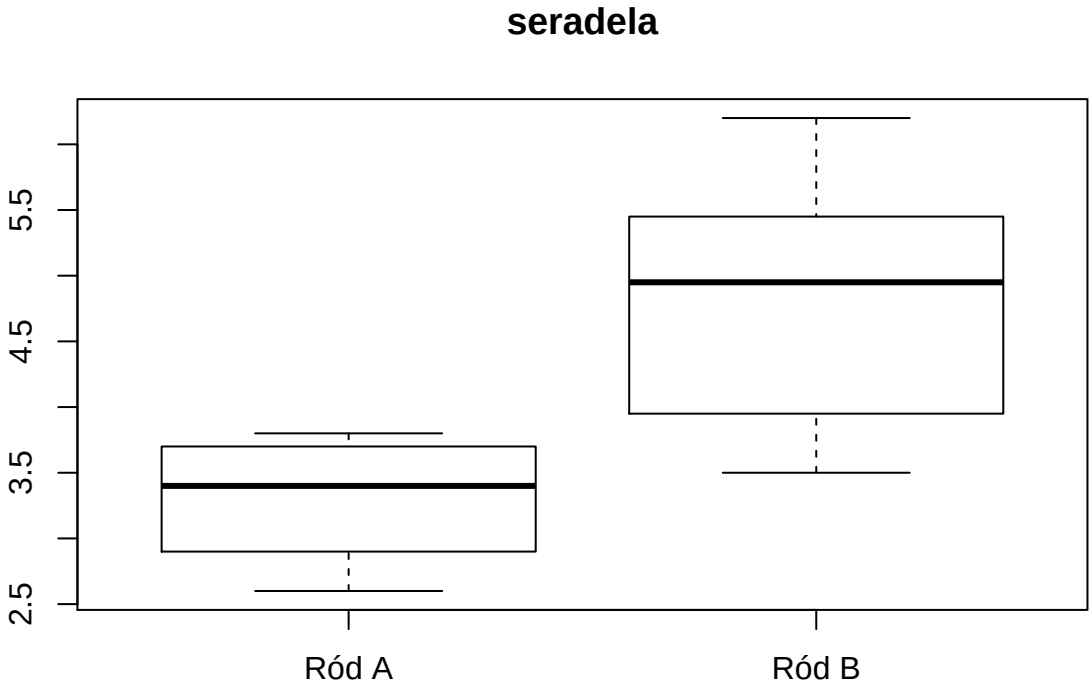
$$H_1 : \mu_1 \neq \mu_2$$

Kod w R:

```
# Elandt, przykład 3.8, str. 102
# Ciężar w g 1000 nasion dwóch rodów hodowlanych seradeli
# tworzenie danych
rodA=c(3.8, 3.7, 2.9, 3.5, 2.6, 3.3)
rodB=c(3.7, 4.6, 5.4, 6.2, 4.2, 3.5, 5.3, 5.5)
# prezentacja graficzna danych boxplotem
boxplot(rodA, rodB, names=c("Ród A","Ród B"), main="seradela")
```

Realizacja w R

```
> # Elandt, przykład 3.8, str. 102
> # Ciężar w g 1000 nasion dwóch rodów hodowlanych seradeli
> # tworzenie danych
> rodA=c(3.8, 3.7, 2.9, 3.5, 2.6, 3.3)
> rodB=c(3.7, 4.6, 5.4, 6.2, 4.2, 3.5, 5.3, 5.5)
> # prezentacja graficzna danych boxplotem
> boxplot(rodA, rodB, names=c("Ród A","Ród B"), main="seradela")
```



Sprawdzamy założenie o normalności rozkładów rodu A oraz rodu B

H_0: mamy rozkład normalny

H_1: rozkład normalny nie jest spełniony

Kod w R:

```
# sprawdzenie założeń o normalności rozkładów dla rodu A oraz rodu B
shapiro.test(rodA)
shapiro.test(rodB)
```

Realizacja w R

```
> # sprawdzenie założeń o normalności rozkładów dla rodu A oraz rodu B
> shapiro.test(rodA)


    Shapiro-Wilk normality test


data:  rodA
W = 0.93433, p-value = 0.6139
> shapiro.test(rodB)


    Shapiro-Wilk normality test


data:  rodB
W = 0.94586, p-value = 0.6694
```

Interpretacja: Założenia o normalności rozkładów są spełnione. Wykonujemy testowanie wykorzystując dwustronny test t.

Kod w R:

```
# obustronny test t
t.test(rodA, rodB)
```

Realizacja w R

```
> # obustronny test t
> t.test(rodA, rodB)


    Welch Two Sample t-test
```

```
data:  rodA and rodB
t = -3.8699, df = 10.69, p-value = 0.002749
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -2.3561458 -0.6438542
sample estimates:
mean of x mean of y
      3.3       4.8
```

Interpretacja: ponieważ p-wartość=0.002749 < 0.05, więc stwierdzamy, że ciężar 1000 nasion rodu A seradeli różni się od rodu B. Ponadto, analizując boxplot można przypuszczać, że ciężar 1000 nasion dla rodu A seradeli jest mniejszy niż ciężar 1000 nasion dla rodu B seradeli. Wobec tego, teraz zastosujemy lewostronny test t:

$$H_0 : \mu_1 = \mu_2$$
$$H_1 : \mu_1 < \mu_2$$

Kod w R:

```
# lewostronny test t
t.test(rodA, rodB, alternative="less")
```

Realizacja w R

```
> # lewostronny test t
> t.test(rodA, rodB, alternative="less")


    Welch Two Sample t-test

data:  rodA and rodB
t = -3.8699, df = 10.69, p-value = 0.001374
alternative hypothesis: true difference in means is less than 0
95 percent confidence interval:
      -Inf -0.802051
sample estimates:
mean of x mean of y
      3.3       4.8
```

Interpretacja: ponieważ p-wartość=0.001374 < 0.05, to stwierdzamy, że ciężar 1000 nasion rodu A seradeli jest mniejszy niż rodu B.

5.2.2 Próby zależne

Przykład (Elandt 1964, s. 109)

Oznaczono procent tłuszczu w 18 próbkach mleka za pomocą dwóch metod: metody Gerbera (metoda G) i metody Burata (metoda B).

Tablica 5.5: Dane do przykładu Elandt 1964, s. 109

Lp.	Metoda G	Metoda B	Lp.	Metoda G	Metoda B
1	2,73	2,88	10	3,07	3,23
2	2,84	2,93	11	2,66	2,81
3	3,18	3,38	12	2,78	2,94
4	2,79	2,99	13	3,62	3,59
5	3,05	3,30	14	3,31	3,41
6	3,03	3,19	15	2,71	2,88
7	3,10	3,34	16	2,80	2,99
8	2,88	3,08	17	2,95	3,16
9	3,00	3,20	18	3,52	3,66

Czy metody te dają takie same wyniki?

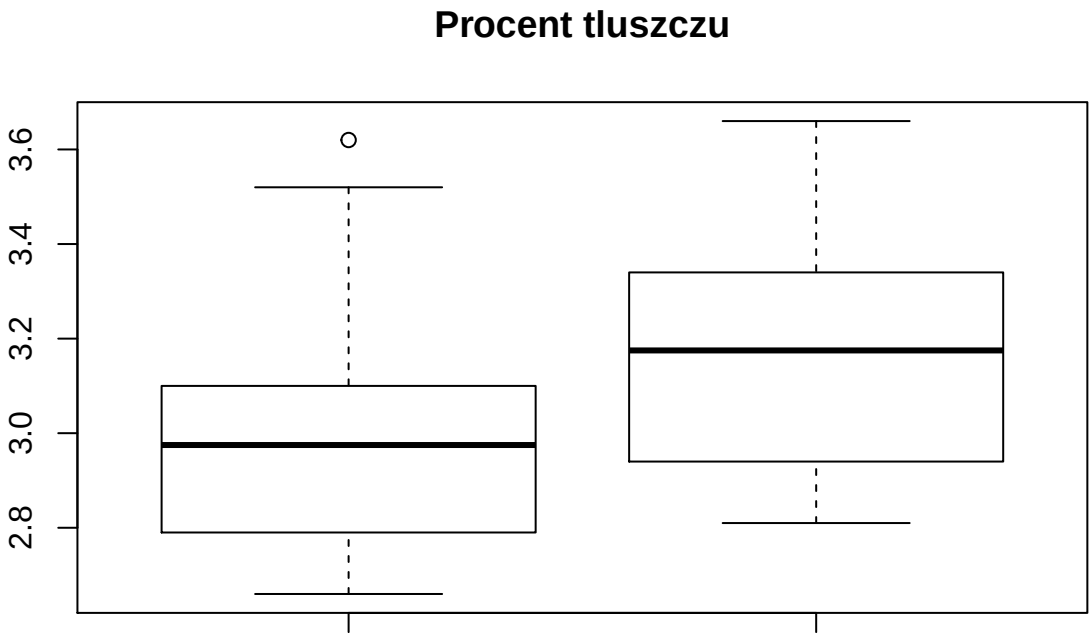
Kod w R:

```
rm(list=ls()) # usuwanie wszystkich zmiennych z przestrzeni roboczej
# Elandt, przyklad 3.12, str. 109
# Procent tluszczu w mleku
# zadanie: porownac zmiennosc wynikow w obu metodach (Gerbera, Burata)
# tworzenie danych
metodaG=c(2.73, 2.84, 3.18, 2.79, 3.05, 3.03, 3.10, 2.88, 3.00, 3.07,
          2.66, 2.78, 3.62, 3.31, 2.71, 2.80, 2.95, 3.52)
metodaB=c(2.88, 2.93, 3.38, 2.99, 3.30, 3.19, 3.34, 3.08, 3.20, 3.23,
          2.81, 2.94, 3.59, 3.41, 2.88, 2.99, 3.16, 3.66)
# prezentacja graficzna danych - boxplot
boxplot(metodaG, metodaB, main="Procent tłuszczu")
```

```
# Sprawdzamy założenia o normalności rozkładów
shapiro.test(metodaG)
shapiro.test(metodaB)
```

Realizacja w R

```
> rm(list=ls()) # usuwanie wszystkich zmiennych z przestrzeni roboczej
> # Elandt, przyklad 3.12, str. 109
> # Procent tluszczu w mleku
> # zadanie: porownac zmiennosc wynikow w obu metodach (Gerbera, Burata)
> # tworzenie danych
> metodaG=c(2.73, 2.84, 3.18, 2.79, 3.05, 3.03, 3.10, 2.88, 3.00, 3.07,
+           2.66, 2.78, 3.62, 3.31, 2.71, 2.80, 2.95, 3.52)
> metodaB=c(2.88, 2.93, 3.38, 2.99, 3.30, 3.19, 3.34, 3.08, 3.20, 3.23,
+           2.81, 2.94, 3.59, 3.41, 2.88, 2.99, 3.16, 3.66)
> # prezentacja graficzna danych - boxplot
> boxplot(metodaG, metodaB, main="Procent tluszczu")
```




```
> # Sprawdzamy założenia o normalności rozkładów
> shapiro.test(metodaG)
```

```
    Shapiro-Wilk normality test
```

```
data:  metodaG
W = 0.91487, p-value = 0.1049
```

```
> shapiro.test(metodaB)
```

```
    Shapiro-Wilk normality test

data:  metodaB
W = 0.95253, p-value = 0.4661
```

Interpretacja: dla obu prób spełnione jest założenie o normalności rozkładów. Możemy wykonać test t.

Uwaga: Ponieważ te same obiekty badane są dwa razy - należy zastosować test t dla par zależnych.

Kod w R:

```
# obustronny test t dla par zależnych
t.test(metodaG, metodaB, paired = TRUE)
```

Realizacja w R

```
> # obustronny test t dla par zależnych
> t.test(metodaG, metodaB, paired = TRUE)

    Paired t-test

data:  metodaG and metodaB
t = -10.846, df = 17, p-value = 4.651e-09
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -0.1951067 -0.1315599
sample estimates:
mean of the differences
             -0.1633333
```

Komentarz: Można zastosować lewostronny test t dla par zależnych.

Kod w R:

```
# lewostronny test t dla par zależnych
t.test(metodaG, metodaB, alternative="less", paired = TRUE)
```

Realizacja w R

```
> # lewostronny test t dla par zależnych
> t.test(metodaG, metodaB, alternative="less", paired = TRUE)


    Paired t-test

data:  metodaG and metodaB
t = -10.846, df = 17, p-value = 2.326e-09
alternative hypothesis: true difference in means is less than 0
95 percent confidence interval:
       -Inf -0.1371352
sample estimates:
mean of the differences
             -0.1633333
```

Interpretacja: metoda Gerbera daje mniejszy procent tłuszczu w badanym mleku niż metoda Burata.

5.3 Testy istotności dla dwóch wartości średnich z dowolnych rozkładów

Założenie: Co najmniej jedna próba nie podlega rozkładowi normalnemu.

Przykład (Grzegorzewski 2011, SM-VI, s.13)

W celu zbadania, czy nowy rodzaj paliwa ma istotny wpływ na zasięg jazdy samochodu, wykonano 10 pomiarów przejechanej drogi na nowym oraz na starym paliwie. Otrzymano:

Tablica 5.6: Dane do przykładu Grzegorzewski 2011, SM-VI, s.13

Stare paliwo	1039	1168	1008	1035	1035	1025	1059	1012	1012	1039
Nowe paliwo	1096	1161	1210	1088	1154	1111	1103	1094	1059	1177

Czy nowy rodzaj paliwa ma istotny wpływ na wzrost przeciętnej przejechanej drogi?

Kod w R:

```
# PG-paliwo
# tworzymy dane
stare = c(1039, 1168, 1008, 1035, 1035, 1025, 1059, 1012, 1012, 1039)
```

```
nowe = c(1096, 1161, 1210, 1088, 1154, 1111, 1103, 1094, 1059, 1177)
# sprawdzenie normalność rozkładów
shapiro.test(stare)
shapiro.test(nowe)
```

Realizacja w R

```
> # PG-paliwo
> # tworzymy dane
> stare = c(1039, 1168, 1008, 1035, 1035, 1025, 1059, 1012, 1012, 1039)
> nowe = c(1096, 1161, 1210, 1088, 1154, 1111, 1103, 1094, 1059, 1177)
> # sprawdzenie normalność rozkładów
> shapiro.test(stare)

    Shapiro-Wilk normality test

data:  stare
W = 0.675, p-value = 0.0004392

> shapiro.test(nowe)

    Shapiro-Wilk normality test

data:  nowe
W = 0.93552, p-value = 0.5043
```

Uwaga. Ponieważ jedna z prób nie spełnia warunku rozkładu normalnego, więc nie możemy skorzystać z testu t. Zastosujemy test Wilcoxona.

Kod w R:

```
# test wilcoxona
wilcox.test(stare,  nowe,  alternative="less")
```

Realizacja w R

```
> # test wilcoxona
> wilcox.test(stare,  nowe,  alternative="less")

Warning in wilcox.test.default(stare, nowe, alternative = "less"): cannot
compute exact p-value with ties
```

```
    Wilcoxon rank sum test with continuity correction

data:  stare and nowe
W = 8.5, p-value = 0.0009547
alternative hypothesis: true location shift is less than 0
```

Interpretacja: Nowy rodzaj paliwa umożliwia przejechanie więcej kilometrów.

Przykład 4.

Na pierwszym roku studiów przebadano 5 studentów oraz 4 studentki pod względem zdolności matematycznych w celu weryfikacji przypuszczenia, że studenci są pod tym względem lepsi od studentek. Wyniki testu są następujące :

Tablica 5.7: Wyniki studentów

studenci	15	21	22	24	18	19	23	19	23
studentki	15	19	23	25	10	15	22	21	

$$H_0 : \mu_1 = \mu_2$$
$$H_1 : \mu_1 > \mu_2$$

Uwagi:

1) ponieważ otrzymane wyniki są liczbami naturalnymi, więc populacje nie mogą spełniać warunku o normalności rozkładów – rozkład normalny jest rozkładem ciągłym, a my mamy tutaj rozkład dyskretny.
2) nie stosujemy testu t, tylko test Wilcoxona.

Kod w R:

```
studenci = c(15, 21, 22, 24, 18, 19, 23, 19, 23)
studentki = c(15, 19, 23, 25, 10, 15, 22, 21)
wilcox.test(studenci, studentki, alternative="greater")
```

Realizacja w R

```
> studenci = c(15, 21, 22, 24, 18, 19, 23, 19, 23)
> studentki = c(15, 19, 23, 25, 10, 15, 22, 21)
> wilcox.test(studenci, studentki, alternative="greater")
```

```
Warning in wilcox.test.default(studenci, studentki, alternative =
"greater"): cannot compute exact p-value with ties


    Wilcoxon rank sum test with continuity correction

data:  studenci and studentki
W = 42, p-value = 0.2967
alternative hypothesis: true location shift is greater than 0
```

Decyzja: nie ma podstaw do odrzucenia hipotezy H_0.

Interpretacja: Zdolności matematyczne studentów i studentek I roku są takie same.

5.4 Analiza wariancji - ANOVA

Mamy $r > 2$ populacji. Z każdej, losowo pobieramy po jednej próbie.

Założenia ANOVY

1. Niezależność - próby zostały pobrane nieżalenie od siebie z każdej z r populacji.

2. Normalność - w każdej z r populacji rozkład badanej cechy jest normalny

$H_0 : X\ rozkadnormalnyvsH_1 : \neg H_0$

3. Jednorodność wariancji - wariancje rozkładu badanej cechy są takie same w r populacjach

$$H_0 : \sigma_{12} = \sigma_{22} = ... = \sigma_{r2}$$
$$H1 : \neg H_0$$

Przykład (Greń 1975, s.161)

Wylosowano po 12 pędów żyta trzech różnych gatunków i otrzymano dla nich następujące długości kłosów żyta (w cm):

Tablica 5.8: Dane do przykładu Greń 1975, s.161

Gatunek			Gatunek		
A	B	C	A	B	C
6,7	7,5	5,9	10,1	10,6	9,6
7,3	7,7	6,9	9,2	10,2	10,3
8,0	7,7	7,0	8,3	9,4	8,1
8,0	8,2	7,0	8,4	9,4	8,5
7,9	8,9	9,5	8,0	8,2	8,6
9,2	8,9	9,6	7,9	7,8	8,8

Czy długości kłosów badanych gatunków są różne?

Rozwiązanie Należy zweryfikować następujące hipotezy:

$$H_0 : \mu_A = \mu_B = \mu_C$$
$$H_1 : \neg H_0$$

,gdzie μ_K oznacza średnią długość kłosów gatunku K.

Kod w R:

```
# Przykład (Greń 1975, s.161)
rm(list=ls()) # usuwanie wszystkich zmiennych z przestrzeni roboczej
# tworzenie danych
A = c(6.7,7.3,8.0,8.0,7.9,9.2,10.1,9.2,8.3,8.4,8.0,7.9)
B = c(7.5,7.7,7.7,8.2,8.9,8.9,10.6,10.2,9.4,9.4,8.2,7.8)
C = c(5.9,6.9,7.0,7.0,9.5,9.6,9.6,10.3,8.1,8.5,8.6,8.8)
# sprawdzamy założenie o normalności rozkładów
shapiro.test(A)
shapiro.test(B)
shapiro.test(C)
# przygotowanie danych
zyto=data.frame(Dlugosc=c(A, B, C), Gat=c(rep(c("A","B","C"), c(12,12,12))))
zyto
# weryfikacja założenia o jednorodności wariancji - test Bartleta
bartlett.test(zyto$Dlugosc,zyto$Gat)
```

```
# ANOVA
model=aov(Dlugosc~Gat, data=zyto)
summary(model)
```

Realizacja w R

```
> # Przykład (Greń 1975, s.161)
> rm(list=ls()) # usuwanie wszystkich zmiennych z przestrzeni roboczej
> # tworzenie danych
> A = c(6.7,7.3,8.0,8.0,7.9,9.2,10.1,9.2,8.3,8.4,8.0,7.9)
> B = c(7.5,7.7,7.7,8.2,8.9,8.9,10.6,10.2,9.4,9.4,8.2,7.8)
> C = c(5.9,6.9,7.0,7.0,9.5,9.6,9.6,10.3,8.1,8.5,8.6,8.8)
> # sprawdzamy założenie o normalności rozkładów
> shapiro.test(A)

    Shapiro-Wilk normality test

data:  A
W = 0.93886, p-value = 0.4835
> shapiro.test(B)

    Shapiro-Wilk normality test

data:  B
W = 0.91484, p-value = 0.246
> shapiro.test(C)

    Shapiro-Wilk normality test

data:  C
W = 0.94392, p-value = 0.5505
> # przygotowanie danych
> zyto=data.frame(Dlugosc=c(A, B, C), Gat=c(rep(c("A","B","C"), c(12,12,12))))
> zyto
   Dlugosc Gat
```

1	6.7	A
2	7.3	A
3	8.0	A
4	8.0	A
5	7.9	A
6	9.2	A
7	10.1	A
8	9.2	A
9	8.3	A
10	8.4	A
11	8.0	A
12	7.9	A
13	7.5	B
14	7.7	B
15	7.7	B
16	8.2	B
17	8.9	B
18	8.9	B
19	10.6	B
20	10.2	B
21	9.4	B
22	9.4	B
23	8.2	B
24	7.8	B
25	5.9	C
26	6.9	C
27	7.0	C
28	7.0	C
29	9.5	C
30	9.6	C
31	9.6	C
32	10.3	C
33	8.1	C
34	8.5	C
35	8.6	C
36	8.8	C

Interpretacja: wszystkie p-wartości > 0.05, więc H_0 nie odrzucamy co oznacza, że próby pochodzą z rozkładu normalnego.

```
> # weryfikacja założenia o jednorodności wariancji - test Bartleta
> bartlett.test(zyto$Dlugosc,zyto$Gat)


    Bartlett test of homogeneity of variances


data:  zyto$Dlugosc and zyto$Gat
Bartlett's K-squared = 1.8934, df = 2, p-value = 0.388
```

Interpretacja: p-wartość=0.388 > 0.05, więc nie odrzucamy H_0, a to oznacza, że założenie o jednorodności wariancji jest spełnione - możemy wykonać analizę wariancji ANOVA.

```
> # ANOVA
> model=aov(Dlugosc~Gat, data=zyto)
> summary(model)

            Df Sum Sq Mean Sq F value Pr(>F)
Gat          2   1.47  0.7358   0.592  0.559
Residuals   33  41.00  1.2423
```

Interpretacja: p-wartość=0.559 > 0.05, więc nie odrzucamy H_0, czyli długości kłosów badanych trzech gatunków żyta nie różnią się.

Uwaga: W takiej sytuacji nie wykonuje się porównań wielokrotnych.

Przykład (Kala 2005, s.163).

Porównano długości kłosów czterech odmian uprawnych D, A, J i N pewnej trawy. Uzyskano następujące obserwacje (w cm): D: 24.7, 26.6, 23.7, 18.8, 23.4, 20.6, 26.0, 27.9, 25.6 A: 19.2, 24.2, 14.2, 19.2, 18.1, 21.2, 19.0, 16.8, 15.0, 14.6 J: 22.7, 18.5, 23.6, 21.9, 20.0, 23.5, 17.0, 18.0 N: 19.9, 13.7, 16.8, 18.6, 23.0, 16.3, 15.2, 14.1, 16.9, 13.7 Dokonać szczegółowych porównań odmian.

Rozwiązanie

Formułujemy następujące hipotezy:

H_0: długości kłosów nie różnią się,

H_1: długości kłosów różnią się.

Kod w R:

```
# Kala 2005, s. 163
```

```
rm(list=ls()) # usuwanie wszystkich zmiennych z przestrzeni roboczej
library(agricolae) # aktywowanie pakietu agricolae
# tworzenie danych
D = c(24.7,26.6,23.7,18.8,23.4,20.6,26,27.9,25.6)
A = c(19.2,24.2,14.2,19.2,18.1,21.2,19,16.8,15,14.6)
J = c(22.7,18.5,23.6,21.9,20,23.5,17,18)
N = c(19.9,13.7,16.8,18.6,23,16.3,15.2,14.1,16.9,13.7)
B=c(rep("D",9), rep("A",10), rep("J",8), rep("N",10))
B
trawa=data.frame(Dlugosc=c(D,A,J,N), Odmiany=B)
trawa
# sprawdzamy założenie o normalności rozkładów dla odmian
shapiro.test(D)
shapiro.test(A)
shapiro.test(J)
shapiro.test(N)
# weryfikacja założenia o jednorodności wariancji
bartlett.test(trawa$Dlugosc,trawa$Odmiany)
# ANOVA
model = aov(Dlugosc~Odmiany, trawa)
summary(model)
# testowanie szczegółowe - test wielokrotny Tukeya
(HSD.test(model,"Odmiany"))   # funkcja z pakietu agricolae
TukeyHSD(model,"Odmiany", ordered = TRUE)  # funkcja z pakietu stats
plot(TukeyHSD(model,"Odmiany"))
```

Realizacja w R

```
> # Kala 2005, s. 163
> rm(list=ls()) # usuwanie wszystkich zmiennych z przestrzeni roboczej
> library(agricolae) # aktywowanie pakietu agricolae
```

```
Warning: package 'agricolae' was built under R version 3.2.5
```

```
Warning: replacing previous import by 'expm::expm' when loading 'spdep'
```

```
> # tworzenie danych
> D = c(24.7,26.6,23.7,18.8,23.4,20.6,26,27.9,25.6)
> A = c(19.2,24.2,14.2,19.2,18.1,21.2,19,16.8,15,14.6)
```

```
> J = c(22.7,18.5,23.6,21.9,20,23.5,17,18)
> N = c(19.9,13.7,16.8,18.6,23,16.3,15.2,14.1,16.9,13.7)
> B=c(rep("D",9), rep("A",10), rep("J",8), rep("N",10))
> B
```

```
 [1] "D" "D" "D" "D" "D" "D" "D" "D" "D" "A" "A" "A" "A" "A" "A" "A" "A"
[18] "A" "A" "J" "J" "J" "J" "J" "J" "J" "J" "N" "N" "N" "N" "N" "N" "N"
[35] "N" "N" "N"
```

```
> trawa=data.frame(Dlugosc=c(D,A,J,N), Odmiany=B)
> trawa
```

```
   Dlugosc Odmiany
1     24.7       D
2     26.6       D
3     23.7       D
4     18.8       D
5     23.4       D
6     20.6       D
7     26.0       D
8     27.9       D
9     25.6       D
10    19.2       A
11    24.2       A
12    14.2       A
13    19.2       A
14    18.1       A
15    21.2       A
16    19.0       A
17    16.8       A
18    15.0       A
19    14.6       A
20    22.7       J
21    18.5       J
22    23.6       J
23    21.9       J
24    20.0       J
25    23.5       J
```

```
26    17.0       J
27    18.0       J
28    19.9       N
29    13.7       N
30    16.8       N
31    18.6       N
32    23.0       N
33    16.3       N
34    15.2       N
35    14.1       N
36    16.9       N
37    13.7       N
```

```
> # sprawdzamy założenie o normalności rozkładów dla odmian
> shapiro.test(D)
```

```
    Shapiro-Wilk normality test

data:  D
W = 0.94245, p-value = 0.608
```

```
> shapiro.test(A)
```

```
    Shapiro-Wilk normality test

data:  A
W = 0.9408, p-value = 0.5619
```

```
> shapiro.test(J)
```

```
    Shapiro-Wilk normality test

data:  J
W = 0.90125, p-value = 0.2965
```

```
> shapiro.test(N)
```

```
    Shapiro-Wilk normality test


data:  N
W = 0.91073, p-value = 0.2861
```

Wniosek: zachodzą warunki normalności dla odmian D, A, J, N.

```
> # weryfikacja założenia o jednorodności wariancji
> bartlett.test(trawa$Dlugosc,trawa$Odmiany)


    Bartlett test of homogeneity of variances


data:  trawa$Dlugosc and trawa$Odmiany
Bartlett's K-squared = 0.25106, df = 3, p-value = 0.969
```

Wniosek: warunek jednorodności jest spełniony.

```
> # ANOVA
> model = aov(Dlugosc~Odmiany, trawa)
> summary(model)

            Df Sum Sq Mean Sq F value   Pr(>F)
Odmiany      3  291.7   97.22   11.25 3.07e-05 ***
Residuals   33  285.3    8.64
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Decyzja: ponieważ p-wartość < 0.05, więc odrzucamy hipotezę H_0 i przyjmujemy H_1.

Interpretacja: długości kłosów czterech odmian uprawnych D, A, J i N badanej trawy różnią się statystycznie istotnie.

Uwaga: Ponieważ odrzuciliśmy hipotezę zerową H_0 i przyjęliśmy hipotezę alternatywną H_1, to możemy zastosować testy szczegółowe, czyli testy wielokrotne, np. test Tukeya.

5.5 Testy wielokrotne

Najczęściej stosowane testy wielokrotne:

1. Test HSD Tukeya (Honestly Significant Differences)
2. Test LSD Fishera (Least Significant Differences) – NIR: Najmniejsza Istotna Różnica

3. Test Scheffego
4. Test Duncana
5. Test Newmana-Keulsa

Uwagi: 1.Test Tukeya jest bardziej konserwatywny (ostrożny, rzadziej odrzuca H_0) niż test Fishera, a test Fishera jest bardziej konserwatywny niż test Scheffego. 2.Test Tukeya jest preferowany i najczęściej stosowany, ponieważ mamy zagwarantowany poziom istotności α dla wszystkich porównywanych par.

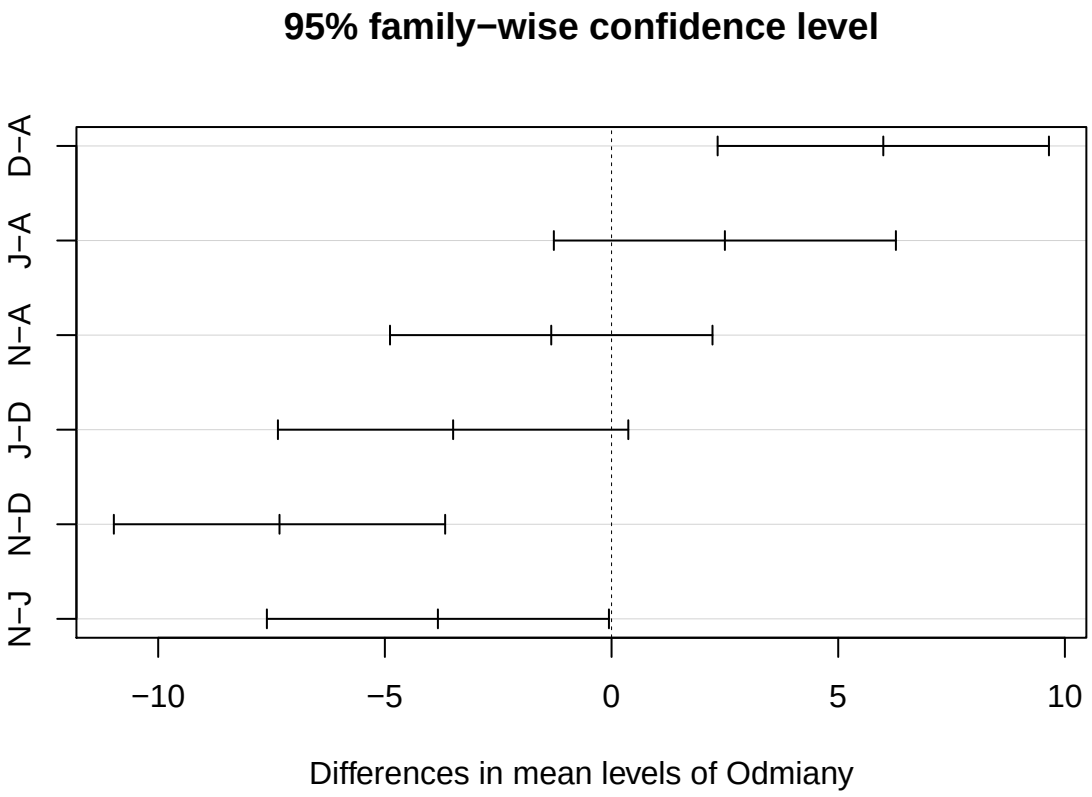
Zastosujemy test Tukeya.

Kod w R:

```
# testowanie szczegółowe - test wielokrotny Tukeya
HSD.test(model,"Odmiany")  # funkcja z pakietu agricolae
TukeyHSD(model,"Odmiany", ordered = TRUE)  # funkcja z pakietu stats
plot(TukeyHSD(model,"Odmiany"))
```

Realizacja w R

```
> # testowanie szczegółowe - test wielokrotny Tukeya
> HSD.test(model,"Odmiany")  # funkcja z pakietu agricolae
> TukeyHSD(model,"Odmiany", ordered = TRUE)  # funkcja z pakietu stats

  Tukey multiple comparisons of means
    95% family-wise confidence level
    factor levels have been ordered

Fit: aov(formula = Dlugosc ~ Odmiany, data = trawa)

$Odmiany
        diff         lwr       upr     p adj
A-N 1.330000 -2.22663884  4.886639 0.7438813
J-N 3.830000  0.05761484  7.602385 0.0455162
D-N 7.324444  3.67034540 10.978543 0.0000304
J-A 2.500000 -1.27238516  6.272385 0.2948589
D-A 5.994444  2.34034540  9.648543 0.0005319
D-J 3.494444 -0.36996363  7.358853 0.0879854

> plot(TukeyHSD(model,"Odmiany"))
```



Rysunek 5.1: Graficzne przedstawienie porównań wielokrotnych.

Tablica 5.9: Porównania pomiędzy odmianami (p-wartość)

	A	J	N
D	0.0005319	0.0879854	0.0000304
A		0.2948589	0.7438813
J			0.0455162

lub

	A	J	N
D	x	ns	x
A		ns	ns
J			x

x - statystycznie istotna różnica, ns – nie ma różnicy

Odmiany	Średnie*
D	24.14^{a}
J	20.65^{ab}
A	18.25^{bc}
N	16.82^{c}

- mała literka (indeks górny) oznacza grupę odmian podobnych.

Rozdział 6

Badanie zależności cech

6.1 Korelacje

Korelacja wskazuje siłę i kierunek zależności pomiędzy dwiema cechami. Korelacja dla pró-by wyrażona jest za pomocą współczynnika korelacji r, gdzie $r < -1; 1 >$.

Klasyfikacja współczynnika korelacji r:

$|r| = 0$ - brak korelacji,

$0,0 < |r| 0,1$ korelacja nikła,

$0,1 < |r| 0,3$ korelacja słaba,

$0,3 < |r| 0,5$ korelacja przeciętna,

$0,5 < |r| 0,7$ - korelacja wysoka,

$0,7 < |r| 0,9$ korelacja bardzo wysoka,

$0,9 < |r| < 1,0$ korelacja niemal pełna,

$|r| = 1$ korelacja pełna.

Jeśli wartość współczynnika korelacji r jest dodatnia to mamy zależność liniową dodatnią. Oznacza to, że wraz ze wzrostem jednostkowym wartości jednej cechy rosną z tym samym przyrostem wartości drugiej z cech. Natomiast, jeśli wartość współczynnika korelacji r jest ujemna to mamy zależność liniową ujrmną, tzn. wraz ze wzrostem jednostkowym wartości jednej cechy maleją o tą samą wielkość wartości drugiej z cech.

Cecha ilościowa (mierzalna) jest to cecha, która przyjmuje wartości liczbowe. Cecha jakościowa jest to cecha, która ma charakter opisowy lub podlega kategoryzacji.

6.1.1 Cechy ilościowe

Współczynnik korelacji liniowej Pearsona

- Współczynnik ten (r) jest miernikiem siły związku prostoliniowego między dwoma cechami mierzalnymi.
- Związkiem prostoliniowym nazywamy taką zależność, w której jednostkowym przyrostom jednej zmiennej (przyczyny) towarzyszy, średnio biorąc, stały przyrost drugiej zmiennej (skutku).

Przykład. (Dobek, Szwaczkowski s. 153, p. 10.2.2.1) Badano zależność pomiędzy długością pędu (cm) a długością kłosa (cm) pewnej odmiany pszenicy. Z poletka wybrano losowo 25 roślin, u których dokonano pomiaru obydwu cech. Wyniki zaprezentowano w Tabeli 6.1.

Tablica 6.1: Dane do przykładu ...

Numer rośliny	długość pędu (cm)	długość kłosa (cm)	Numer rośliny	długość pędu (cm)	długość kłosa (cm)
nr	dp	dk	nr	dp	dk
1	105	5,6	14	107	6,6
2	103	6,2	15	106	6,4
3	101	4,8	16	102	5,0
4	107	6,5	17	100	4,9
5	103	5,4	18	100	5,0
6	102	5,0	19	106	6,0
7	104	5,6	20	105	4,9
8	103	6,0	21	105	4,8
9	102	4,9	22	101	5,2
10	106	6,3	23	105	4,8
11	105	5,2	24	101	5,1
12	101	4,9	25	101	5,0
13	103	5,3			

Czy korelacja między badanymi cechami jest istotna?

Kod w R:

```
# Dobek, Szwaczkowski s. 153, p. 10.2.2.1
rm(list=ls()) # usuwanie wszystkich zmiennych z przestrzeni roboczej
# tworzenie danych
```

```
pszenica=read.table("~/Desktop/Dobek_153.txt",header=T)
head(pszenica)
# współczynnik korelacji
round(cor(pszenica$dk,pszenica$dp),2)
# testowanie korelacji
cor.test(pszenica$dk,pszenica$dp)
```

Realizacja w R:

```
> # Dobek, Szwaczkowski s. 153, p. 10.2.2.1
> rm(list=ls()) # usuwanie wszystkich zmiennych z przestrzeni roboczej
> setwd("D://abc") # ustanowienie aktualnego folderu
> # tworzenie danych
> pszenica=read.table("Dobek-153.txt", header=T)
> head(pszenica)
> # współczynnik korelacji
> round(cor(pszenica$dk,pszenica$dp),2)
> # testowanie korelacji
> cor.test(pszenica$dk,pszenica$dp)
```

Interpretacja. Wartość współczynnika korelacji Pearsona wynosi 0,664, więc korelacja jest wysoka. Ponadto, ponieważ p-wartość = 0,0003, odrzucamy hipotezę zerową i przyjmujemy hipotezę alternatywną stwierdzając, że zależność pomiędzy długością pędu a długością kłosa pewnej od-miany pszenicy jest istotna.

6.1.2 Cechy jakościowe

Współczynnik korelacji Spearmana

Współczynnik korelacji Spearmana rS używamy w przypadku gdy:

1. choć jedna z badanych cech jest cecha jakościowa (niemierzalna), ale istnieje możliwość uporządkowania (ponumerowania) wariantów każdej z cech;
2. cechy mają charakter ilościowy (mierzalny), ale liczebność zbiorowości jest mała (n<30).

Współczynnik korelacji Spearmana.

Współczynnik ten służy do opisu siły korelacji dwóch cech, szczególnie wtedy, gdy mają one charakter jakościowy i istnieje możliwość uporządkowania obserwacji w określonej kolejno-ści

Przykład. (Dobek, Szwaczkowski 2007, s. 163, zad. 4) Dwaj eksperci niezależnie oceniali stopień porażenia ziarniaków w skali od 1do 20. Uzyskali następujące oznaczenia:

ekspert 1: 5, 7, 34, 9, 12, 16, 9, 13, 18, 6, 17

ekspert 2: 6, 6, 3, 10, 8, 18, 10, 11, 16, 8, 15

Czy oceny obu ekspertów są skorelowane?

Kod w R:

```
# Dobek, Szwaczkowski s. 163, zad. 4.
eksp1=c(5, 7, 34, 9, 12, 16, 9, 13, 18, 6, 17)
eksp2=c(6, 6, 3, 10, 8, 18, 10, 11, 16, 8, 15)
cor.test(eksp1, eksp2, method = "spearman")
```

Realizacja w R:

```
> # Dobek, Szwaczkowski s. 163, zad. 4.
> eksp1=c(5, 7, 34, 9, 12, 16, 9, 13, 18, 6, 17)
> eksp2=c(6, 6, 3, 10, 8, 18, 10, 11, 16, 8, 15)
> cor.test(eksp1, eksp2, method = "spearman")
```

```
Warning in cor.test.default(eksp1, eksp2, method = "spearman"): Cannot
compute exact p-value with ties


    Spearman's rank correlation rho

data:  eksp1 and eksp2
S = 130.18, p-value = 0.2126
alternative hypothesis: true rho is not equal to 0
sample estimates:
      rho
0.4082612
```

Interpretacja. Wartość współczynnika korelacji Spearmana wynosi 0,408, więc korelacja jest przeciętna. Ponadto, ponieważ p-wartość = 0,2126, więc nie odrzucamy hipotezy zerowej i stwierdzamy, że współczynnika korelacji Spearmana nie różni się istotnie od zera. Oznacza to, że oceniani eksperci niezależnie ocenili stopień porażenia ziarniaków.

6.2 Tablice kontyngencji

Przykład. (Kala 2005, s. 87, p. 9.2)

Badając jakość jabłek oceniono owoce ze względu na uszkodzenia spowodowane przez owo-cówkę jabłkóweczkę (U - owoce uszkodzone, N - owoce nieuszkodzone) oraz porażone parchem jabłoniowym (C - owoce czyste, P - owoce z plamami). W wyniku klasyfikacji owoców uzyskano następujące liczebności:

Tablica 6.2: Dane do przykładu...

Parch	Owocówka	
	U	N
C	29	194
P	17	68

Czy na poziomie istotności 0,01 można uznać, że badane zmienne są niezależne?

Kod w R:

```
# analiza tablicy kontyngencji
# Kala 2005, s. 87
x <- matrix(c(29, 17, 194, 68), ncol = 2)
x
chisq.test(x)
chisq.test(x, correct = TRUE)
fisher.test(x)
```

Realizacja w R:

```
> # analiza tablicy kontyngencji
> # Kala 2005, s. 87
> x <- matrix(c(29, 17, 194, 68), ncol = 2)
> x
```

```
     [,1] [,2]
[1,]   29  194
[2,]   17   68
```

```
> chisq.test(x)
```

```
    Pearson's Chi-squared test with Yates' continuity correction

data:  x
X-squared = 1.8519, df = 1, p-value = 0.1736
```

```
> chisq.test(x, correct = TRUE)


    Pearson's Chi-squared test with Yates' continuity correction

data:  x
X-squared = 1.8519, df = 1, p-value = 0.1736
> fisher.test(x)


    Fisher's Exact Test for Count Data

data:  x
p-value = 0.1518
alternative hypothesis: true odds ratio is not equal to 1
95 percent confidence interval:
 0.2965529 1.2393053
sample estimates:
odds ratio
 0.5990351
```

Przykład. (Hanusz, Tarasińska 2006, s. 84) W celu sprawdzenia, czy przy ocenie stanu technicznego pewnego urządzenia można się posługiwać łatwym do wyznaczenia pomiarem, wybrano losowo 100 urządzeń i zanotowano następujące dane:

Tablica 6.3: Dane do przykładu ..

Stan urządzenia	Wartość pomiaru		
	niska	średnia	wysoka
Dobry	37	22	11
Zły	6	9	15

Kod w R:

```
# Hanusz, Tarasińska 2006, s. 84
x <- matrix(c(37, 6, 22, 9, 11, 15), ncol = 3)
x
chisq.test(x)
```

```
fisher.test(x)
# Hanusz 2006, s. 118, zad. 4
x <- matrix(c(25, 10, 50, 65), ncol = 2)
x
chisq.test(x)
fisher.test(x)
```

Realizacja w R:

```
> # Hanusz, Tarasińska 2006, s. 84
> x <- matrix(c(37, 6, 22, 9, 11, 15), ncol = 3)
> x
```

```
     [,1] [,2] [,3]
[1,]   37   22   11
[2,]    6    9   15
```

```
> chisq.test(x)
```

```
    Pearson's Chi-squared test

data:  x
X-squared = 14.781, df = 2, p-value = 0.0006172
```

```
> fisher.test(x)
```

```
    Fisher's Exact Test for Count Data

data:  x
p-value = 0.000676
alternative hypothesis: two.sided
```

```
> # Hanusz 2006, s. 118, zad. 4
> x <- matrix(c(25, 10, 50, 65), ncol = 2)
> x
```

```
     [,1] [,2]
[1,]   25   50
[2,]   10   65
```

```
> chisq.test(x)


    Pearson's Chi-squared test with Yates' continuity correction

data:  x
X-squared = 7.3043, df = 1, p-value = 0.006879
> fisher.test(x)


    Fisher's Exact Test for Count Data

data:  x
p-value = 0.006362
alternative hypothesis: true odds ratio is not equal to 1
95 percent confidence interval:
 1.346986 8.254811
sample estimates:
odds ratio
  3.224463
```

Rozdział 7

Regresja liniowa i wielokrotna

7.1 Regresja liniowa

Mamy dane dwie zmienne (cechy) x i y. Chcemy określić zależność liniową pomiędzy tymi zmiennymi, tzn. wyznaczyć liniowy wpływ zmiennej x na zmienną y. W tym celu wyznaczymy linię prostą zwaną regresją liniową postaci

$y = ax + b$

gdzie

y - zmienna zależna, objaśniana (the response variable)

x - zmienna niezależna, objaśniająca (the predyctor variable)]

a - współczynnik regresji

b - wyraz wolny (intercept).

Miarą dopasowania regresji liniowej do danych jest współczynnik determinacji R^2.

W R wyznaczenie regresji liniowej można wykonać za pomocą funkcji postaci

$$lm(y\ x)$$

$$lub$$

$$lm(y\ x, dane)$$

Przykład (Greń 1975, s. 176) Badając zależność między wielkością produkcji X pewnego wyrobu a zużyciem Y pewnego surowca zużywanego w produkcji tego wyrobu otrzymano dla losowej próby 7 obserwacji następujące wyniki (x_i w tys. sztuk, y_i w tonach):

Tablica 7.1: Dane do przykładu..

sztuki	xi	1	2	3	4	5	6	7
surowiec	yi	8	13	14	17	18	20	22

Wyznaczyć równanie regresji liniowej.

Kod w R:

```
# Greń 1975, s. 176 - regresja liniowa
rm(list=ls()) # usuwanie wszystkich zmiennych z przestrzeni roboczej
library(HH) # aktywowanie pakietu HH
# tworzenie danych
sztuki = c(1, 2, 3, 4, 5, 6, 7)
sztuki
surowiec=c(8, 13, 14, 17, 18, 20, 22)
surowiec
# wykres danych
plot(sztuki, surowiec, main="produkcja")
# model: y = a + bx
# model: surowiec=a+b*sztuki
# (Intercept) = a, sztuki = b
# wyznaczanie równania regresji liniowej
model1=lm(surowiec~sztuki)
summary(model1)
# na wykresie danych wyznaczana jest prosta regresji
abline(model1)
# rysunek danych, równania regresji liniowej oraz przedziały ufności
ci.plot(model1)
```

Realizacja w R:

```
> # Greń 1975, s. 176 - regresja liniowa
> rm(list=ls()) # usuwanie wszystkich zmiennych z przestrzeni roboczej
> library(HH) # aktywowanie pakietu HH
```

```
Warning: package 'HH' was built under R version 3.2.5

Loading required package: lattice

Warning: package 'lattice' was built under R version 3.2.5

Loading required package: grid

Loading required package: latticeExtra

Loading required package: RColorBrewer

Loading required package: multcomp

Warning: package 'multcomp' was built under R version 3.2.5

Loading required package: mvtnorm

Loading required package: survival

Warning: package 'survival' was built under R version 3.2.5

Loading required package: TH.data

Warning: package 'TH.data' was built under R version 3.2.5

Loading required package: MASS


Attaching package: 'TH.data'

Następujący obiekt został zakryty z 'package:MASS':


    geyser

Loading required package: gridExtra

Warning: package 'gridExtra' was built under R version 3.2.4
```

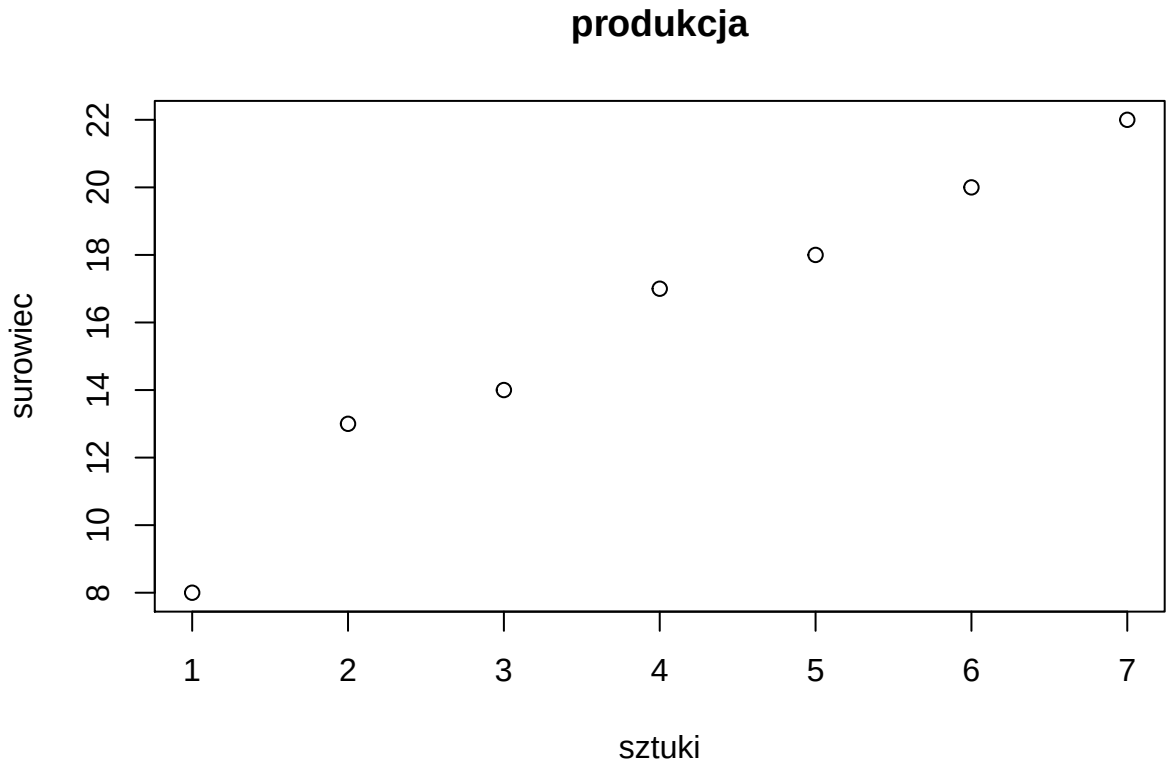
```
> # tworzenie danych
> sztuki = c(1, 2, 3, 4, 5, 6, 7)
> sztuki
```

```
[1] 1 2 3 4 5 6 7
```

```
> surowiec=c(8, 13, 14, 17, 18, 20, 22)
> surowiec
```

```
[1]  8 13 14 17 18 20 22
```

```
> # wykres danych
> plot(sztuki, surowiec, main="produkcja")
```

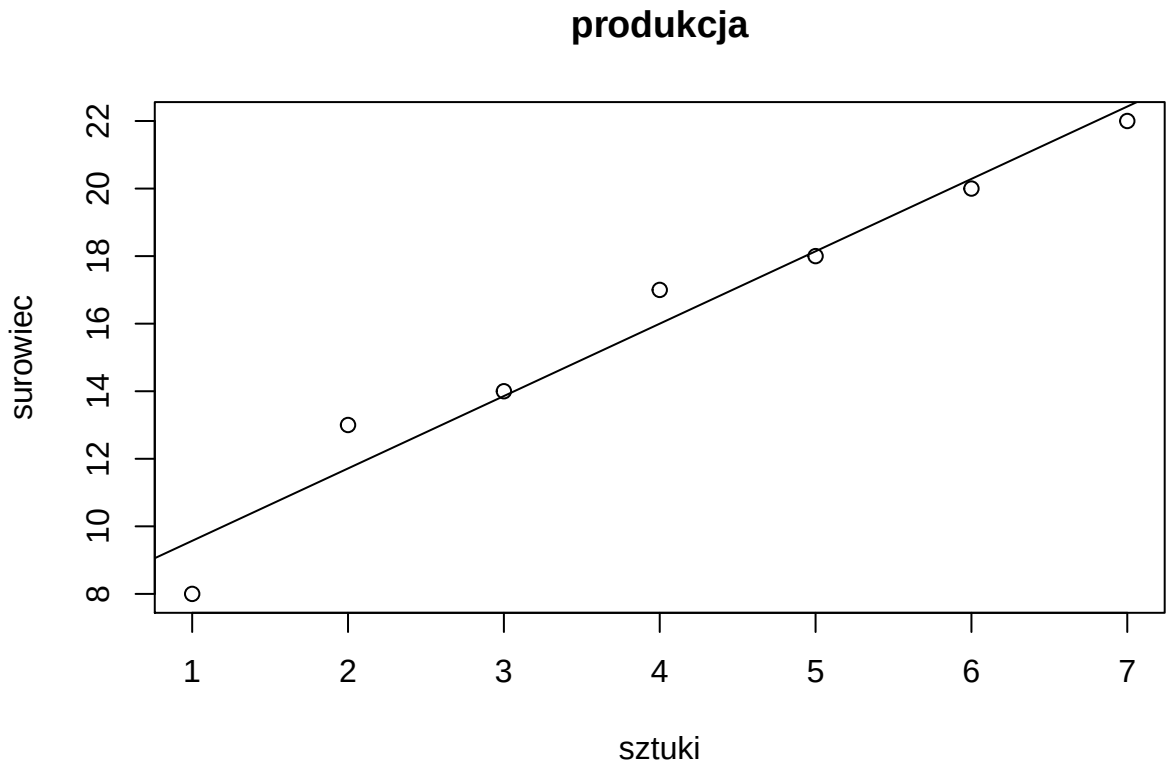



```
> # model: y = a + bx
> # model: surowiec=a+b*sztuki
> # (Intercept) = a, sztuki = b
> # wyznaczanie równania regresji liniowej
> model1=lm(surowiec~sztuki)
> summary(model1)


Call:
lm(formula = surowiec ~ sztuki)

Residuals:
      1       2       3       4       5       6       7
-1.5714  1.2857  0.1429  1.0000 -0.1429 -0.2857 -0.4286

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   7.4286     0.8806   8.436 0.000384 ***
```

```
sztuki        2.1429     0.1969  10.882 0.000114 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.042 on 5 degrees of freedom
Multiple R-squared:  0.9595,    Adjusted R-squared:  0.9514
F-statistic: 118.4 on 1 and 5 DF,  p-value: 0.0001138
```

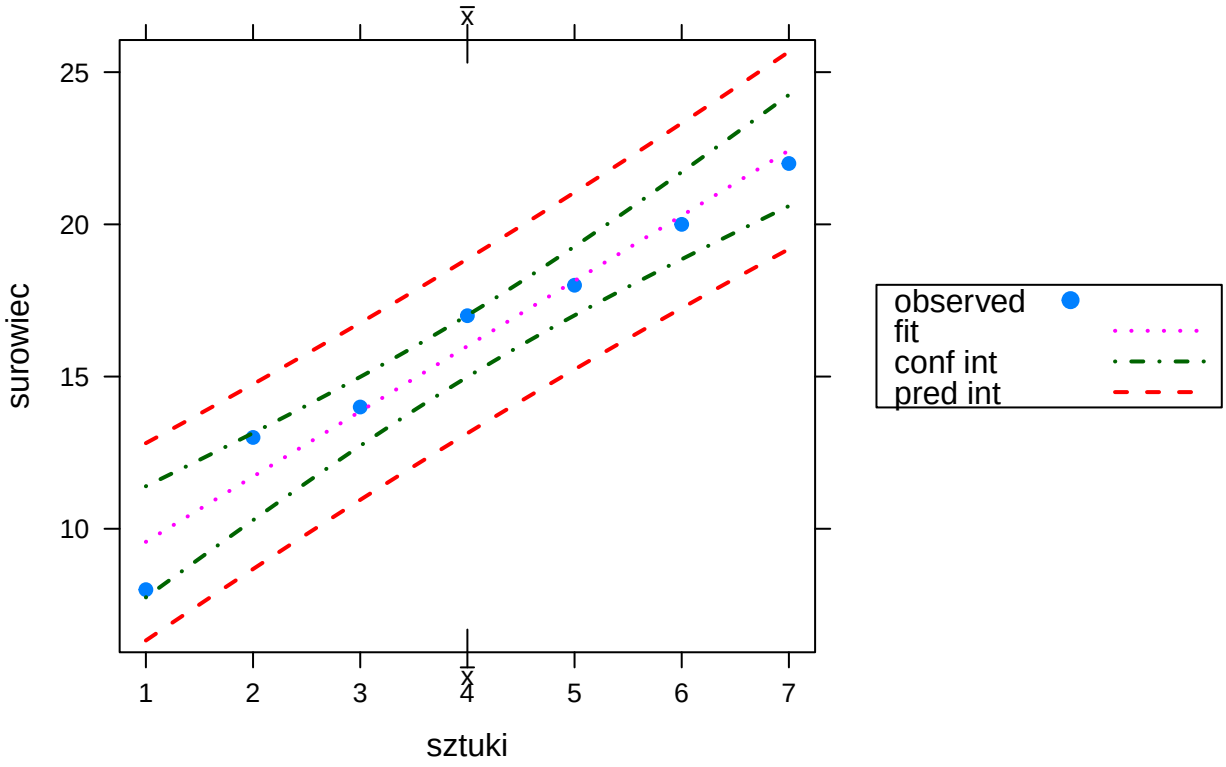
Decyzje: Ponieważ p-wartości dla wyrazu wolnego a (Intercept) oraz dla współczynnika kierunkowego b są mniejsze od 0,05 więc odrzucamy hipotezy zerowe i przyjmujemy hipotezy alternatywne.

Interpretacja: Wyraz wolny a (Intercept) oraz współczynnik kierunkowy b są istotne dla równania regresji liniowej $y = a + bx$.




```
> abline(model1)
```

```
> # rysunek danych, równania regresji liniowej oraz przedziały ufności
> ci.plot(model1)
```



Równanie regresji liniowej jest postaci:

surowiec = 7.4286 + 2.1429 * sztuki

7.2 Regresja wielokrotna

Mamy dane cechy $x_1, x_2, ..., x_n iy$. Chcemy określić zależność liniową pomiędzy zmienną y a zmiennymi $x_1, x_2, ..., x_n$, tzn. wyznaczyć liniowy wpływ zmiennych $x_1, x_2, ..., x_n$ na zmienną y. W tym celu wyznaczymy regresją wielokrotną postaci $y = b_0 + b_1x_1 + b_2x_2 + b_3x_3 + … + b_nx_n $ gdzie y - zmienna zależna, objaśniana (the response variable), $x_1, x_2, ..., x_n$ - zmienne niezależne, objaśniające (the predyctor variables), b_0 - wyraz wolny (intercept), $b_1, ..., b_n$ - współczynniki regresji.

Miarą dopasowania regresji wielokrotnej do danych jest współczynnik determinacji R^2.

W R wyznaczenie regresji wielokrotnej, tak jak regresji liniowej, można wykonać za pomocą funkcji postaci

$$lm(y\ x1 + x2 + ... + xn)$$

$$lub$$

$$lm(y\ x1 + x2 + ... + xn, dane)$$

Przykład (Elandt 1964, s. 441)

Badano cztery cechy słomy konopi: ciężar włókna (g), długość łodygi (cm), grubość ło-dygi (mm) oraz ciężar łodygi (g). Znaleźć równanie regresji wielokrotnej liniowej określają-cej zależność ciężaru włókna od długości, grubości oraz ciężaru łodygi.

Tablica 7.2: Ciężar włókna (g), długość łodygi (cm), grubość łodygi (mm) oraz ciężar łodygi (g) konopi.

Lp.	ciężar włókna y	długość łodygi x1	grubość łodygi x2	ciężar łodygi x3	Lp.	ciężar włókna y	długość łodygi x1	g
1	7,4	251	9,25	47,5	26	8,3	248	
2	9,2	255	10,50	57,7	27	8,5	248	
3	9,6	253	9,50	47,1	28	8,9	256	
4	6,7	242	8,50	38,8	29	6,7	246	
5	7,8	246	9,50	45,2	30	7,6	247	
6	7,8	246	10,25	49,8	31	4,6	242	
7	6,3	243	8,75	43,4	32	6,2	247	
8	7,6	246	9,00	50,8	33	7,0	250	
9	6,4	249	9,00	41,5	34	8,9	280	
10	7,0	247	9,50	38,8	35	6,9	240	
11	6,6	237	9,75	47,1	36	8,7	243	
12	8,2	246	9,50	51,2	37	8,5	229	
13	8,2	257	9,50	52,6	38	10,4	271	
14	7,0	250	8,75	46,1	39	8,5	266	
15	6,8	235	8,00	36,0	40	9,8	267	
16	6,8	247	10,00	44,8	41	7,8	260	
17	9,7	234	9,50	47,1	42	7,3	247	
18	9,3	259	10,50	68,3	43	7,0	242	
19	12,0	255	10,25	62,8	44	9,8	254	
20	8,4	264	8,50	45,7	45	8,9	262	
21	9,5	261	10,75	60,9	46	10,2	260	
22	9,0	242	9,50	45,2	47	8,7	254	
23	6,8	240	8,25	37,8	48	6,8	249	
24	7,3	235	10,25	48,0	49	7,5	244	
25	7,0	245	8,75	44,3				

Kod w R:

```
# Elandt 1964, s.441 - regresja liniowa wielokrotna
rm(list=ls()) # usuwanie wszystkich zmiennych z przestrzeni roboczej
setwd("D://abc") # ustanowienie aktualnego folderu
# tworzenie danych
sloma=read.table("Elandt-441-regresja-wielokrotna.txt", header=T)
sloma
head(sloma)
# korelacje cząstkowe
round(cor(sloma),2)
# regresja liniowa wielokrotna
regresja<-lm(ciezwlokna~dluglodygi+grublodygi+ciezlodygi, data=sloma)
summary(regresja)
# ANOVA dla regresji
anova(regresja)
```

Realizacja w R:

```
> # Elandt 1964, s.441 - regresja liniowa wielokrotna
> rm(list=ls()) # usuwanie wszystkich zmiennych z przestrzeni roboczej
> setwd("D://abc") # ustanowienie aktualnego folderu
> # tworzenie danych
> sloma=read.table("Elandt-441-regresja-wielokrotna.txt", header=T)
> sloma
> head(sloma)
> # korelacje cząstkowe
> round(cor(sloma),2)
> # regresja liniowa wielokrotna
> regresja<-lm(ciezwlokna~dluglodygi+grublodygi+ciezlodygi, data=sloma)
> summary(regresja)
> # ANOVA dla regresji
> anova(regresja)
```

Bibliografia