Wprowadzenie do środowiska R, statystyka opisowa

Bartosz Kozak

28 marca 2019

Wprowadzenie do środowiska R

Więcej informacji na temat R można znaleźć pod tymi linkami:

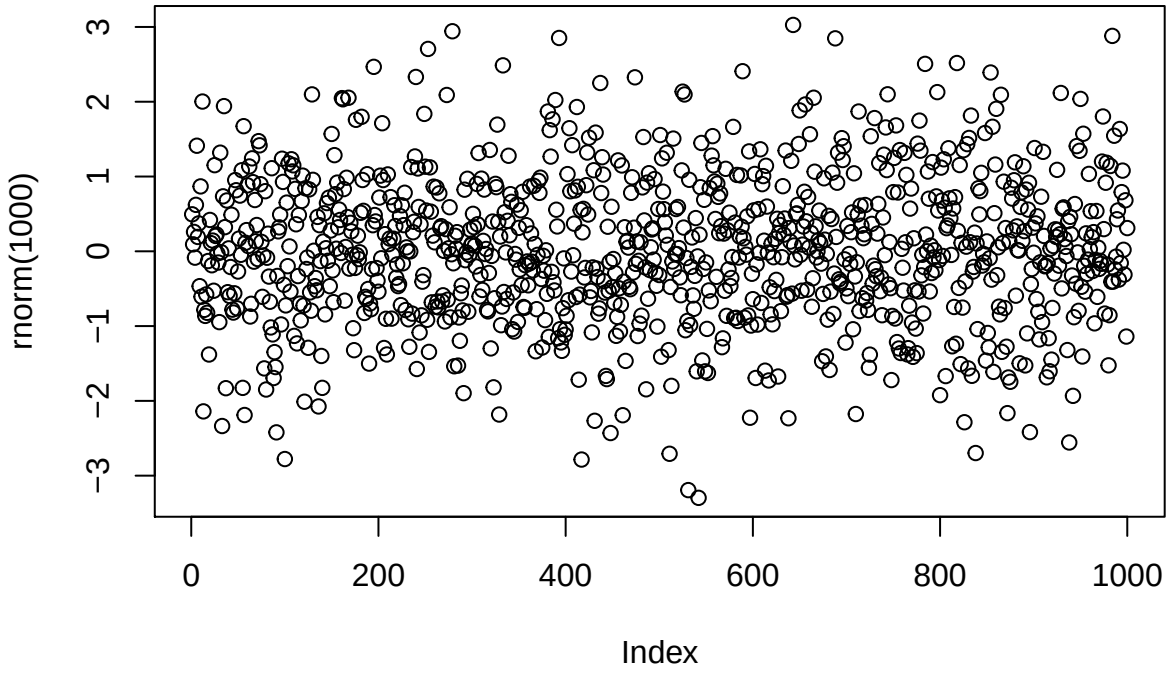
- Wikipedia
- Przewodnik po pakiecie R

Program R działa na zasadzie "pytanie-odpowiedź". Wprowadzamy dane lub polecenie do konsoli, a program wykonuje operacje i wyświetla wynik oraz czeka na kolejne instrukcje. Znakiem zachęty (prompt) do wprowadzania kolejnych instrukcji jest symbol >.

Wszystkie prezentowane przykłady powinny działać jeżeli zostaną wpisane DOKŁADNIE tak jak w instrukcjach. Niektóre polecenia R stają się dostępne dopiero po uruchomieniu odpowiednich bibliotek. Należy zawsze upewnić się, że wymagane biblioteki są zainstalowane i załadowane. Do ładowania bibliotek służy polecenie `library`.

Zobaczmy jak działa R w praktyce.

```
plot(rnorm(1000))
```



"Przerośnięty" kalkulator

Jednym z najprostszych zastosowań programu R jest wykorzystanie go do obliczeń arytmetycznych. Jeżeli wprowadzimy działanie komputer wyświetli nam wynik działania np.

```
2+3
```

```
## [1] 5
```

```
11-5
```

```
## [1] 6
```

```
2*4
```

```
## [1] 8
```

```
33/11
```

```
## [1] 3
```

R pozwala także na wykonanie bardziej skomplikowanych obliczeń np. e^{-2}

```
exp(-2)
```

```
## [1] 0.1353353
```

Symbol `[1]` wyświetlany przed wynikiem jest 'Rową' konwencją wyświetlania liczb oraz wektorów. Tutaj nie wydaje się on szczególnie przydatny. Jest on jednak bardzo użyteczny przy większej liczbie wyników. Liczba w nawiasie kwadratowym jest indeksem liczby w danej linijce. Zobaczmy jak to działa na przykładzie. Wygenerujmy 15 losowych liczb z rozkładu normalnego $\mu = 0, \sigma = 1$.

```
set.seed(123)
rnorm(15)
```

```
##  [1] -0.56047565 -0.23017749  1.55870831  0.07050839  0.12928774
##  [6]  1.71506499  0.46091621 -1.26506123 -0.68685285 -0.44566197
## [11]  1.22408180  0.35981383  0.40077145  0.11068272 -0.55584113
```

Przypisywanie zmiennych

Nawet korzystając z prostego kalkulatora często spotykamy się z potrzebą zapisania cząstkowych lub pośrednich wyników obliczeń, tak aby nie trzeba było ich ponownie obliczać.

R podobnie jak inne języki porgramowania pozwala na zapisywanie informacji w pamięci podręcznej komputera jako tak zwane "*zmienne symboliczne*". Oznacza to, że możemy wartościom (np. liczbowym) przypisać nazwy, a następnie ponownie użyć tych wartości przy pomocy stworzonej nazwy. Aby przypisać wartość wartość do zmiennej w języku R używa się symbolu `<-`. Przykładowo jeżeli chcemy stworzyć zmienną `x` i przypisać jej wartość 2 wykonujemy:

```
x <- 2
```

Utworzoną zmienną możemy następnie przywołać:

```
x
```

```
## [1] 2
```

Lub użyć w dalszych obliczeniach:

```
2*x + 3
```

```
## [1] 7
```

Nazwy zmiennych mogą być w R tworzone dość swobodnie. Możemy stosować litery, liczby oraz symbol kropki (.) oraz podkreślenia (_). Istnieją jednak pewne ograniczenia, nazwa nie może rozpoczynać się od liczby, lub od kropki, a po niej liczby. Typowa nazwa zmiennej w R może wyglądać tak: `wysokość.1rok`. Nazwy zmiennych w R są wrażliwe na wielkość liter, czyli `WT` i `wt` dla R oznaczać będą dwie różne zmienne. Niektóre zmienne są zarezerwowane dla samego R. Zaleca się niestosowanie tych nazw do nazywania zmiennych. Zmienne jednoliterowe wykorzystywane przez R to: `c,q,t,C,D,F,I` oraz `T`. Inne nazwy wykorzystywane przez R to:

`diff`, `df`, oraz `pt`. Większość z nich to nazwy funkcji. Użycie tych nazw do nazwania własnych zmiennych nie zawsze będzie prowadziło do problemów, jednak zaleca się nie używanie tych nazw. Przykładowo `F` oraz `T` są skrótami słów kluczowych `True` oraz `False`. Użycie `F` i/lub `T` spowoduje, że te skróty nie będą działać poprawnie.

Arytmetyka wektorowa

Nie możemy wykonać zbyt wielu analiz statystycznych na pojedynczej liczbie! Analizy statystyczne wykonywać będziemy na danych pochodzących z grupy pomiarów (np. grupa pacjentów). Jedną z zalet R jest to, że może przechowywać cały wektor danych (np. pomiarów) jako pojedynczy obiekt. Wektor jest prostym szeregiem (ciągiem) liczb. Wektor w R tworzymy następująco:

```
zaw.Hg <- c(20,21,22,23,27)
zaw.Hg
```

```
## [1] 20 21 22 23 27
```

Nie jest to jedyna droga wprowadzania danych do R, nie jest to też preferowana metoda, jednak krótkie wektory są wykorzystywane powszechnie w R do różnych zastosowań. W późniejszej części przedstawione zostaną inne metody wczytywania danych do programu R.

Na wektorach możliwe jest wykonywanie operacji arytmetycznych, tak jak na "zwykłych" liczbach. Jeżeli wykonujemy operacje na dwóch wektorach muszą mieć one tą samą długość. Zobaczmy to na przykładzie. Załóżmy że mamy wektor z wysokością i masą ciała 6 osób. Możemy utworzyć nowy wektor z wartością współczynnika BMI, który zdefiniowany jest jako stosunek wagi (kg) do kwadratu wzrostu (m). Przykładowo:

```
wzrost <- c(1.75, 1.80, 1.65, 1.90, 1.74, 1.91)
waga <- c(60, 72, 57, 90, 95, 72)
BMI <- waga/wzrost^2
BMI
```

```
## [1] 19.59184 22.22222 20.93664 24.93075 31.37799 19.73630
```

Zauważmy, że obliczenia wykonywane są na liczbach o tym samym indeksie w obydwu wektorach, to jest BMI `19.59184` został obliczony jako $60/1.75^2$.

W istocie R pozwala na operacje arytmetyczne wektorów o różnej długości. W powyższym przykładzie wykonaliśmy potęgowania wektora wzrost (długość 6) przez wektor `2` (długość 1). Jeżeli wektory są różnej długości do elementy wektora krótszego są iterowane (powtarzane) na elementach wektora dłuższego. Zazwyczaj operacje na wektorach różnej długości wykonuje się przy modyfikacji wektora przez skalar (liczbę) przykładowo:

```
waga.g <- waga * 1000
waga.g
```

```
## [1] 60000 72000 57000 90000 95000 72000
```

Arytmetyka wektorowa jest bardzo użyteczna w przypadku obliczeń statystycznych. Przypuśćmy, że chcemy obliczyć wartość średnią oraz odchylenie standardowe dla wartości w wektorze `waga`.

W pierwszej kolejności obliczymy wartość średnią, $\overline{x} = \frac{\sum x_i}{n}$

```
sum(waga)
```

```
## [1] 446
```

```
sum(waga)/length(waga)
```

```
## [1] 74.33333
```

Następnie zapiszmy średnią wartość wektora waga jako zmienna `xbar` i przejdźmy do obliczenia odchylenia standardowego, $SD = \sqrt{(\sum(x_i - \overline{x})^2)/(n-1)}$. Zrobimy to w kilku krokach, żeby zobaczyć obliczenia cząstkowe.

```
# zapisujemy średnią pod nazwą xbar
xbar <- sum(waga)/length(waga)
# obliczamy różnice między każdą wartością, a wartością średnią
waga - xbar
```

```
## [1] -14.333333  -2.333333 -17.333333  15.666667  20.666667  -2.333333
```

```
# obliczamy kwadrat tych różnic
(waga - xbar)^2
```

```
## [1] 205.444444   5.444444 300.444444 245.444444 427.111111   5.444444
```

```
# obliczamy sumę kwadratów (SS)
sum((waga - xbar)^2)
```

```
## [1] 1189.333
```

```
# obliczamy odchylenie standardowe
sqrt(sum((waga - xbar)^2)/(length(waga)-1))
```

```
## [1] 15.42293
```

Oczywiście R jako pakiet statystyczny ma już wbudowane funkcje służące do obliczania wartości średniej i odchylenia standardowego:

```
mean(waga)
```

```
## [1] 74.33333
```

```
sd(waga)
```

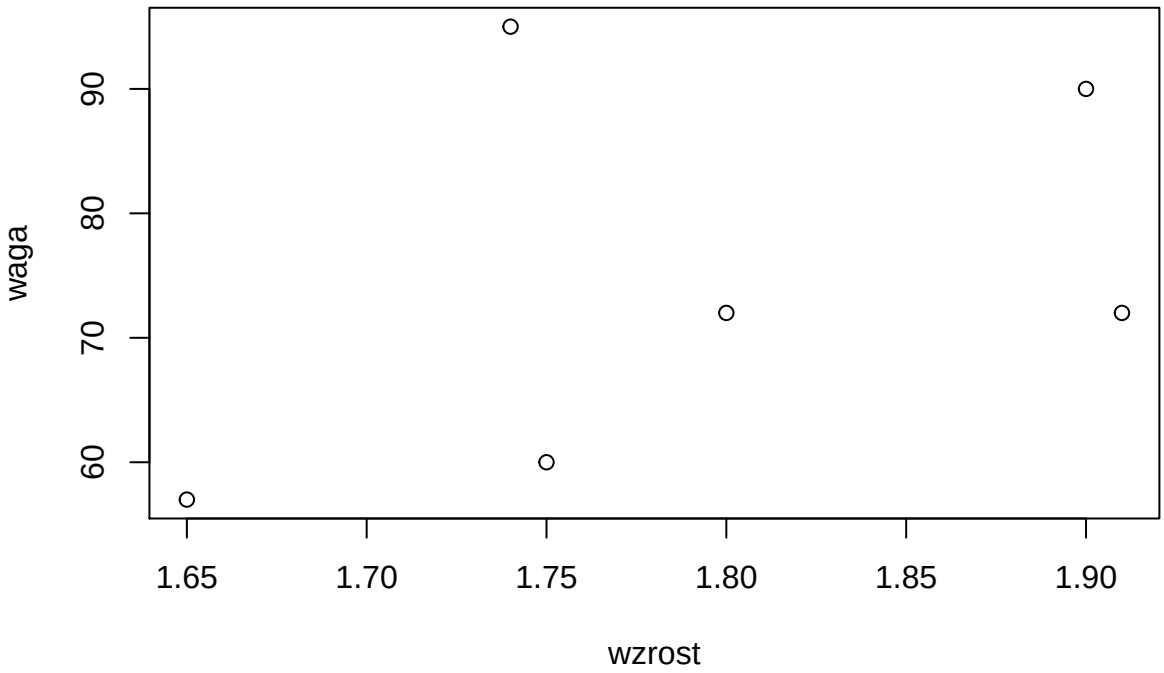
```
## [1] 15.42293
```

Grafika w R

Jednym z najważniejszych aspektów prezentacji i analizy danych jest tworzenie odpowiednich grafik. R podobnie jak S (język z którego R się wywodzi) ma model do tworzenia wykresów, który pozwala na łatwe tworzenie prostych wykresów, a jednocześnie pozwala na dużą kontrolę nad generowaną grafiką.

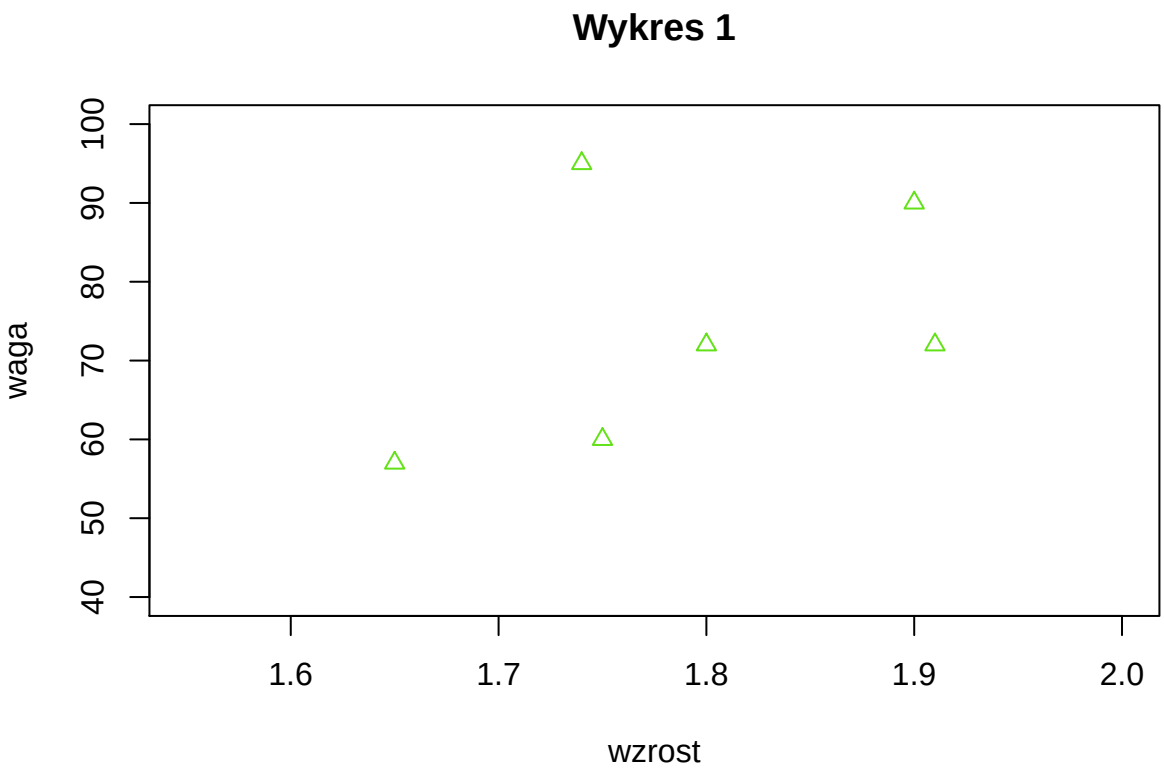
Jeżeli chcemy zbadać zależność między wagą i wzrostem, pierwszym krokiem powinno być stworzenie wykresu dla tych zmiennych:

```
plot(wzrost,waga)
```



Wygenerowany wykres może być modyfikowany na wiele sposobów, poprzez dodanie odpowiednich argumentów do funkcji `plot`. Zmodyfikujmy nasz wykres dodając tytuł, oraz zmieniając kształt symboli punktów, a także ich kolor oraz zakres na osi x i y:

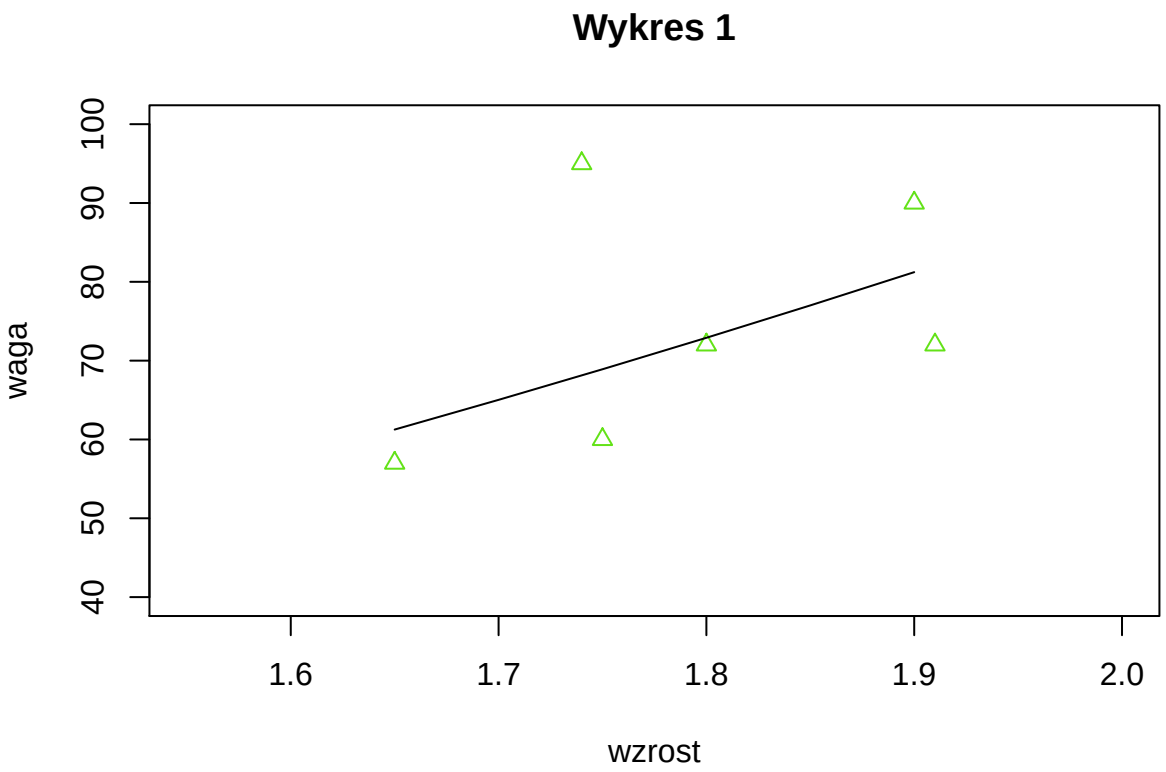
```
plot(wzrost,waga, main="Wykres 1", pch = 2, col = '#63e218', xlim = c(1.55,2), ylim = c(40,100))
```



Idea za obliczaniem indeksu BMI jest taka, aby można było określić czy dana osoba jest otyła, na postawie jednej wartości (niezależnie od wzrostu). Ponieważ normalny BMI powinien być w przybliżeniu równy 22.5 możemy oczekiwać, że $waga \approx 22.5 * wzrost^2$. Możemy następnie na tej podstawie dopasować krzywą do

naszego wykresu, określającą oczekiwaną wagę przy danym wzroście (zakładając prawidłowy BMI). Możemy to zrobić następująco:

```
plot(wzrost,waga, main="Wykres 1", pch = 2, col = '#63e218', xlim = c(1.55,2), ylim = c(40,100))
hh <- c(1.65, 1.70, 1.75, 1.80, 1.85, 1.90)
lines(hh, 22.5 * hh^2)
```



Funkcja `line` wrysowuje linie o zadanych koordynatach (x,y) do ostatniego wykresu. W powyższym przykładzie stworzyliśmy nowy wektor `hh` zawierający wartości wzrostu (x), które posłużyły do stworzenia prostej dodanej do wykresu. Użycie tego wektora zamiast surowych danych w wektorze wzrost podyktowane jest dwoma przesłankami:

- wartości w wektorze wzrost nie są równo rozmieszczone
- wartości w wektorze wzrost nie są posortowane

Zastosowanie wektora wzrost do stworzenia krzywej dodanej do wykresu nie pozwoliłoby na wygenerowanie prawidłowej krzywej.

Wczytywanie danych do R

Najłatwiejszą drogą wczytywania danych do R jest poprzez wczytanie danych z pliku tekstowego poprzez funkcję `read.table` lub jej modyfikacje `read.csv` oraz `read.csv2`. Plik tekstowy z danymi musi być zapisany jako "plain text", a nie "rich text" (czyli nie zapisywać danych w formacie .doc, .docx, lub .odt). Program R w założeniu tworzony był (i nadal jest) jako część systemu GNU. GNU jest projektem, który stara się imitować system UNIX, a w systemie UNIX dane zapisywane były jako "proste" pliki tekstowe (zazwyczaj rozdzielone znakiem tabulacji). Do R można wczytywać dane zapisane w arkuszach kalkulacyjnych (MS Excel), jednak wymaga to zastosowania dodatkowych bibliotek i nie jest polecaną (przynajmniej dla początkujących) metodą wczytywania danych do programu R.

Funkcja `read.table`wczytuje dane jako obiekt `data.frame` - tabela i oczekuje, że dane w pliku tekstowym będą miały odpowiedni układ, wartości odpowiadające wierszom będą w poszczególnych liniach (bez pustych linii), a

wartości odpowiadające kolumną będą rozdzielone odpowiednim symbolem (tabulator,spacja,przecinek,średnik, itp.). Pierwsza linijka może zawierać nagłówki kolumn (np. nazwy zmiennych).

Statystyka opisowa i grafika

Analizę statystyczną rozpoczynamy od prostej charakteryzacji danych poprzez ich podsumowanie, a także graficzną prezentację.

Charakterystyka zbioru danych - pojedyncza grupa

Obliczenie podstawowych statystyk dla zbioru danych w programie R jest bardzo proste. Oto jak możemy obliczyć średnią, odchylenie standardowe, wariancję oraz medianę:

```
set.seed(123)
# generujemy 50 losowych liczb
x <- rnorm(50)
# średnia
mean(x)
```

```
## [1] 0.03440355
```

```
# odchylenie standardowe
sd(x)
```

```
## [1] 0.92587
```

```
# wariancja
var(x)
```

```
## [1] 0.8572352
```

```
# mediana
median(x)
```

```
## [1] -0.07264039
```

Kwantyle mogą zostać obliczone za pomocą funkcji:

```
quantile(x)
```

```
##          0%         25%         50%         75%        100%
## -1.96661716 -0.55931702 -0.07264039  0.69817699  2.16895597
```

Jak widać dzięki tej funkcji dostajemy wartość minimalną oraz maksymalną z naszego zbioru, a także kwantyle 0.25, 0.5 i 0.75. Możemy obliczyć różnicę między trzecim a pierwszym kwantylem. Rożnica ta nosi nazwę *interquartile range - rozstęp ćwiartkowy* (IQR) i bywa wykorzystywana jako alternatywa dla odchylenia standardowego.

```
q <- quantile(x)
IQR <- q[4]-q[2]
as.numeric(IQR)
```

```
## [1] 1.257494
```

Jest możliwe uzyskanie kwantyli dla innych (niż 0.25,0.5 i 0.75) wartości. Jeżeli chcemy uzyskać wartości dla *decyli* czyli wartości 0.1,0.2,0.3 itd. w R możemy wykorzystać funkcje `quantile` z odpowiednim parametrem:

```
pvec <- seq(0,1,0.1)
pvec
```

```
##  [1] 0.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0
```

```
quantile(x,pvec)
```

```
##          0%         10%         20%         30%         40%         50%
## -1.96661716 -1.12461142 -0.68842368 -0.46849617 -0.29942796 -0.07264039
##         60%         70%         80%         90%        100%
##  0.23594940  0.51467063  0.82482227  1.22705511  2.16895597
```

Zwróćmy uwagę, że istnieje wiele różnych algorytmów na wyznaczenie kwantyli. R domyślnie używa algorytmu sumy poligonu, gdzie i-ty ranga obserwacji jest równa $(i-1)/(n-1)$ kwantylowi, a pośrednie kwantyle są uzyskiwane poprzez liniową interpolację. Inne definicje kwantyli są możliwe do użycia po zmianie wartości parametru `type` funkcji `quantile`.

Więcej informacji na temat każdej funkcji R możemy uzyskać sprawdzając dokumentację dotyczącą danej funkcji. Dostęp do dokumentacji możliwy jest po wpisaniu symbolu `?`, a następnie nazwy interesującej nas funkcji. Można też skorzystać z okna pomocy programu RStudio.

```
?quantile
```

Kolejnym często wykorzystywanym parametrem w statystyce opisowej jest współczynnik zmienności definiowany jako: $cv = \frac{sd}{\overline{x}}$. Współczynnik zmienności bywa nazywany statystyką kształtu rozkładu. Dla naszego przykładu możemy obliczyć współczynnik zmienności w następujący sposób:

```
cv <- sd(x)/mean(x)
cv
```

```
## [1] 26.91205
```

Jeżeli nasz zbór danych zawiera puste wartości (brak wartości) - `NA`, obliczenie średniej (i innych parametrów opisujących dane) staje się bardziej skomplikowane. R z założenia nie pomija wartości pustych (`NA`). Zilustruje to następujący przykład:

```
x1 <- c(x,NA)
x1
```

```
##  [1] -0.56047565 -0.23017749  1.55870831  0.07050839  0.12928774
##  [6]  1.71506499  0.46091621 -1.26506123 -0.68685285 -0.44566197
## [11]  1.22408180  0.35981383  0.40077145  0.11068272 -0.55584113
## [16]  1.78691314  0.49785048 -1.96661716  0.70135590 -0.47279141
## [21] -1.06782371 -0.21797491 -1.02600445 -0.72889123 -0.62503927
## [26] -1.68669331  0.83778704  0.15337312 -1.13813694  1.25381492
## [31]  0.42646422 -0.29507148  0.89512566  0.87813349  0.82158108
## [36]  0.68864025  0.55391765 -0.06191171 -0.30596266 -0.38047100
## [41] -0.69470698 -0.20791728 -1.26539635  2.16895597  1.20796200
## [46] -1.12310858 -0.40288484 -0.46665535  0.77996512 -0.08336907
## [51]          NA
```

```
mean(x1)
```

```
## [1] NA
```

Wartość `NA` należy rozumieć jako nieznaną wartość. Nie można określić wartości średniej zbioru, który zawiera nieznaną wartość. Dlatego przy próbie obliczenia średniej R zwraca wartość `NA`. Jeżeli chcemy pominąć wartości `NA` przy liczeniu średniej (lub innych parametrów) musimy użyć parametru `na.rm=T`, spowoduje to pominięcie przez R wszystkich wartości `NA` w zbiorze danych:

```
mean(x1, na.rm = T)
```

```
## [1] 0.03440355
```

Niestety funkcja `length` nie rozpoznaje parametru `na.rm`. Jeżeli chcemy otrzymać liczbę elementów w naszym zbiorze danych z pominięciem wartości `NA` możemy to zrobić w następujący sposób:

```
# liczba elementów wektora x
length(x)
```

```
## [1] 50
```

```
# liczba elementów wektora x1
length(x1)
```

```
## [1] 51
```

```
# liczba elementów wektora x1 innych niż NA
sum(!is.na(x1))
```

```
## [1] 50
```

Powyższy kod wykorzystuje właściwość wartości `boolen` (`True` i `False`), które w operacjach matematycznych zamieniane są na 1 (`True`) i 0 (`False`).

Podsumowanie dotyczące zbioru danych możemy też uzyskać korzystając z funkcji `summary`:

```
summary(x)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -1.96662 -0.55932 -0.07264  0.03440  0.69818  2.16896
```

Możemy także uzyskać podsumowanie dla całej tabeli.

W R wbudowanych jest kilka powszechnie znanych i wykorzystywanych w celach edukacyjnych zbiorów danych. Jednym z nich jest zbór 'Irish data set', więcej informacji można na temat tego zbioru można znaleźć tu. Jeżeli chcemy wczytać do pamięci ten zbór danych możemy wykorzystać funkcję `data`.

```
data("iris")
summary(iris)
```

```
##   Sepal.Length    Sepal.Width     Petal.Length    Petal.Width
##  Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100
##  1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300
##  Median :5.800   Median :3.000   Median :4.350   Median :1.300
##  Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199
##  3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800
##  Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
##        Species
##  setosa    :50
##  versicolor:50
##  virginica :50
##
##
##
```

Zwróćmy uwagę, że podsumowanie dla danych numerycznych (kolumny 1:4) i danych kategoryzujących (kolumna 5) wygląda inaczej. Dla danych numerycznych uzyskujemy:

- minimum i maksimum
- 1 i 3 kwantyl
- średnią oraz medianę

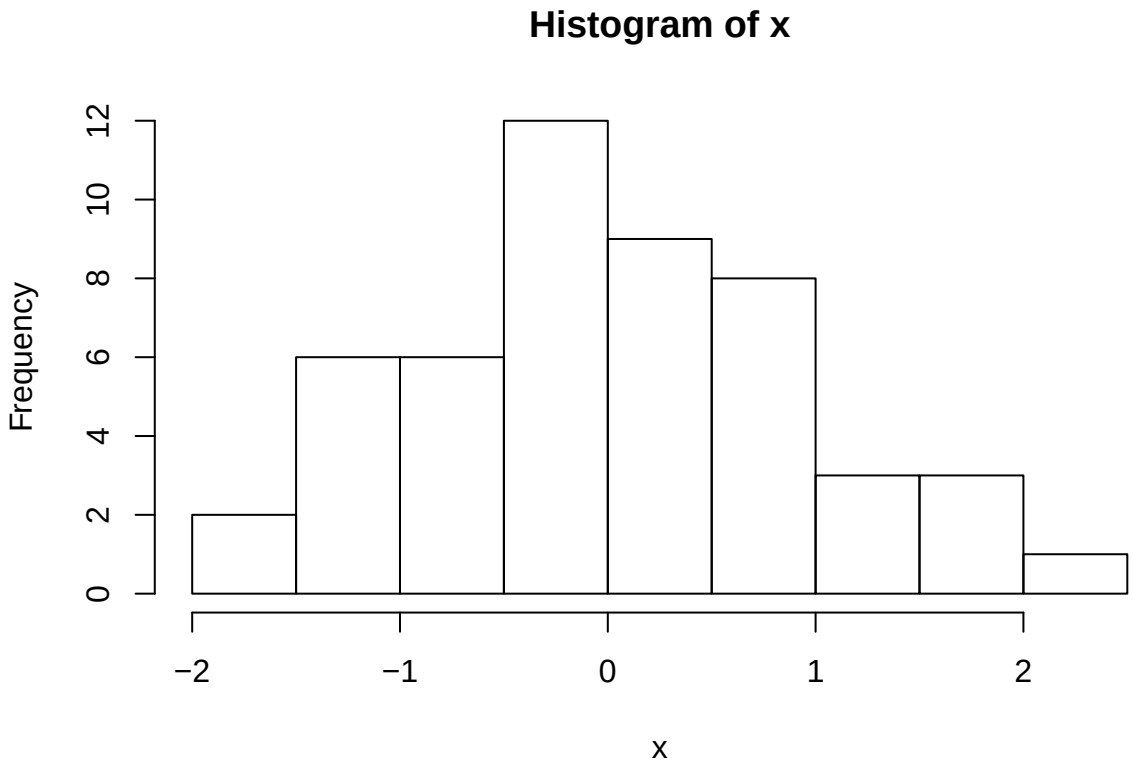
Dla danych kategoryzujących otrzymujemy liczebności poszczególnych kategorii (grup).

Graficzna prezentacja dystrybucji

Histogram

Możemy otrzymać dość dokładny obraz dystrybucji wartości w badanej próbie wykonując histogram, czyli liczbę obserwacji przypadających na dany przedział ("bins"):

```
hist(x)
```



Ćwiczenie 1

Dla danych zamieszczonych w tabeli data.txt wykonać analizę opisową (wyznaczyć średnią, odchylenie standardowe, rozstęp, medianę, decyle, współczynnik zmienności, histogram). *Wskazówka*: Dane w tabeli data.txt zapisane są w 5 kolumnach, ale reprezentują jeden zbiór danych. Przed przystąpieniem do analiz należy połączyć dane z wszystkich w jeden zbiór (wektor) danych.