Kurs R - Tutorial

Agnieszka Strzałka

2021-08-26

Contents

Chapter 1

Wstęp

Książka jest dodatkiem do kursu R i ma stanowić zbiór przykładów, które mogą zostać wykorzystane podczas pisania własnych skryptów R. Materiały będą na bieżąco aktualizowane tak aby wszystkie przedstawione przykłady były możliwe do odtworzenia.

Większość przedstawionych przykładów będzie się opierać na funkcjach zawartych w pakietach tidyverse, które tworzą spójną całość obejmującą wczytanie i analizę danych oraz ich wizualizację. Chociaż wiele z przedstawionych zadań może być wykonane korzystając z podstawowych funkcji R uważam, że nauczenie się od razu korzystania z pakietów tidyverse za bardziej korzystne gdyż zawarte w nich funkcje są często łatwiejsze i bardziej intuicyjne w stosowaniu, szczególnie gdy ktoś nie ma dużego doświadczenia w programowaniu ;)

```
##
## R is the lingua franca of statistical research. Work in all other languages
## should be discouraged.
##    -- Jan de Leeuw (as quoted by Matt Pocernich on R-help)
##       JSM 2003, San Francisco (August 2003)
```

Chapter 2

Skąd ściągnąć R

- Postawowe środowisko R najlepiej ściągnąć bezpośrednio ze strony R project
- Polecam również ściągnąć R Studio, które bardzo ułatwia pracę z R
- Podstawowe pakiety zostaną zainstalowane razem z R, kolejne można pobrać bezpośrednio przez R Studio (zakładka Packages), bardziej bioinformatyczne pakiety znajdują się na stronie bioconductor, wiele pakietów można też pobrac bezpośrednio ze strony Github korzystając z pakietu devtools.

```
##
## Friends don't let friends use Excel for statistics!
##    -- Jonathan D. Cryer (about problems with using Microsoft Excel for
##       statistics)
##       JSM 2001, Atlanta (August 2001)
```

2.1 Pomoc

- Pomoc do funkcji można otworzyć bezpośrednio w R wpisując: `?nazwa_funkcji` albo naciskając F1 podczas wpisywania nazwy funkcji, jednak opisy często są dość enigmatyczne i zakładają już pewne zrozumienie tematu
- Odpowiedzi na konkretne pytania najlepiej szukać w internecie wpisując po prostu: "How do sth R", najczęściej pojawiająca się strona to Stack Overflow
- Ciekawe pomysły i tutoriale pojawiają się też na stronie R-bloggers
- Interaktywny kurs R można też znaleźć na stronie datacamp, ale obecnie tylko pierwsze lekcje każdego kursu są darmowe
- Pomoc do ggplot2

- Polecam też książkę Przemysława Biecka "Przewodnik po pakiecie R", jedna z niewielu jakie są po polsku, pierwsze rozdziały można bezpłatnie pobrać ze strony autora
- Introduction to R for Biologists
- How to make any plot in ggplot2?
- Fundamentals of Data Visualization
- R Graphics Cookbook, 2nd edition
- No i oczywiście ja chętnie pomogę :)

```
##
## You need to get the hang of reading the online help. The information required
## is actually there in ?dotchart --- it's just tersely and obscurely expressed. A
## certain degree of optimism is required. You need to ***believe*** that the
## information is there; then ask yourself "What could they possibly mean by what
## they have written that would tell me what I need to know?".
##    -- Rolf Turner (on reading the help pages)
##       R-help (June 2013)
```

2.2 Packages

Żeby użyć funkcję z danego pakietu, który nie należy do podstawowych należy go najpierw zainstalować (Install w zakładce Packages), a potem załadować. (funkcje `library(nazwa)` albo `require(nazwa)`). Można też włączać pakiety w zakładce Packages.

Dobrze jest ładować tylko te pakiety, które są faktycznie potrzebne - nie przeciążamy pamięci i niektóre nazwy funkcji mogą się powtarzać w różnych pakietach.

Jeżeli ładowanie całego pakietu jest niepotrzebne albo prowadzi do konflktów można odwołać się do konkretnej funckji przy pomocy :: np. `readxl::read_excel()`.

Każdy pakiet zawiera podstawowy opis funkcji, niektóre posiadają bardziej rozbudowane przykłady analiz jakie można przy ich pomocy wykonać - vignette. Jeżeli chcemy sprawdzić które pakiety zawierają winietki można to zrobić funkcją `browseVignettes`.

2.2.1 Lista pakietów, które pojawiają się w książce (uwaga może być niekompletna)

Przed rozpoczęciem należy mieć zainstalowane następujące pakiety: ggplot2, tidyr, knitr, dplyr, readr, readxl oraz najlepiej posiadać najnowszą wersję R (4.1.0 - "Camp Pontanezen") i R Studio (przynajmniej 1.4).

Pozostałe pakiety można pobrać tylko jeśli będą potrzebne.

Inne pakiety jakie pojawiają się to: w części dotyczącej wykresów: car, likert, gplots, VennDiagram, corrplot, hexbin, aplpack, GGally, ggthemes, ggthemr i w części dotyczącej statystyki i obróbki danych: car, MASS, nortest, modeest, moments, agricolae, drc, broom.

2.3 R Studio

Jest to obecnie najpopularniejszy dostępny edytor R, pozwalający na tworzenie projektów, pisanie i zarządzanie skryptami, pozwalający na łatwy dostęp do historii wpisywanych komend i tworzonych wykresów. Został zintegrowany z wieloma przydatnymi pakietami np. knitr, który pozwala tworzenie raportów w języku markdown, latex, prezentacji multimedialnych itp.

R Studio pozwala również na założenie projektu do konkretnego zadania, wszystkie pliki tworzone w trakcie pracy (wykresy, tabele, skrypty) zostaną umieszczone w folderze przypisanym do projektu. Jeżeli nasze dane wejściowe umieścimy w tym samym miejscu nie będzie konieczne podawanie całej ścieżki dostępu przy wczytywaniu pliku. Wykorzystanie projektów ułtwia uporządkowanie pracy. Każdy projekt można też poddać kontroli wersji przy pomocy Git i Github bezpośrednio z Rstudio. Więcej informacji na ten temat można znaleźć w Happy Git with R.

Cały Tutorial to zbiór skryptów napisanych w markdown, które zostały połączone w książkę dzięki pakietowi R bookdown. Wszystkie przykłady można skopiować albo przepisać do własnych skryptów R. Trzeba jedynie pamiętać o wcześniejszym załadowaniu odpowiednich pakietów i ewentualnie danych. Samodzielne wpisywanie nazw funkcji przyśpiesza proces zapamiętywania, a korzystając z autouzpełniania trzeba tak naprawdę pamiętać 2-3 pierwsze litery :)

Chapter 3

Dane

3.1 Wczytanie danych

Dane najłatwiej wczytać z plików .txt lub .csv. Można również z plików .xlsx lub .xls ale często trwa to dłużej i może wymagać zainstalowanych innych pakietów/programów.

W każdym przypadku niezbędne jest podanie ścieżki dostępu do pliku. Jeżeli znajduje się on folderze projektu to wystarczy jego nazwa. Dobrą praktyka jest przechowywanie danych w osobnym katalogu w obrębie projektu - wtedy ścieżka może wyglądać np. data/dane_1.txt. Jeżeli plik z danymi jest przechowywany gdzie indziej konieczne jest podanie pełnej ścieżki dostępu, ale można w tym wypadku korzystać z autouzupełniania klawiszem Tab.

3.1.1 Wczytanie danych z plików tekstowych

R Studio posiada funkcję Import Dataset (zakładka Environment), która pozwala na wczytanie danych wraz z wyborem podstawowych opcji jak separator, separator dziesiętny, obecność nagłówków. Opcja ta obejmuje podstawowe funkcje R oraz pakiety readr i readxl będące częścią tidyverse. Na początek jest łatwiej korzystać z opcji Load Dataset, ponieważ po jej użyciu zostaje również wygenerowany odpowiedni kod R, który można wkleić do własnych skryptów tak aby te same lub podobne dane wczytać już bez problemu następnym razem.

Można również wczytać dane wpisując komendę (wygodne przy pisaniu skryptów) `read.table` albo `read.csv`. Argumenty: file określa nam plik, który wczytujemy, header - nagłówki, sep - separator np. "/t" to tabulator, dec - separator dziesiętny (domyślnie kropka), quote - obecność "". W przypadku nagłówków kolumn wpisanie `header=TRUE` spowoduje, że pierwszy

wiersz naszej tabeli zostanie potraktowany jako tytuły kolumn, `header=FALSE` oznacza domyślne nazwy kolumn - V1, V2, …

W pakiecie readr znajdują się analogiczne funckje: * read_delim - do plików tekstowych, funckja spróbuje zgadnąć jak rozdzielone są kolumny, można podać argumentem delim * read_csv i read_csv2 - do plików csv, odpowiednio do danych wykorzystujących kropki i przecinki jako separatory dziesiętne * read_tsv i read_table - do plików gdzie kolumny są rodzielone przy pomocy tab.

Wczytane dane będą widoczne w zakładce Environment. Kliknięcie na nazwę tabeli spowoduje otwarcie widoku danych podobnego do arkusza kalkulacyjnego. Można w nim dane sortować i filtrować, ale nie edytować.

Pierwsze albo ostanie wiersze można zobaczyć używając funkcji `head` albo `tail`. Strukturę danych pokaże funkcja `str`, a podstawowe informacje `summary`. Dobrze jest po wczytaniu danych sprawdzić czy wyglądają faktycznie tak jak miały ;)

```
dane1 <- read.table(file = "data/dane1.txt", header=TRUE, quote="\"")

head(dane1)
```

```
##     pomiar Szczep warunki
## 1 3.389738      A       1
## 2 5.992065      A       1
## 3 4.464553      A       1
## 4 4.546082      A       1
## 5 6.521751      A       1
## 6 5.713088      A       1
```

```
str(dane1)
```

```
## 'data.frame':    1800 obs. of  3 variables:
##  $ pomiar : num  3.39 5.99 4.46 4.55 6.52 ...
##  $ Szczep : chr  "A" "A" "A" "A" ...
##  $ warunki: int  1 1 1 1 1 1 1 1 1 ...
```

```
summary(dane1)
```

```
##      pomiar          Szczep             warunki
##  Min.   : 1.434   Length:1800        Min.   :1.0
##  1st Qu.: 4.450   Class :character   1st Qu.:1.0
##  Median : 5.478   Mode  :character   Median :1.5
##  Mean   : 5.846                      Mean   :1.5
##  3rd Qu.: 6.743                      3rd Qu.:2.0
##  Max.   :27.953                      Max.   :2.0
```

3.1.2 Wczytanie danych z plików excel

Do wczytania danych z plików xlsx można wykorzystać funkcję read_excel z pakietu readxl. Przy pracy z excelem należy jednak pamiętać żeby wczytywane dane nie zawierały formuł albo połączonych komórek oraz, że funkcja wczyta dane zawarte tylko w jednym arkuszu (sheet), ale można wybrać z której przy pomocy argumentu sheet.

```
library(readxl)

data_excel <- read_excel('data/test.xlsx')

data_excel
```

```
## # A tibble: 16 x 3
##        A     B     C
##    <dbl> <dbl> <dbl>
##  1     1     2     3
##  2     1     3     4
##  3     1     4     5
##  4     1     5     6
##  5     1     6     7
##  6     1     7     8
##  7     1     8     9
##  8     1     9    10
##  9     1    10    11
## 10     1    11    12
## 11     1    12    13
## 12     1    13    14
## 13     1    14    15
## 14     1    15    16
## 15     1    16    17
## 16     1    17    18
```

3.2 Zapisanie danych

Zapisać tabelę danych można korzystając z funkcji `write.table`, wystarczy określić, co chcemy zapisać i nazwę pliku wyjściowego. Możemy zapisywać pliki w formacie np. txt, csv. Jeżeli nie określimy ścieżki dostępu plik zostanie zapisany w folderze projektu.

Można również dopisywać dane do istniejącego pliku ustawiając parametr `append = TRUE`. Domyślnie ten argument jest zawsze ustawiony jako `append =`

`FALSE` i jeżeli w nazwie pliku podamy istniejący plik to **zostanie on nadpisany bez ostrzeżenia**.

Można też dane zapisać w formacie rds. Można je potem otworzyć tylko w R, ale można w takim formacie zapisać każdy obiekt R, nie tylko tabele. Do zapisania danych służy funkcja saveRDS, a do wczytanie readRDS.

```
write.table(dane1, "data/dane4.txt")
```

3.3 Podstawowe typy danych w R

W R dane do zmiennej przypisujemy korzystając ze strzałki ;) `<-` albo `->`. Można też użyć `=`

Do najbardziej podstawowych typów danych należą: typ liczbowy, znakowy, logiczny i czynnikowy (o tym później). Typ danych można sprawdzić funkcją `class`

Najczęściej wykorzystywane rodzaje danych to wektor i ramka danych. Pozostałe to m.in. macierz i lista. W R można również pisać własne funkcje.

3.3.1 Wektor (vector)

Wektor to ciąg elementów tego samego typu np. liczbowy, znakowy. W R nawet pojedyncza cyfra jest wektorem. Wektor można stworzyć korzystając z funkcji:

- `c`. Sekwencję liczb można zapisać jako `1:10` - utworzy wektor liczb od 1 do 10.
- Przy bardziej skomplikowanych sekwencjach można użyć funkcji `seq`, ustawiamy start, koniec i co ile ma być kolejny element
- funkcję `rep`, podajemy jakie elementy mają zostać powtórzone i ile razy. `each` - powtarzanie każdego elementu, `times` - powtarzanie całego wektora.

```
a <- c(1,2,5,8)
a
```

```
## [1] 1 2 5 8
```

```
b <- c("A", "B", "V")
b
```

```
## [1] "A" "B" "V"
```

```
c <- 1:15
c
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15
```

```
d <- seq(10, 100, by=10)
d
```

```
##  [1]  10  20  30  40  50  60  70  80  90 100
```

```
e <- rep(c(1,2,3), times=3)
e
```

```
## [1] 1 2 3 1 2 3 1 2 3
```

```
e <- rep(c(1,2,3), each=3)
e
```

```
## [1] 1 1 1 2 2 2 3 3 3
```

Do poszczególnych elementów wektora można się odwołać stosując nawiasy: `wektor[x]`, gdzie x to numer elementu. Można się odwołać do kilku elementów jednocześnie np. `wektor[1:3]`, `wektor[c(2,4)]`. Numeracja rozpoczyna się od 1. Elementy wektora mogą również mieć swoje nazwy i wtedy można je także wykorzystać do wyświetlania danych.

Ilość elementów wektora podaje funkcja `length`.

```
a <- c(1:5)
a[2]
```

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

```
b <- c( "jeden" = 1, "dwa" = 2, "trzy" = 3)
b[2]
```

```
## dwa
##   2
```

```
b["dwa"]
```

```
## dwa
##   2
```

```
length(a)
```

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

Na wektorach można wykonywać wszystkie operacje matematyczne. Przykładowo jeżeli dodamy do siebie dwa wektory to zawsze pierwszy element zostanie dodany do pierwszego elementu z drugiego wektora itd.

```
1:10 + 11:20
```

```
##  [1] 12 14 16 18 20 22 24 26 28 30
```

```
1:10 * 11:20
```

```
##  [1]  11  24  39  56  75  96 119 144 171 200
```

3.3.2 Ramka danych (data frame)

Ramka danych to po prostu kilka wektorów ułożonych w tabelę. Wektory mogą być różnego typu, powinny być tej samej długości. Jeżeli mają różną długość można uzupełnić brakujące elementy stosując NA - brak danych

Skoro każda kolumna ramki danych jest wektorem to można stosować na nich te same operacje matematyczne np. dodawać albo dzielić.

Ramkę danych tworzymy przy użyciu funkcji `data.frame`

Jeżeli chcemy coś zmienić w ramkę danych używamy `as.data.frame`

Podstawowe informacje o ramce danych

Funkcja	Opis
`nrow`	liczba wierszy
`ncol`	liczba kolumn
`colnames`	nazwy kolumn
`rownames`	nazwy wierszy
`dim`	wymiary

```
x <- data.frame(a=1:5, b=rep("A",5), c=c(T,T,T,F,NA))
x
```

```
##   a b     c
## 1 1 A  TRUE
## 2 2 A  TRUE
## 3 3 A  TRUE
## 4 4 A FALSE
## 5 5 A    NA
```

```
colnames(x)
```

```
## [1] "a" "b" "c"
```

Do elementów ramki danych również można się odwołać przy pomocy nawiasów: `ramka[x,y]`, gdzie x to numer wiersza, a y numer kolumny. Jeżeli podamy jedynie numer kolumny otrzymamy wszystkie jej elementy.

Innym sposobem jest wykorzystanie nazw kolumn i $ np `ramka$kolumna_1`. W ten sposób można łatwo dodać nowe kolumny do ramki danych.

Podobnie możemy usuwać kolumny i wiersze z ramki np. `ramka<-ramka[,-1]` usunie pierwszą kolumnę.

```
ramka <- data.frame(kol1=c(1:10), kol2=c(11:20))

ramka
```

```
##    kol1 kol2
## 1     1   11
## 2     2   12
## 3     3   13
## 4     4   14
## 5     5   15
## 6     6   16
## 7     7   17
## 8     8   18
## 9     9   19
## 10   10   20
```

```
ramka$kol_3 <- ramka$kol1 + ramka$kol2

ramka
```

```
##    kol1 kol2 kol_3
## 1     1   11    12
## 2     2   12    14
## 3     3   13    16
## 4     4   14    18
## 5     5   15    20
## 6     6   16    22
## 7     7   17    24
## 8     8   18    26
## 9     9   19    28
## 10   10   20    30
```

Funkcje zawarte w pakietach tidyverse (w tym read_delim i inne) często zamiast zwykłej ramki danych używają ulepszonej struktury zwanej tibble. W codziennym użytkowaniu nie ma między nimi wielu różnic. Tibble są bezpieczniejsze od zwykłych data.frame, ponieważ nigdy nie zmieniają typów danych w kolumnach. Może się jednak czasem zdarzyć że funkcje starszych pakietów nie będą akceptować tibble zamiast data.frame (można łatwo zmienić przy pomocy `as.data.frame`.

3.3.3 Matryca (matrix)

Matryca jest trochę podobna do ramki danych, ale może zawierać tylko jeden typ danych np. liczbowe. Może mieć więcej wymiarów niż dwa. Tworzymy funkcją `matrix`, zmiana istniejących danych na matrycę - `as.matrix`.

Indeksowanie z matrycach działa tak samo jak w ramkach danych - `[wiersz, kolumna]`.

```
matrix(0, 4, 5)
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    0    0    0    0    0
## [2,]    0    0    0    0    0
## [3,]    0    0    0    0    0
## [4,]    0    0    0    0    0
```

```
matrix(1:15, 3, 5)
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    1    4    7   10   13
## [2,]    2    5    8   11   14
## [3,]    3    6    9   12   15
```

```
(x <- matrix(1:9, 3,3))
```

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

```
x[1,3]
```

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

3.3.4 Lista (list)

Lista to taki rozbudowany wektor ;) i najbardziej elastyczny typ danych w R. Każdy element listy może być innego rodzaju, mieć inną długość, można stworzyć listę, której elementami będzie np. ramka danych, wektor, wykres i nawet inna lista.

Listę tworzymy funkcją `list`, podobnie jak w wektorze każdy element może mieć swoją nazwę. Wiele funkcji jako swój wynik zwraca listę, więc często przydaje się wiedza jak dostać się do poszczególnych elementów.

Funkcją `str` możemy wyświetlić podsumowanie wszystkich elementów listy.

Do poszczególnych części można dostać się przez nazwy albo indeksowanie [] albo [[]]. Przy zastosowaniu nawiasów [] uzyskany element nadal będzie częścią listy.

```
lista <- list( ramka = data.frame(a=rnorm(10), b="test"),
               wektor = seq(1,16,by=3),
               tekst = "To jest lista" )

lista
```

```
## $ramka
##              a    b
## 1   0.34463532 test
## 2   0.05200946 test
## 3   1.15809489 test
## 4  -1.50140830 test
## 5  -1.56323482 test
## 6  -1.18813964 test
## 7  -2.05674156 test
## 8  -1.56002519 test
```

```
## 9   0.69512998 test
## 10 -0.31774708 test
##
## $wektor
## [1]  1  4  7 10 13 16
##
## $tekst
## [1] "To jest lista"
```

```
str(lista)
```

```
## List of 3
##  $ ramka :'data.frame':  10 obs. of  2 variables:
##   ..$ a: num [1:10] 0.345 0.052 1.158 -1.501 -1.563 ...
##   ..$ b: chr [1:10] "test" "test" "test" "test" ...
##  $ wektor: num [1:6] 1 4 7 10 13 16
##  $ tekst : chr "To jest lista"
```

```
lista$tekst
```

```
## [1] "To jest lista"
```

```
lista[3]
```

```
## $tekst
## [1] "To jest lista"
```

```
lista[[3]]
```

```
## [1] "To jest lista"
```

```
# pierwszy element wektora, będącego drugim elementem listy
lista[[2]][1]
```

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

3.3.5 Typ liczbowy (numeric), znakowy (character), logiczny (logical)

- Typ liczbowy przechowuje liczby całkowite i rzeczywiste. Kropką dziesiętną jest kropka :). Na liczbach można łatwo prowadzić podstawowe działania matematyczne - dodawanie, odejmowanie, mnożenie itp. Takie same operacje możemy prowadzić na wektorach.

```
2*4
```

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

```
A <- c(2,4,6)
B <- c(1,2,3)

A*B
```

```
## [1]  2  8 18
```

- Typ znakowy zawiera napisy umieszczone pomiędzy "" albo ''. Napisy można wyświetleć np. funkcją `cat`, możemy wymusić pisanie kolejnego elementu w nowej linii poprzez `"\n"`. Sklejanie np. tekstu i liczb można wykonać funkcją `paste`.

Do pracy z typem znakowym można wykorzystać pakiet stringr.

```
" To jest napis "
```

```
## [1] " To jest napis "
```

```
cat("coś", "tam")
```

```
## coś tam
```

```
cat(" coś","\n","tam")
```

```
##  coś
##  tam
```

```
a <- 1

cat("Wynik to", a)
```

```
## Wynik to 1
```

```
paste("Wynik równa się", a)
```

```
## [1] "Wynik równa się 1"
```

- Typ logiczny przechowuje tylko wartości: TRUE (T) i FALSE (F) oraz NA (brak danych). Nazwy TRUE i FALSE są w R zastrzeżone, ale T i F już nie. Dlatego lepiej używać pełnych nazw, bo może się zdarzyć, że do T albo F zostanie przypisana jakaś zmienna.

```
wektor <- c(TRUE, FALSE, TRUE)
wektor
```

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

```
summary(wektor)
```

```
##    Mode   FALSE    TRUE
## logical       1       2
```

3.3.6 Typ czynnikowy (factor)

Służy do przechowywania wartości występujących w kilku kategoriach np. płeć, wykształcenie itp. Podczas wczytywania danych R z wykorzystaniem podstawowych funckji, ale nie z pakietu readr, automatycznie zamieni kolumny zawierające tekst na typ czynnikowy, chyba że zaznaczymy opcję `stringAsFactors=FALSE`.

Typ czynnikowy jest przydatny podczas robienia wykresów, gdy chcemy pogrupować dane pod względem jakiejś kategorii. Również legendy (kolejność elementów) są tworzone w oparciu o poziomy czynnika. Może się zdarzyć, że kolejność wybrana przez R (często alfabetyczna) nie będzie odpowiednia. Można łatwo przestawić poziomy korzystając z funkcji `factor`. W tej samej funkcji argument labels pozwala na zmianę nazw kolejnych poziomów.

Jeżeli chcemy zmienić kolejność elementów na wykresie np. boxplotów, słupków najlepiej zrobić to zmieniając poziomy faktora w danych.

Poziomy factor można wyświetlić przy pomocy funkcji `levels`.

Do pracy z typem czynnikowym można wykorzystać pakiet forcats będący częścią tidyverse.

```
faktor <- factor(c("A", "B", "C", "A", "A","C"))

faktor
```

```
## [1] A B C A A C
## Levels: A B C
```

```
str(faktor)
```

```
##  Factor w/ 3 levels "A","B","C": 1 2 3 1 1 3
```

```
levels(faktor)
```

```
## [1] "A" "B" "C"
```

```
summary(faktor)
```

```
## A B C
## 3 1 2
```

```
# Zmiana poziomów

faktor2 <- factor(faktor, levels=c("B", "C", "A"), labels = c("BB", "CC", "AA"))
faktor2
```

```
## [1] AA BB CC AA AA CC
## Levels: BB CC AA
```

3.3.7 Funkcje

Większość operacji na danych w R wykonujemy za pomocą funkcji. Pakiety są to właściwie zbiory funkcji, często dotyczących jednego konkretnego zagadnienia.

Każda funkcja zawiera przynajmniej jeden argument, który jest niezbędny do jej działania. Nazwy wszystkich argumentóœ możemy sprawdzić korzystając z pomocy. Często nie jest konieczne podanie wartości wszystkich argumentów, gdyż mogą mieć ustawione wartości domyślne.

Jeżeli podajemy wartości argumentóœ możemy je wpisywać do funkcji kolejno (tak jak są wypisane w pomocy) albo korzystając z nazw. Jeżeli nie pamiętamy wszystkich nazw można sobie pomóc przy pomocy klawisza Tab :) Kolejne podawane argumenty należy rozdzielać przecinkami np. `funkcja(a=1, b=2, c="model")`. Do argumentu możemy też podać wektor wartości korzystając z `c` np. `funkcja(a=c(1,2,5))`

Np. funkcja licząca średnią to `mean`. Zawiera trzy argumenty:

- x - oznacza obiekt, z którego ma być policzona średnia
- trim (domyślnie 0) oznacza część obserwacji, która ma zostać przycięta z każdej strony x

- na.rm (domyślnie FALSE) - czy mają być brane pod uwagę wartości NA.

```
?mean
```

```
mean {base}  R Documentation
```

```
Arithmetic Mean
```

```
Description
```

```
Generic function for the (trimmed) arithmetic mean.
```

```
Usage
```

```
mean(x, ...)
```

```
## Default S3 method: mean(x, trim = 0, na.rm = FALSE, ...)
```

```
wek <- c(1,5,9,2,7,3,5,9,2,4)

mean(wek)
```

```
## [1] 4.7
```

```
mean(x=wek)
```

```
## [1] 4.7
```

```
mean(x=wek, trim=0.2, na.rm=TRUE)
```

```
## [1] 4.333333
```

W R możliwe jest pisanie własnych funkcji. Nie jest to trudne, a pozwala na uniknięcie wielokrotnego powtarzanie tego samego kodu. Dużo łatwiej jesy zmienić/poprawić jedną funkcję niż dwadzieścia razy wklejone te same 10 linijek kodu :)

Poniżej krótki przykład tworzenia funkcji. Więcej informacji można znaleźć np. w książce R for Data Science.

Załóżmy że chemy napisać funckję która wczyta dane z pliku txt i doda do tego pliki nową kolumnę będącą sumą dwóch pierwszych.

Kod potrzebny żeby to wykonać:

```
data <- read.table('data/test_funkcja.txt')

data$nowa_kolumna <- data$x + data$y
```

A teraz funckja która wykonuje te same czynności. Nazwa funkcji jest dowolna, chociaż lepiej jest nie nadpisywać już istniejących funkcji, a jej nazwa powinna opisywać to co funkcja będzie robić. W nawiasach () należy podać nazwy argumentów z ewentualnymi wartościami domyślnymi, a w nawiadach {} kod który ma zostać wykonany. Na końcu funkcji warto użyć return() - determinuje jaki wynik zwróci funkcja.

```
dodaj_kolumne <- function(file){

  data <- read.table(file)

  data$nowa_kolumna <- data$x + data$y

  return(data)
}
```

Każdą funkcję po napisaniu należy sprawdzić

```
dodaj_kolumne('data/test_funkcja.txt')
```

```
##     x  y nowa_kolumna
## 1   1 11           12
## 2   2 12           14
## 3   3 13           16
## 4   4 14           18
## 5   5 15           20
## 6   6 16           22
## 7   7 17           24
## 8   8 18           26
## 9   9 19           28
## 10 10 20           30
```

Chapter 4

Wykresy - pakiet ggplot2

```
##
## If anything, there should be a Law: Thou Shalt Not Even Think Of Producing A
## Graph That Looks Like Anything From A Spreadsheet.
##    -- Ted Harding (in a discussion about producing graphics)
##       R-help (August 2007)
```

Do tworzenia wykresów można użyć kilku pakietów:

- **graphics** - podstawowy pakiet graficzny instalowany razem z R, pełna kontrola nad wyglądem wykresu, ale często wymaga to więcej pracy niż w innych pakietach. Przydatny jeżeli chcemy coś szybko sprawdzić, a nie interesuje nas wygląd wykresu, niektóre wykresy można stworzyć tylko w tym pakiecie
- **lattice** - umożliwia łatwe tworzenie kilku wykresów na raz np. do porównania różnych cech, ale obecnie mniej popularny niż np. ggplot2
- **ggplot2** - jeden z najpopularniejszych pakietów R i mój ulubiony do przygotowywania wykresów. Oparty na grammar of graphics. Przygotowanie "ładnych" wykresów wymaga mniej pracy niż w podstawowym, ale składnia jest znacząco różna. Łatwe porównywanie i tworzenie kilku wykresów na jednym obrazku
- **plotly** - pozwala na tworzenie interaktywnych wykresów np. do wykorzystania na stronach www lub w shiny. Pakiet ggplotly umożliwia konwersję wykresów ggplot do postaci interaktywnej

Wykresy w ggplot2 składają się z kilku elementów: data, aesthetics, geom, scale, facet, theme itd.

Pierwsze trzy są niezbędne do przygotowania wykresu.

Poszczególne elementy można ze sobą łączyć na różne sposoby, co pozwala w prosty sposób robić bardzo różne wykresy.

Podstawowa funkcja : `ggplot()`. W niej określany jest tylko data i aesthetic, pozostałe elementy dodawane są przy pomocy `+`

- data - określa ramkę danych, na podstawie której będzie przygotowany wykres, najlepiej żeby była w formacie 'tidy'
- aesthetics (aes) - pokazuje (mapuje), które kolumny mają być wykorzystane np. x=kolumna_1 dla histogramu, można też określać pod względem których zmiennych dane mają zostać pogrupowane (group), rozróżnione (color, fill, shape itp) np. aes(x = kolumna_1, color = kolumna_2) znaczy że dane z kolumny_1 zostaną wykorzystane do stowrzenia osi X, a dane z kolumny_2 posłużą do stowrzenia skali kolorów.
- geom - rodzaj wykresu - to co widzimy - np histogram, density, bar, boxplot, point, line itp., można użyć kilka jednocześnie np. point i line, histogram i density
- scale - osie wykresu np. czy mają być logarytmiczne, procentowe, miejsce start i koniec, ale też skale kolorów, wypełnienia, kształtów itp.
- facet - umożliwia podział wykresu na kilka mniejszych pod względem danej zmiennej
- theme - wygląd poszczególnych elementów wykresu (motyw) np. czcionki, rodzaj linii, kolor tła, legenda itp., istnieje sporo już przygotowanych. Można przygotować własny i zapisać - łatwo można zrobić kilka wykresów w jednym stylu

Istnieje również funkcja `qplot`, która jest podobna do funkcji plot w podstawowym R, ale jej możliwości są uboższe w porównaniu z `ggplot`.

Wykresy ggplot są przypisywane do zmiennych np `p <- ggplot(...)`, zmienna p przechowuje wszystkie dane dotyczące wykresu. Możemy je sprawdzić funkcją `summary`. Dodanie kolejnych elementów do wykresu np. `p + geom_point()`, jeżeli chcemy nadpisać dotychczasowy wykres - `p <- p + geom_point()`. Korzystając z `+` możemy dodawać tyle elementów ile chcemy.

4.1 Używane dane

Dane 1 opisują jakąś cechę zmierzoną dla trzech szczepów w dwóch rodzajach warunków.

```
library(ggplot2) # ładujemy niezbędne pakiety
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
dane1 <- read.table("data/dane1.txt", header = TRUE, quote = "\"")

head(dane1)
```

```
##     pomiar Szczep warunki
## 1 3.389738      A       1
## 2 5.992065      A       1
## 3 4.464553      A       1
## 4 4.546082      A       1
## 5 6.521751      A       1
## 6 5.713088      A       1
```

Dane 2 zawierają ten sam zestaw szczepów i warunków, ale tym razem zostały one podzielone na 4 grupy: tak, nie, może i NA (brak danych).

```
dane2 <- read.table("data/dane2.txt", header = TRUE, quote = "\"")

head(dane2)
```

```
##   pomiar Szczep warunki
## 1    Tak      A       1
## 2    Nie      A       1
## 3    Tak      A       1
## 4    Tak      A       1
## 5   <NA>      A       1
## 6    Tak      A       1
```

```
summary(dane2)
```

```
##     pomiar             Szczep             warunki
##  Length:1200        Length:1200        Min.   :1.0
##  Class :character   Class :character   1st Qu.:1.0
##  Mode  :character   Mode  :character   Median :1.5
##                                        Mean   :1.5
##                                        3rd Qu.:2.0
##                                        Max.   :2.0
```

Dane 3 to wzrost dwóch szczepów w trzech powtórzeniach w czasie 1-100.

```
dane3 <- read.table("data/dane3.txt", header = TRUE, quote = "\"")

head(dane3)
```

```
##       pomiar Szczep powt czas
## 1 0.09639497      A    1    1
## 2 0.11830818      A    1    2
## 3 0.11839652      A    1    3
## 4 0.13904646      A    1    4
## 5 0.15544315      A    1    5
## 6 0.13080412      A    1    6
```

```
summary(dane3)
```

```
##      pomiar           Szczep               powt        czas
##  Min.   :0.09175   Length:1200        Min.   :1   Min.   :  1.00
##  1st Qu.:0.33428   Class :character   1st Qu.:1   1st Qu.: 25.75
##  Median :0.52688   Mode  :character   Median :2   Median : 50.50
##  Mean   :0.54175                      Mean   :2   Mean   : 50.50
##  3rd Qu.:0.74753                      3rd Qu.:3   3rd Qu.: 75.25
##  Max.   :1.10474                      Max.   :3   Max.   :100.00
```

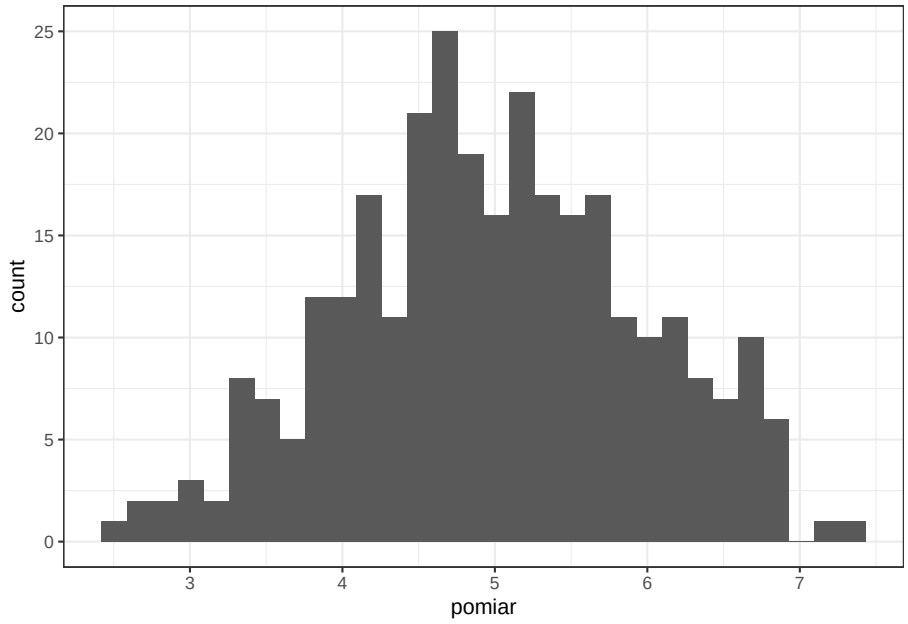
Dane tego typu możemy przedstawić jako wykres np. liniowy, punktowy, ribbon (liniowy z zaznaczonym błędem, przedziałem ufności itp.).

4.2 Wykresy jednowymiarowe - histogram, boxplot i inne

4.2.1 Histogram i density

```
theme_set(theme_bw()) # żeby wszystkie wykresy miały białe tło, a nie szare - będzie o tym późnie

dane1_1 <- subset(dane1, Szczep == "A" & warunki == "1")

p <- ggplot(data = dane1_1, aes(x = pomiar))
p + geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

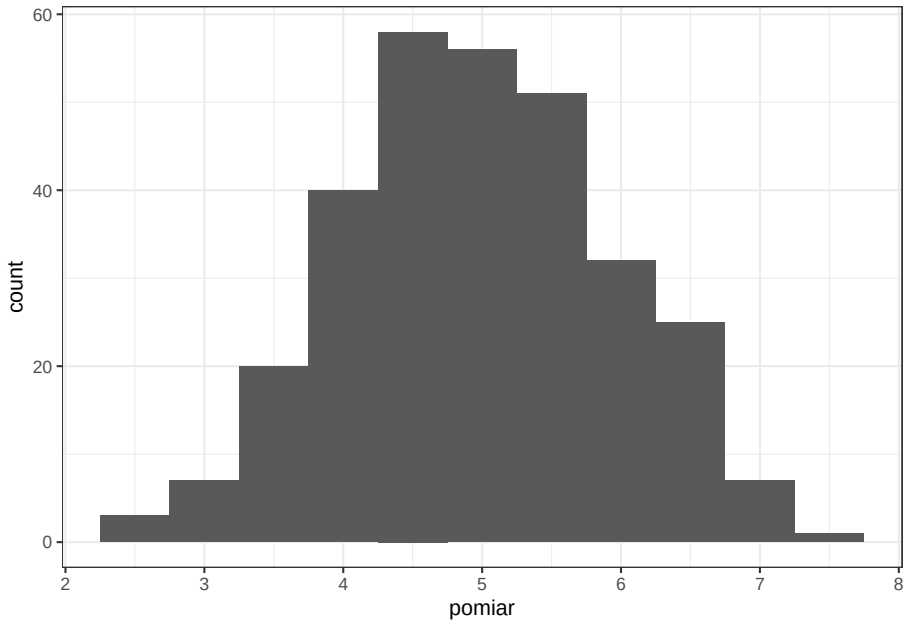


Dla histogramu najważniejszy parametr to `binwidth` określający szerokość "słupków".

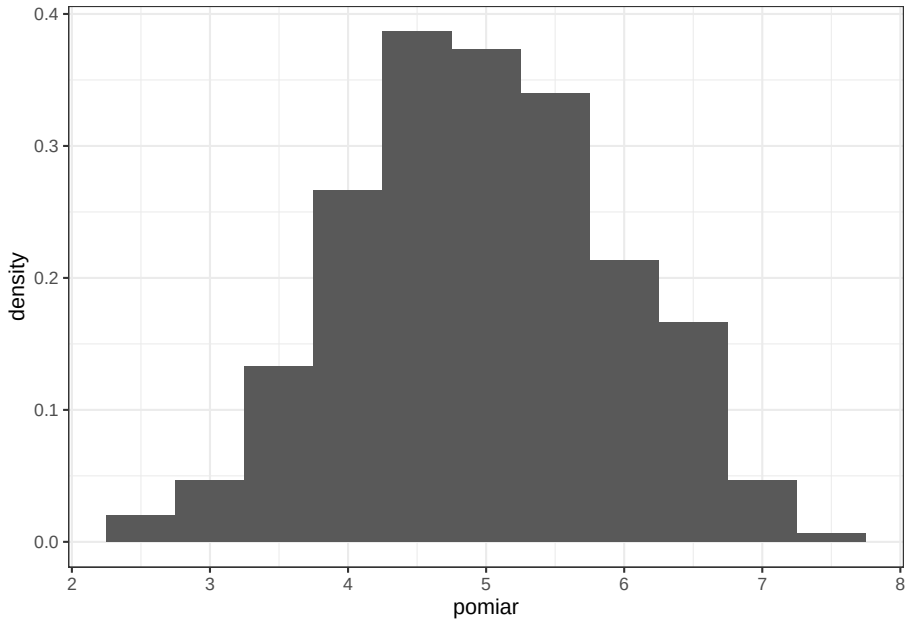
Możemy też wybrać czy chcemy żeby były zliczane ilości elementów (domyślnie), czy ma być pokazana gęstość rozkładu - w `aes` należy wpisać `y=..density..` albo procenty `y=((..count..)/sum(..count..))*100` (w przypadku procentów lepiej jednak policzyć je wcześniej i podać już gotowe wartości do ggplot, w bardziej skomplikowanych przypadkach ggplot może sobie nie poradzić). Do dodania znaków % potrzebna jest zmiana parametrów osi, o tym później.

Można też zmienić kolor wypełnienia słupków - `fill` lub kolor linii - color.

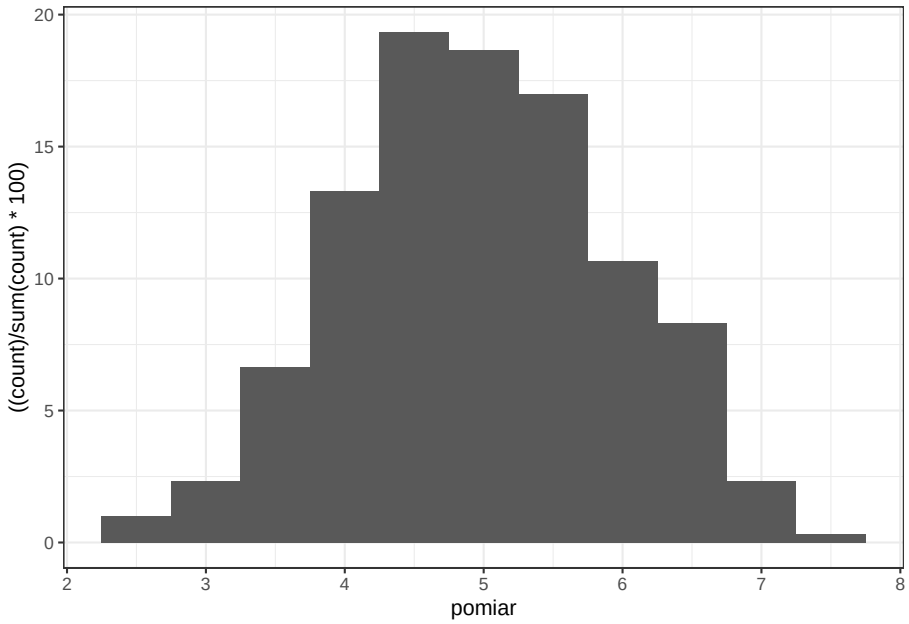
```
p + geom_histogram(binwidth = 0.5)
```



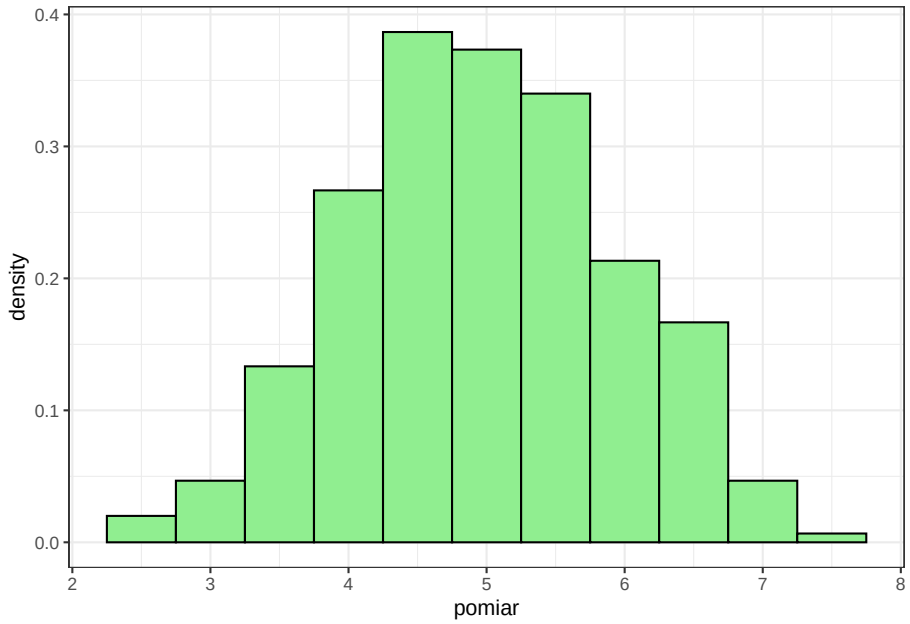

```
# Histogram z gęstością na osi Y
p + geom_histogram(binwidth = 0.5, aes(y = ..density..))
```



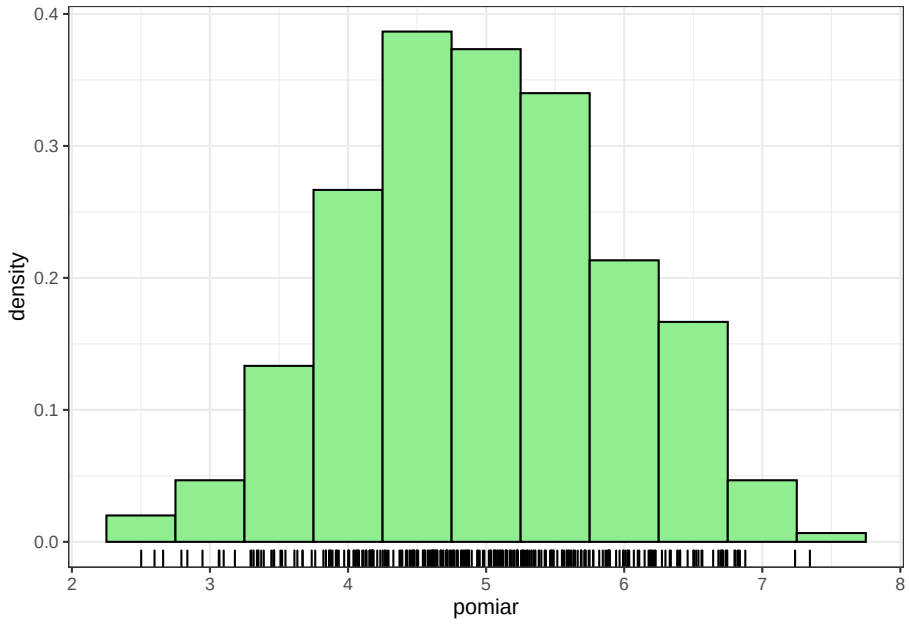

```
# Histogram z wartością % na osi Y
p + geom_histogram(binwidth = 0.5, aes(y = ((..count..)/sum(..count..)*100)))
```



```
# Zmiana koloru wypełnienia histogramu
p + geom_histogram(binwidth = 0.5, aes(y = ..density..),
                   fill = "lightgreen", color = "black")
```



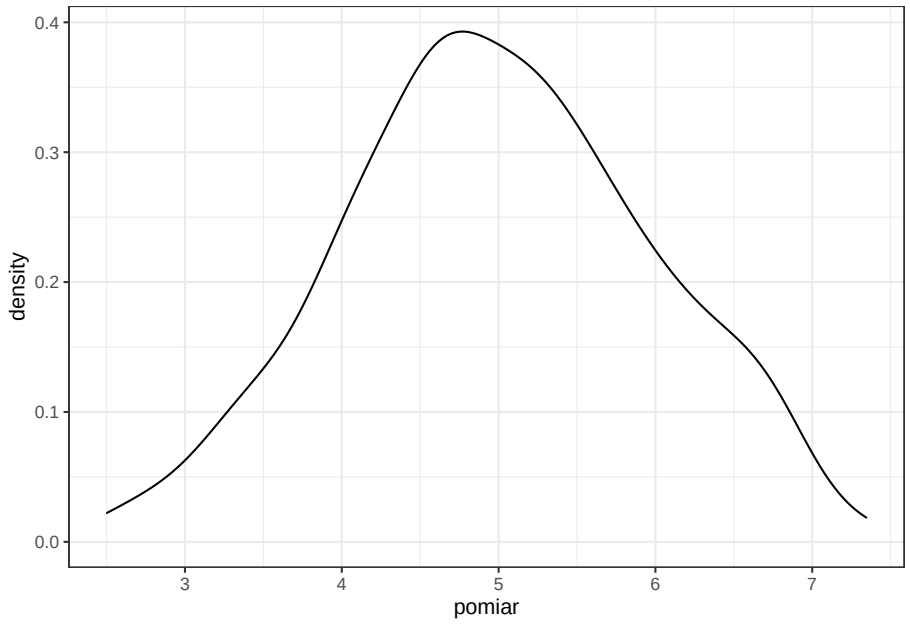

```
# Dodanie wszystkich obserwacji do wykresu - geom_rug()
p + geom_histogram(binwidth = 0.5, aes(y = ..density..),
                   fill = "lightgreen", color = "black")+
  geom_rug()
```



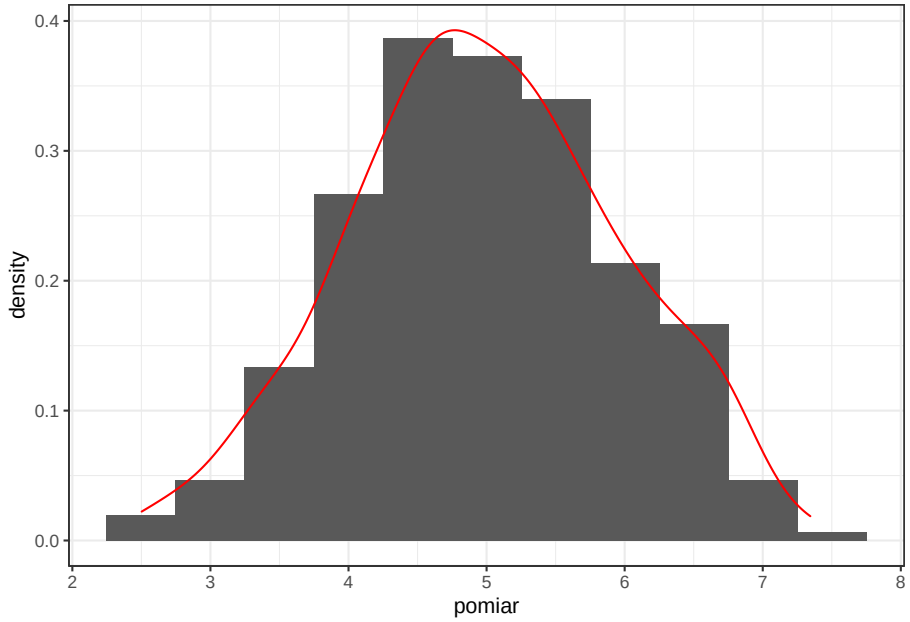
Histogram możemy łatwo zmienić na `geom_density` (gęstość) albo połączyć oba na jednym wykresie (należy pamiętać żeby ujednolicić oś Y - w histogramie ustawić `y=..density..` albo w density `y = ..count..`)

Wykresy gęstości są dużo czytelniejsze od histogramów przy większej liczbie grup/kolorów na wykresie.

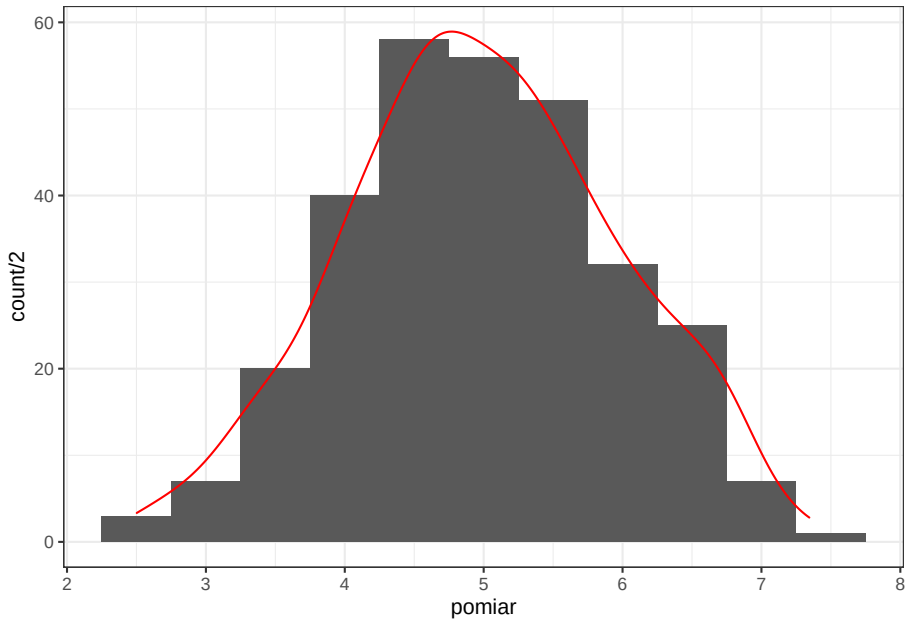
```
p + geom_density()
```



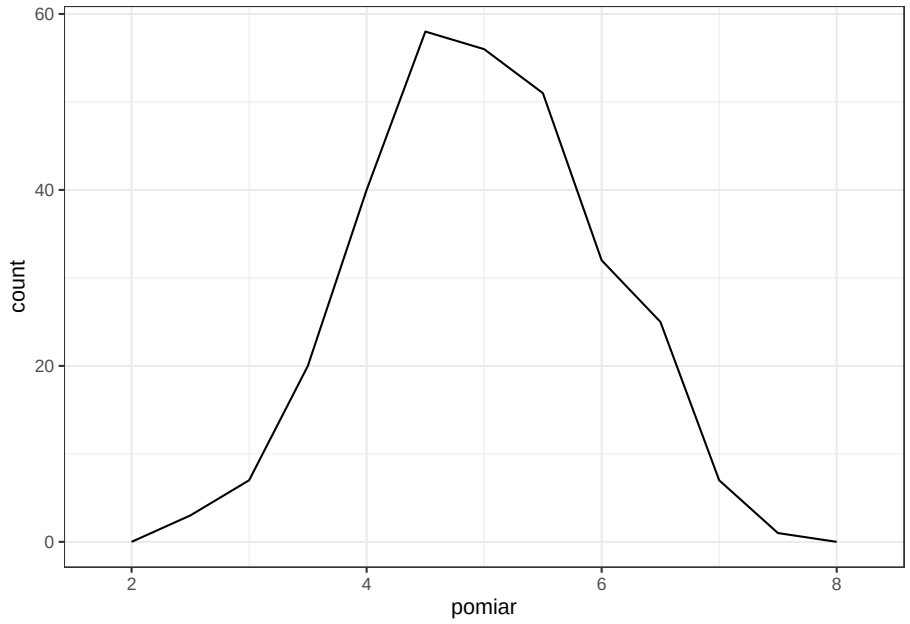

```
# Wykres łączący histogram i gęstość
p + geom_histogram(binwidth = 0.5, aes(y = ..density..))+
  geom_density(color = "red")
```



```
# albo tak, ale wtedy trzeba count w density podzielić przez coś - konkretną liczbę lepiej dobrać
p + geom_histogram(binwidth = 0.5)+
  geom_density(color = "red", aes(y = ..count../2))
```




```
# Zamiast density można też użyć geom_freqpoly, który da bardziej "kanciasty" wykres
# Wymaga parametru binwidth tak samo jak histogram
p + geom_freqpoly(binwidth = 0.5)
```

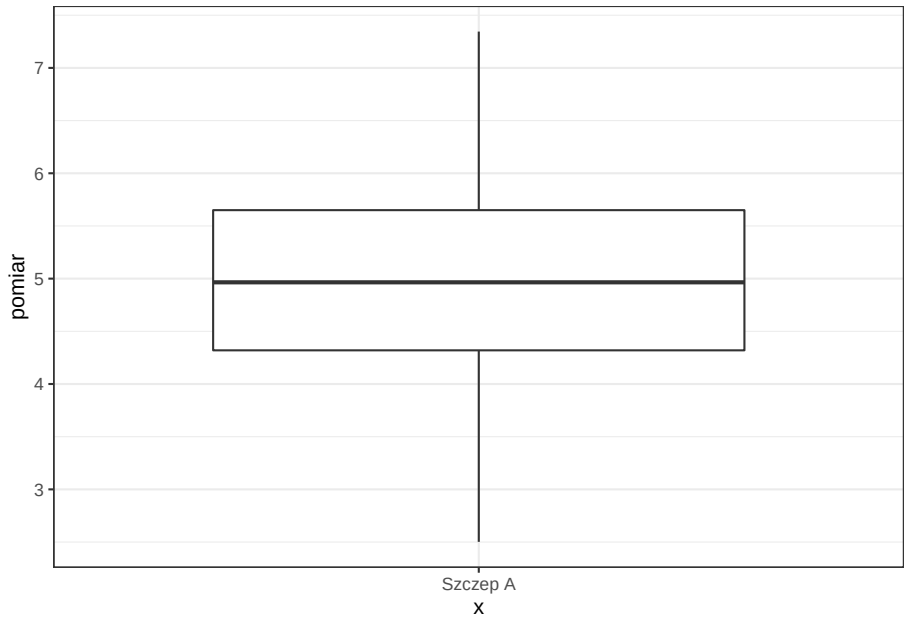


4.2.2 Wykres pudełkowy - boxplot

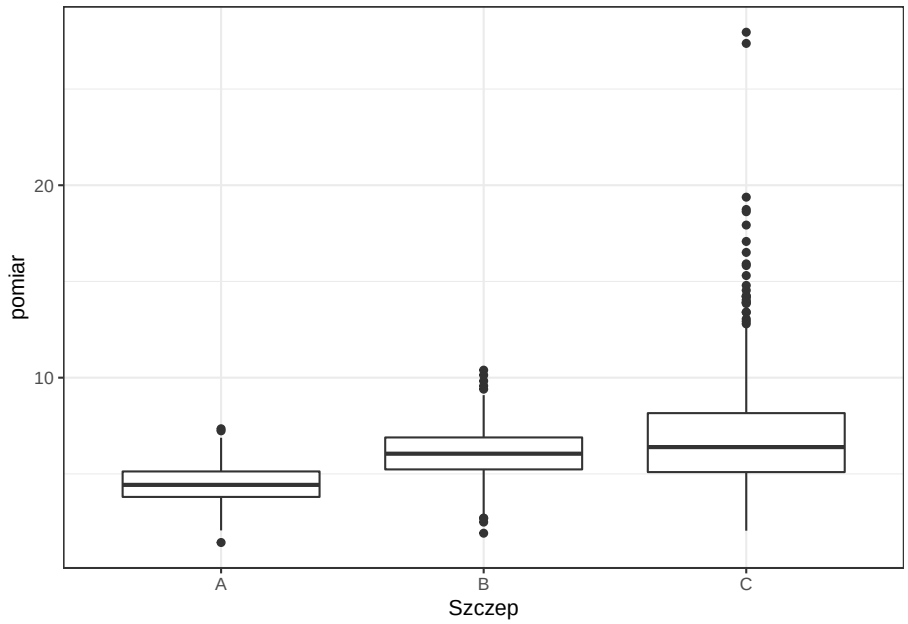
Na wykresie pudełkowym linia obrazuje medianę, pudełko to przestrzeń między 1 i 3 kwantylem, wąsy to zakres danych, a wszystkie punkty to obserwacje odstające.

Zamiast histogramu możemy zrobić boxplot, w tym wypadku x to nazwa szczepu, a y to mierzona cecha. Na osi X należy zawsze umieszczać zmienną jakościową, a na osi Y zmienną ilościową

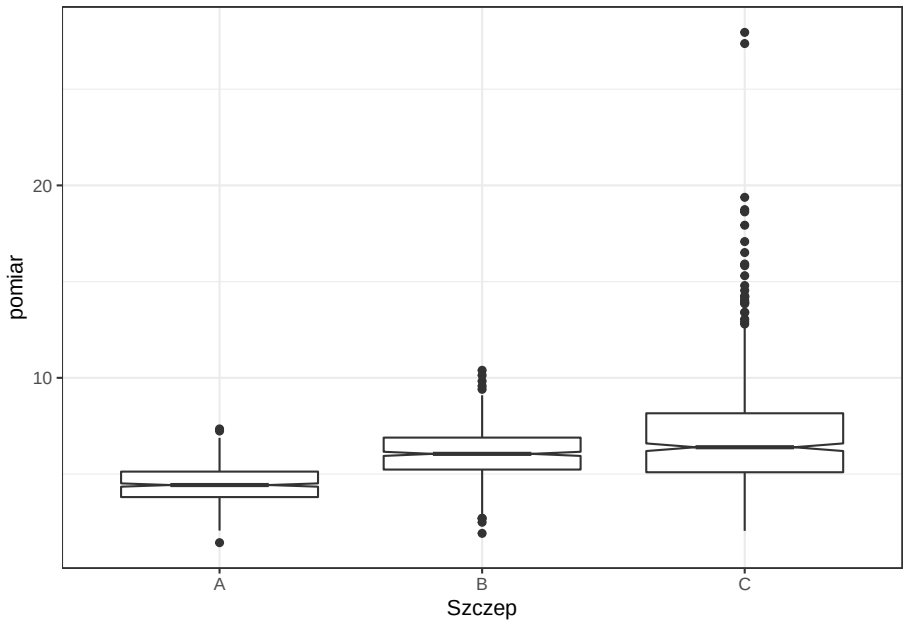
```
# pojedynczy boxplot
p <- ggplot(data = dane1_1, aes(x = "Szczep A", y = pomiar))
p + geom_boxplot()
```




```
# boxplot dla każdej kategorii
p <- ggplot(data = dane1, aes(x = Szczep, y = pomiar))
p + geom_boxplot()
```



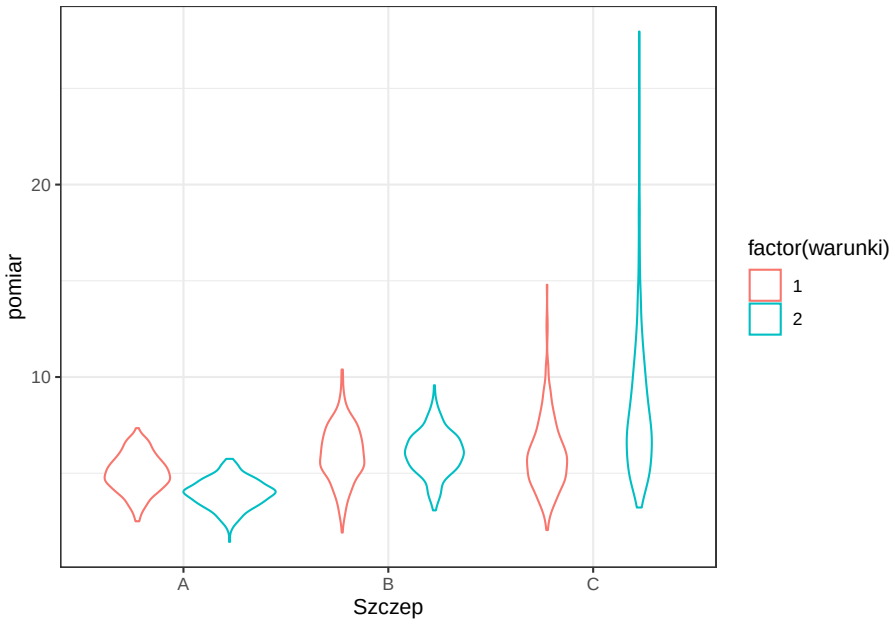
```
# do boxplota można dodać wcięcia, jeżeli wcięcia dwóch boxplotów na siebie nie zachod
# można uznać że mediany tych dwóch grup są od siebie znacząco różne
p + geom_boxplot(notch = TRUE)
```



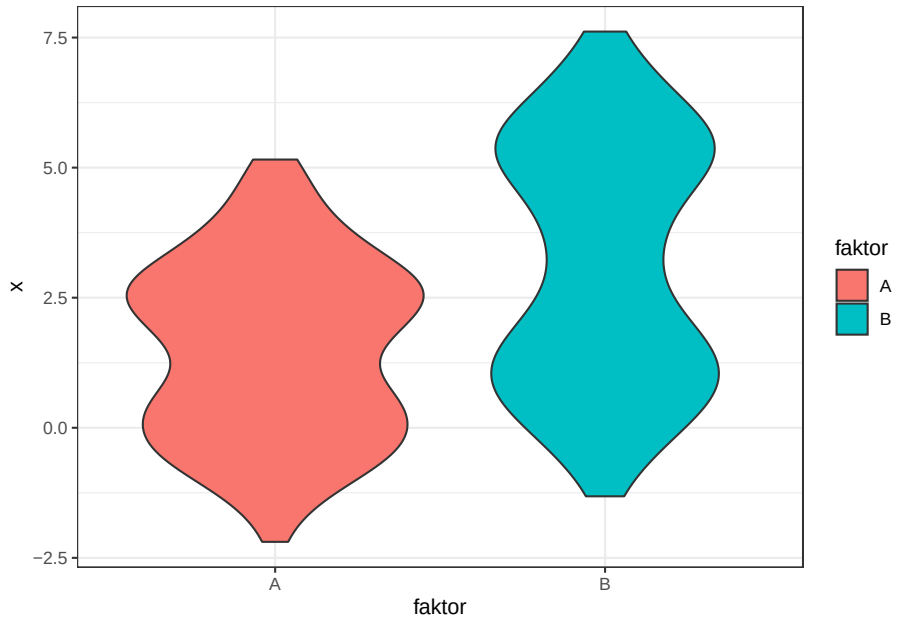
4.2.3 Wykres skrzypcowy

Odmianą boxplotów są tzw. wykresy skrzypcowe, które pozwalają też na pokazanie kształtu rozkładu - pozwala to np. na wykrycie rozkładu, który ma dwa maksima. W ggplot2 można je wygenerować funkcją `geom_violin`.

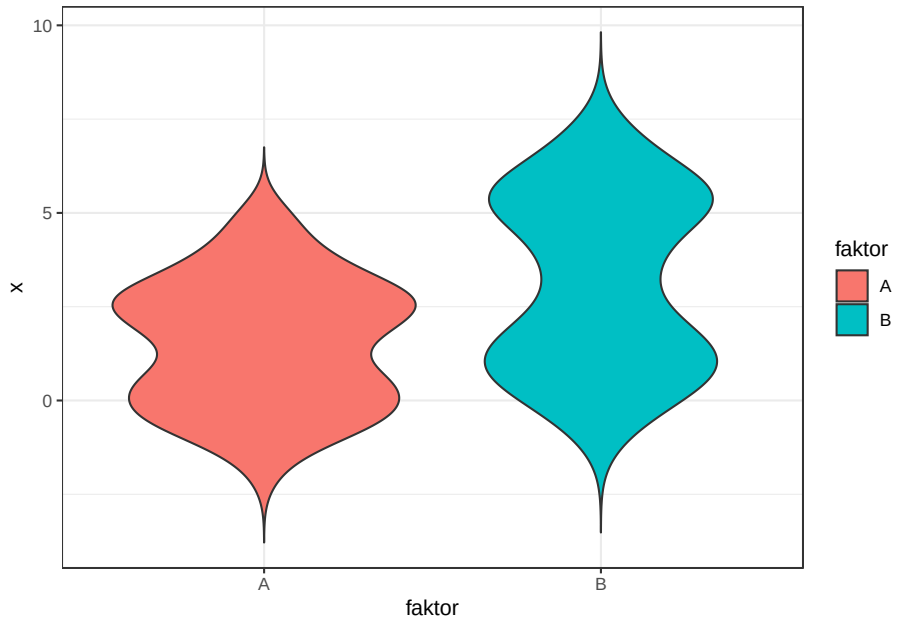
```
p <- ggplot(data=dane1, aes(x = Szczep, y = pomiar))
p + geom_violin(aes(color = factor(warunki)))
```




```
# przykładowe dane
dane <- data.frame(x = c(rnorm(100), rnorm(100, 3), rnorm(100, 1), rnorm(100, 5)),
                   faktor = rep(c("A", "B"), each = 200))

p <- ggplot(data = dane, aes(y = x, x = faktor, fill = faktor))
p + geom_violin()
```




```
# nie przyciety wykres
p + geom_violin(trim=FALSE)
```

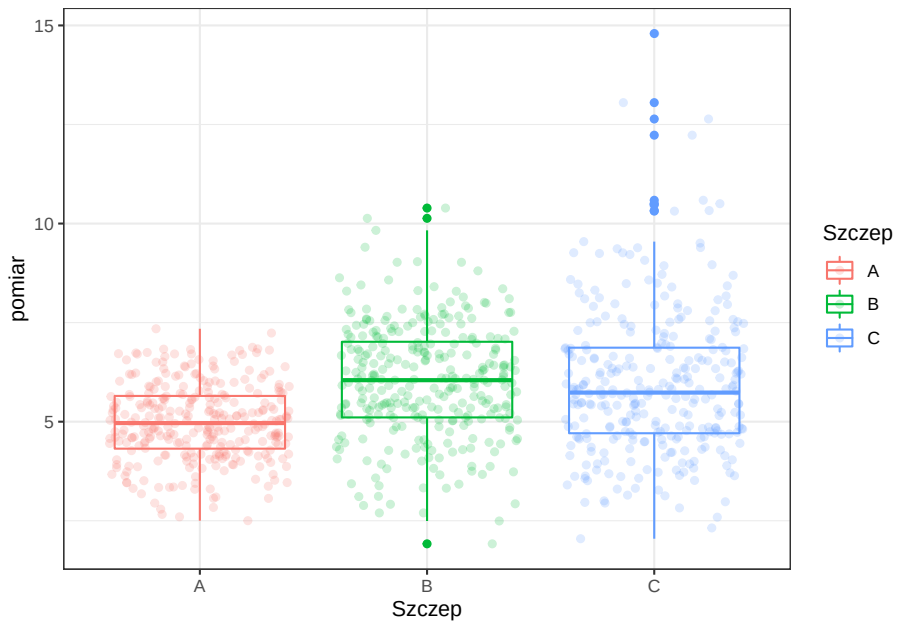


4.2.4 Dodanie wszystkich obserwacji do boxplot/violin

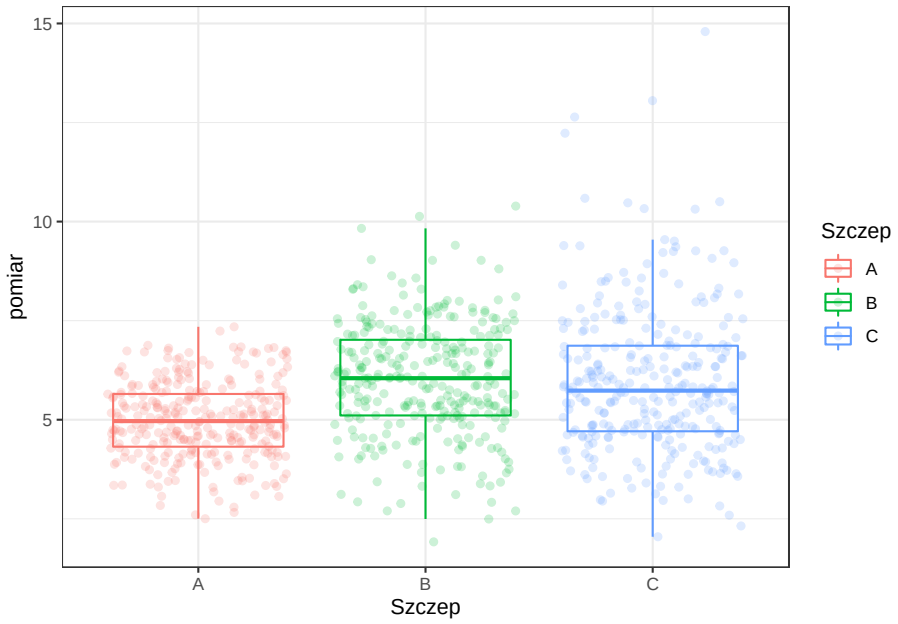
Dobrą praktyką jest przy używaniu wykresów pudełkowych lub skrzypcowych pokazanie wszystkich uzyskanych obserwacji (o ile nie ma ich za dużo). Można w tym celu użyć funkcji `geom_jitter` albo ładniejszych `geom_beeswarm` i `geom_quasirandom` z pakietu ggbeeswarm.

Przy pomocy argumentu alpha można kontrolować przezroczystość punktów - przydatne gdy jest ich dużo.

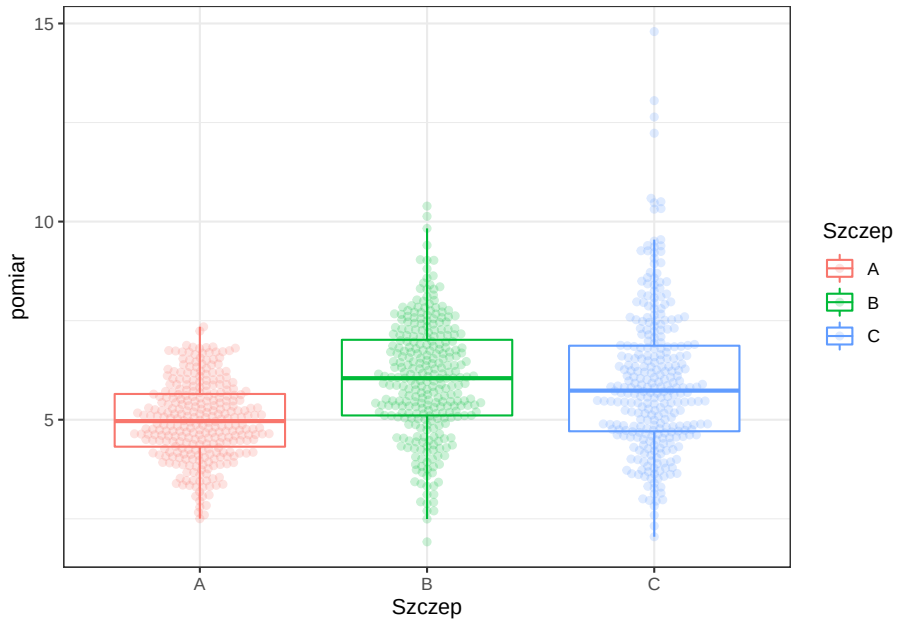
```
dane1_2 <- dane1 %>% filter(warunki == 1)

# Boxplot dla każdego szczepu
p <- ggplot(data = dane1_2, aes(x = Szczep, y = pomiar))
p + geom_boxplot(aes(color = Szczep))+
  geom_jitter(aes(color = Szczep), alpha = 0.2)
```



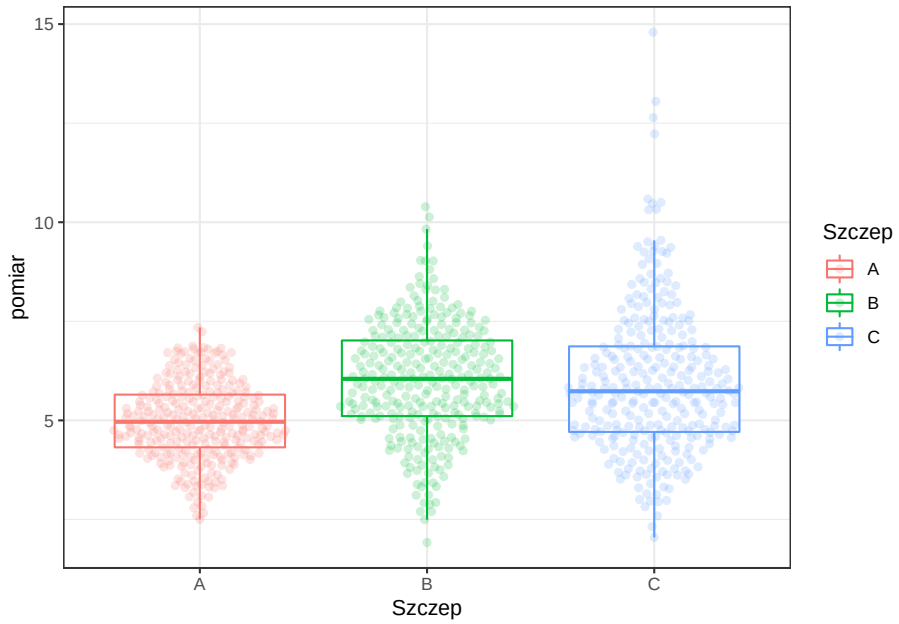

```
# usuwamy punkty pochodzące od boxplota
p + geom_boxplot(aes(color = Szczep), outlier.alpha = 0)+
  geom_jitter(aes(color = Szczep), alpha = 0.2)
```



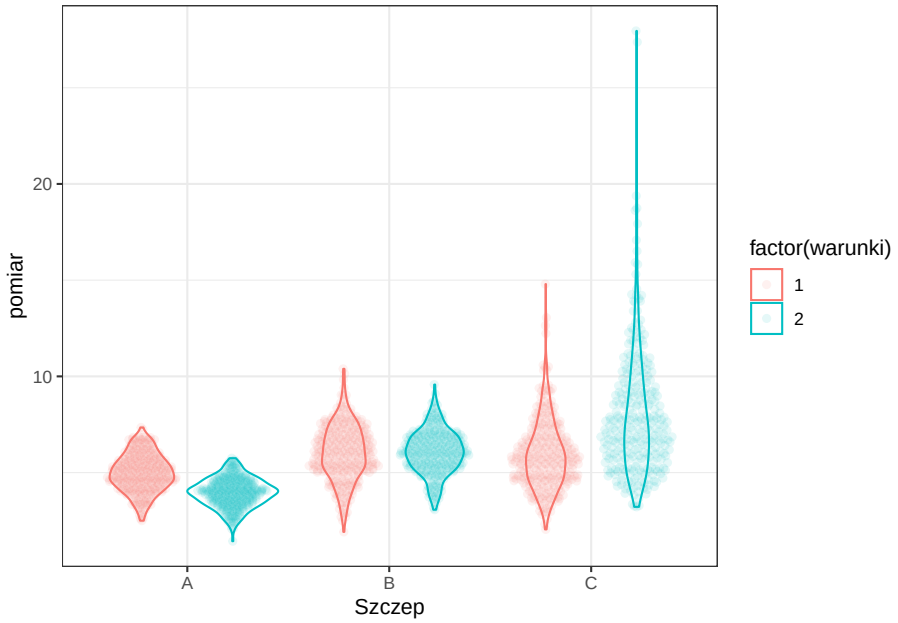

```
# beeswarm
library(ggbeeswarm)

p + geom_boxplot(aes(color = Szczep), outlier.alpha = 0)+
  geom_beeswarm(aes(color = Szczep), alpha = 0.2)
```




```
# quasirandom
p + geom_boxplot(aes(color = Szczep), outlier.alpha = 0)+
  geom_quasirandom(aes(color = Szczep), alpha = 0.2)
```



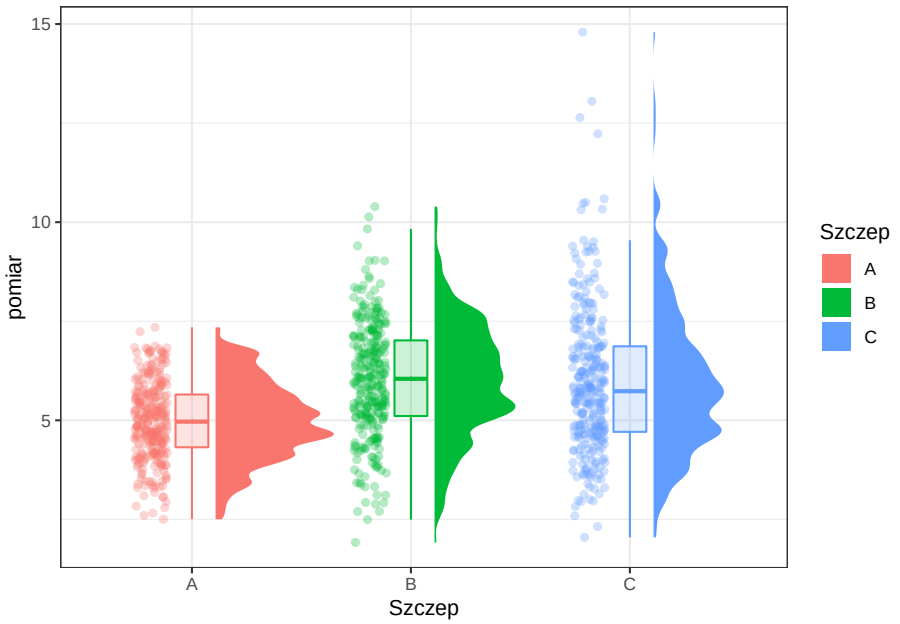
```
# quasirandom + geom_violin
p <- ggplot(data=dane1, aes(x = Szczep, y = pomiar))
p + geom_violin(aes(color = factor(warunki)))+
  geom_quasirandom(aes(color = factor(warunki)),
                   dodge.width = 0.9, # pozwala na rodział kolorów na dwie grupy
                   alpha = 0.1)
```



Można też połączyć wszystko w całość korzystając z dodatkowych pakietów ggdist i gghalves, na podstawie strony: visualizing-distributions-with-raincloud-plots-with-ggplot2

```
ggplot(dane1_2, aes(x = Szczep, y = pomiar, color = Szczep, fill = Szczep)) +
  ggdist::stat_halfeye(
    adjust = .5,
    width = .6,
    .width = 0,
    justification = -.2,
    point_colour = NA
  ) +
  geom_boxplot(
    width = .15,
    outlier.shape = NA,
    alpha = 0.2
  ) +
```

```
  ## add justified jitter from the {gghalves} package
  gghalves::geom_half_point(
    ## draw jitter on the left
    side = "l",
    ## control range of jitter
    range_scale = .4,
    ## add some transparency
    alpha = .3
  )
```

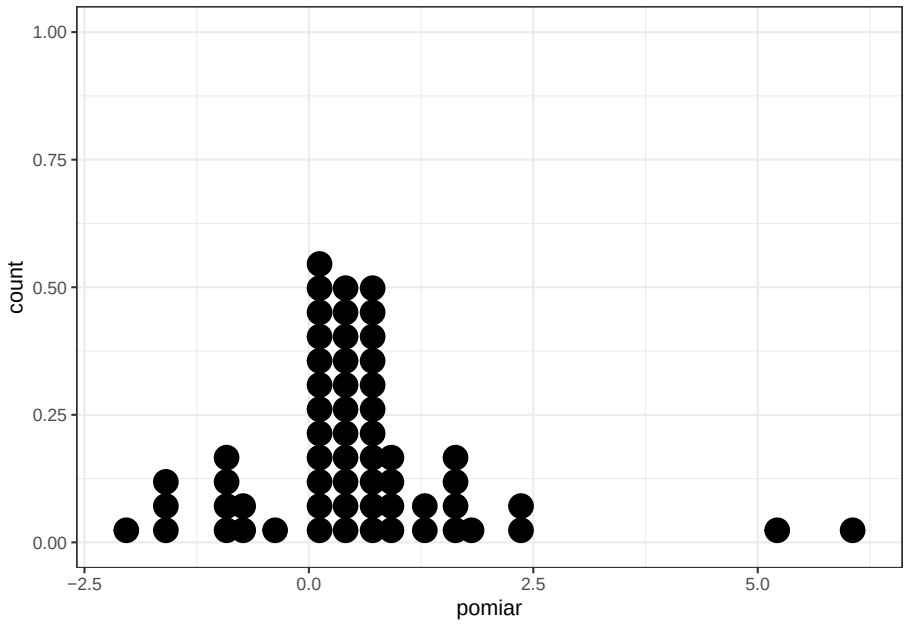


4.2.5 Dotplot

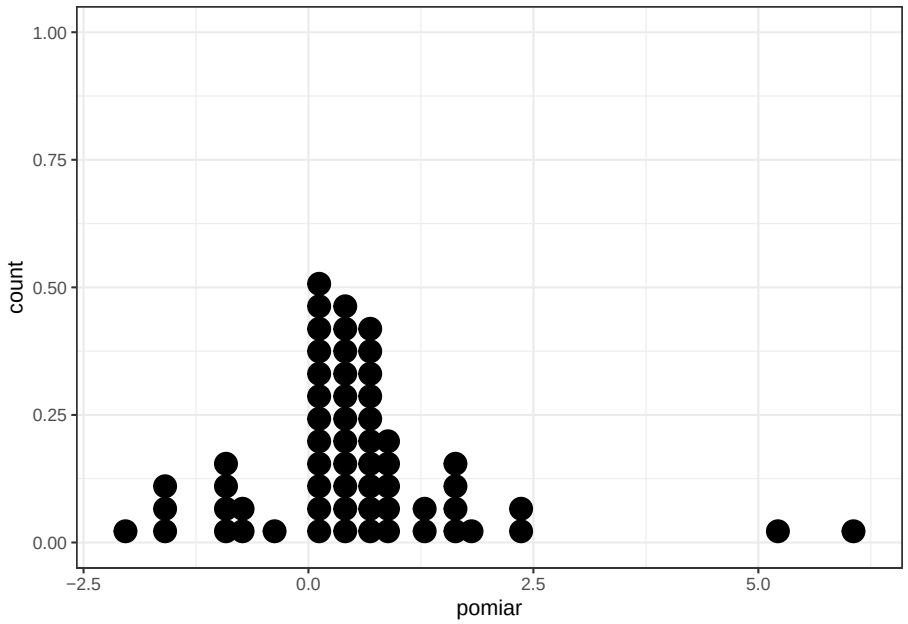
Wykres, na którym obserwacje są zaznaczane jako punkty, może być alternatywą dla histogramu albo gęstości jeżeli mamy małą liczbę danych. Punkty mogą być układane na którejś z osi albo wyśrodkowane. Podobnie jak w histogramie możemy określić parametrem `binwidth` jak mają być ułożone punkty.

```
dane_dot <- data.frame(pomiar = c(rnorm(20), rlnorm(20), runif(20)),
                       kategoria = rep(c("A","B","C"), each = 20))

p <- ggplot(dane_dot)

p + geom_dotplot(aes(x = pomiar))
```

```
## Bin width defaults to 1/30 of the range of the data. Pick better value with `binwidt
```

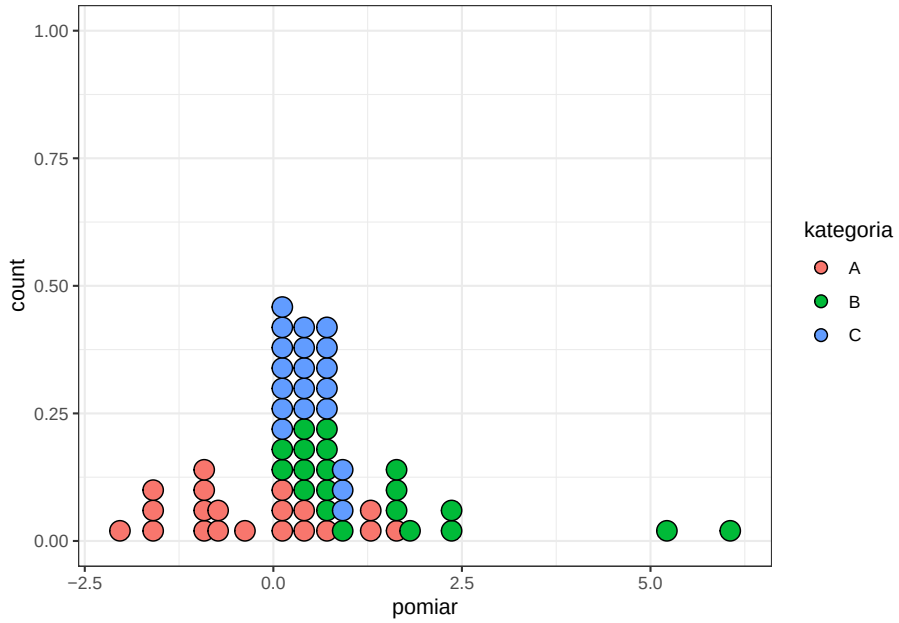



```
p + geom_dotplot(aes(x = pomiar), binwidth = 0.25)
```

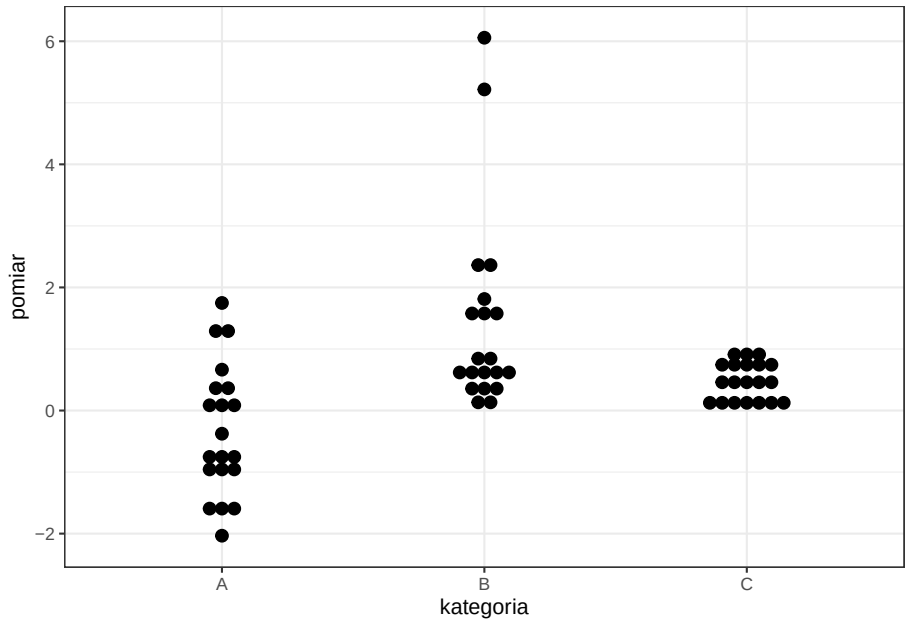


```
# możemy kropki pokolorować według kategorii,
# konieczne parametry stackgroups=TRUE i binpositions="all"

p + geom_dotplot(aes(x = pomiar, fill = kategoria), binpositions = "all", stackgroups = TRUE)
```

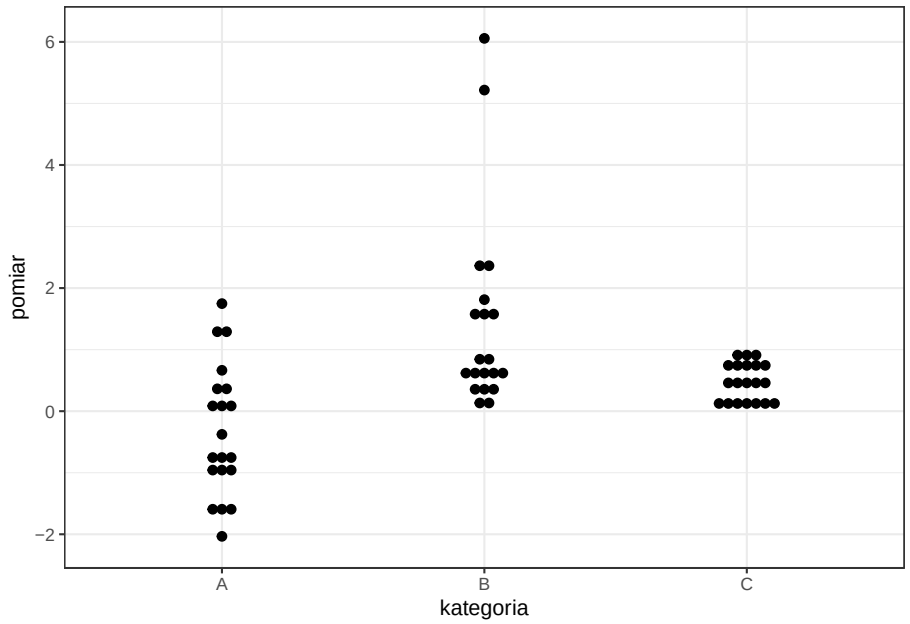
```
## Bin width defaults to 1/30 of the range of the data. Pick better value with `binwidth`.
```



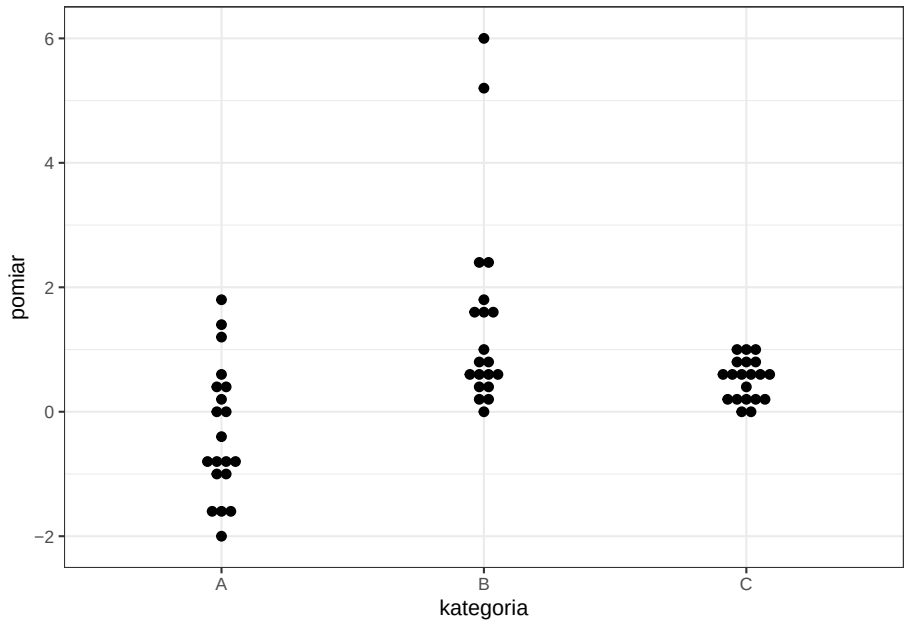

```
# albo przedstawić każdą kategorię osobno, binaxis określa w jakim kierunku układać kropki,
# stackdir czy mają być wyśrodkowane - center lub centerwhole

p + geom_dotplot(aes(y = pomiar, x = kategoria), stackdir = "center", binaxis = "y", binwidth = 0
```




```
# zmiana wielkości kropki przez dotsize

p + geom_dotplot(aes(y = pomiar, x = kategoria), stackdir = "center", binaxis = "y",
                 binwidth = 0.2, dotsize = 0.75)
```



```
# można też wybrać metodę układania kropek, domyślnie wedle gęstości,
# podobnie jak histogram - method="histodot"

p + geom_dotplot(aes(y = pomiar, x = kategoria), stackdir = "center", binaxis = "y",
                 binwidth = 0.2, dotsize = 0.75, method = "histodot")
```



4.3 Barplot - wykres słupkowy

Szybkie zliczenie elementów w poszczególnych grupach możemy wykonać stosując funkcję `table`. Wystarczy podać jej kolumnę z danymi oraz kolumny zawierające wektory według których dane mają zostać podzielone na grupy. Wynikiem `table` nie jest ramka danych, więc bez przekształcenia nie można go użyć do przygotowania wykresu ggplot.

```
table(dane2$pomiar)
```

```
##
## Może  Nie  Tak
##  224  469  437
```

```
table(dane2$pomiar, dane2$Szczep, dane2$warunki)
```

```
## , ,  = 1
##
##
##          A   B   C
##   Może  19  25  38
##   Nie   68  94 123
##   Tak   94  71  21
##
## , ,  = 2
##
##
##          A   B   C
##   Może  51  23  68
##   Nie   15  67 102
##   Tak  134  93  24
```

Można zauważyć, że funkcja `table`, w przeciwieństwie do `summary` pominęła wartości oznaczone jako NA. Można to zmienić używając parametr `useNA`.

```
table(dane2$pomiar, dane2$Szczep, dane2$warunki, useNA = "ifany")
```

```
## , ,  = 1
##
##
##          A   B   C
##   Może  19  25  38
##   Nie   68  94 123
##   Tak   94  71  21
##   <NA>  19  10  18
##
## , ,  = 2
##
##
##          A   B   C
##   Może  51  23  68
##   Nie   15  67 102
##   Tak  134  93  24
##   <NA>   0  17   6
```

Wykres słupkowy wymaga użycia `geom_bar`. Zasady rozróżniania zmiennych według kolorów i dzielenia wykresów na części pozostają takie same.

Domyślnie również wartości NA zostaną wzięte pod uwagę podczas zliczania. Jeżeli chcemy temu zapobiec, należałoby np. usunąć je wcześniej przy pomocy funkcji `filter`.

```
p <- ggplot(data = dane2, aes(x = pomiar, fill = Szczep))
# Słupki ustawione obok siebie
p + geom_bar(position = "dodge") + facet_wrap(~ warunki)
```

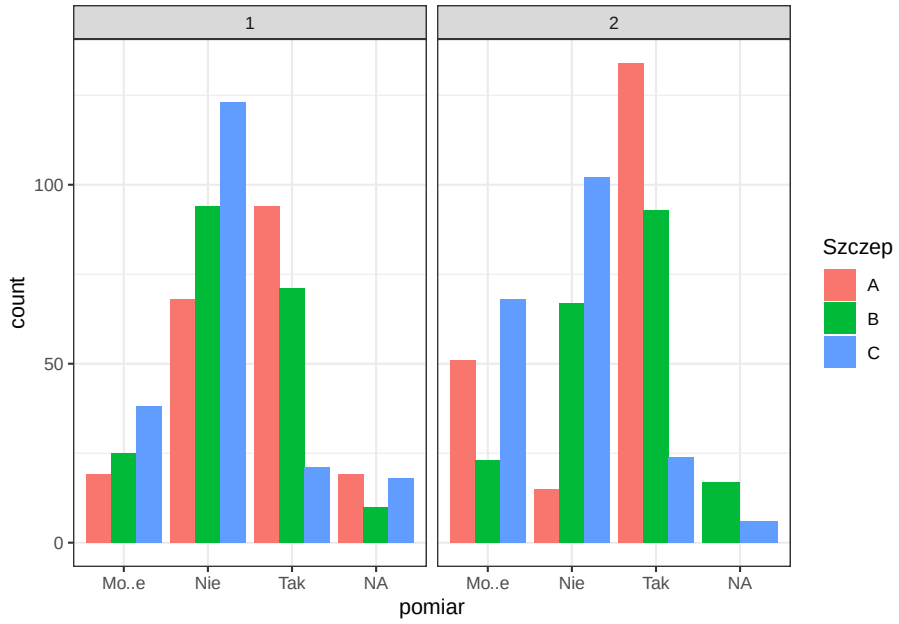
```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>
```

```
## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>
```




```
# Wszystkie słupki tej samej wysokości
p + geom_bar(position = "fill") + facet_wrap(~ warunki)
```

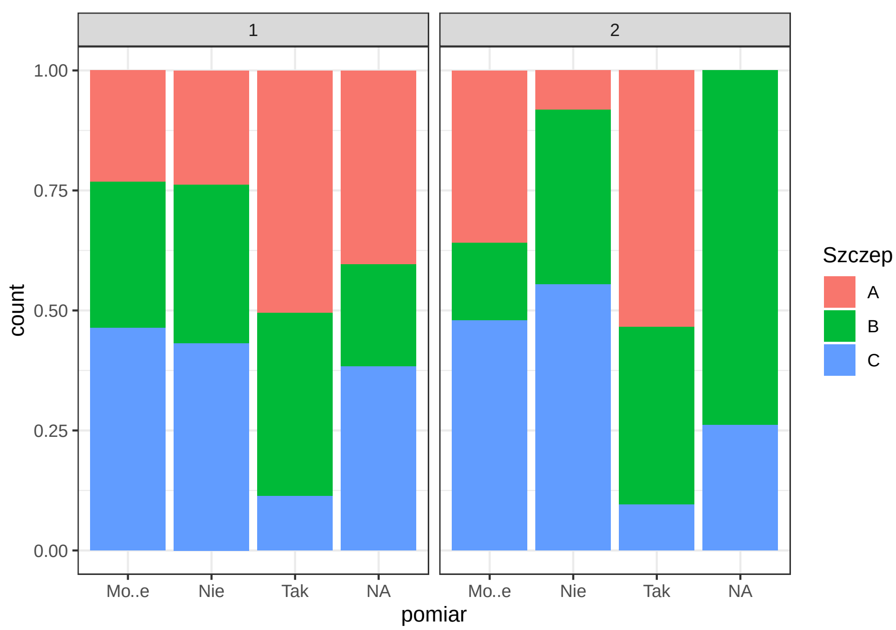
```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>
```

```
## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>
```



```
# Szerokość słupków można zmieniać parametrem width
p + geom_bar(position = "dodge", width = 0.4) + facet_wrap(~ warunki)
```

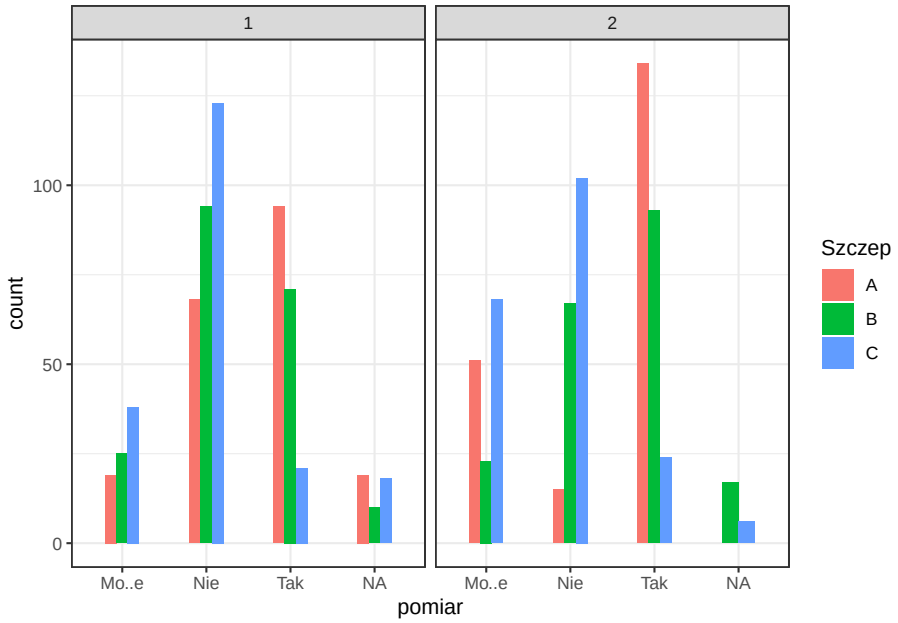
```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>
```

```
## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>
```




```
# Wykres bez wartości NA, filter można zrobić też wcześniej albo w obrębie ggplot
p <- ggplot(dane2 %>% filter(!is.na(pomiar)), aes(x = pomiar, fill = Szczep))
p + geom_bar(position = "dodge") + facet_wrap(~ warunki)
```

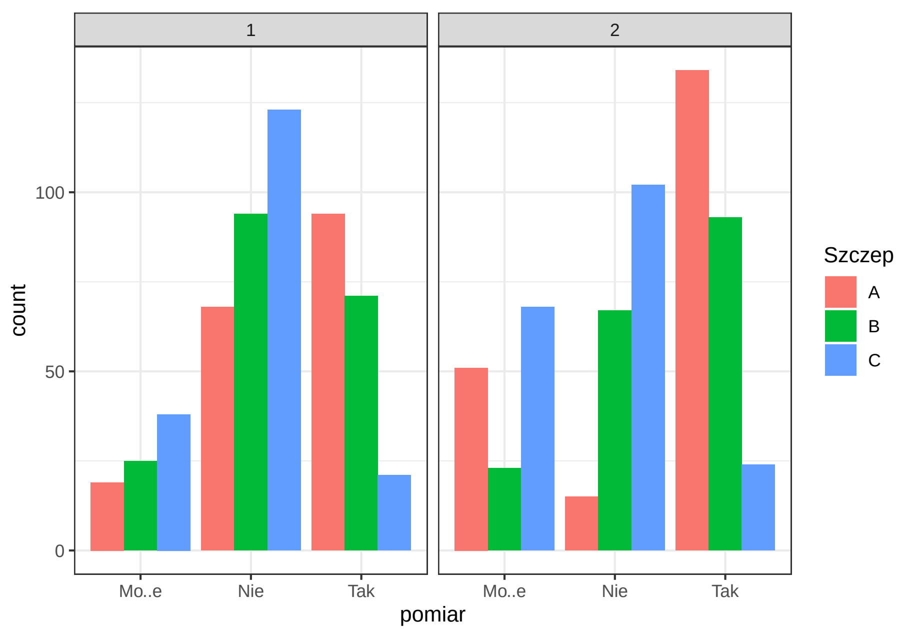
```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>
```



Przy dużej ilości różnokolorowych słupków wykres może być trudny do odczytania. Można wtedy rozważyć zastosowanie dotchart - czyli wykresu na którym wartości zliczenia są oznaczane przez pojedyncze punkty, a nie przez wysokość słupków. W ggplot2 do zrobienia dotchart możemy użyć `geom_point`

```
# Ustawiamy stat="bin" - oznacza że ggplot ma zliczyć częstości występowania elementów,
# a nie narysować każdy z osobna i obracamy wykres - ułatwia odczytanie
p + geom_point(size=3, stat='count', aes(color=Szczep))+
  facet_wrap(~warunki)+coord_flip()
```

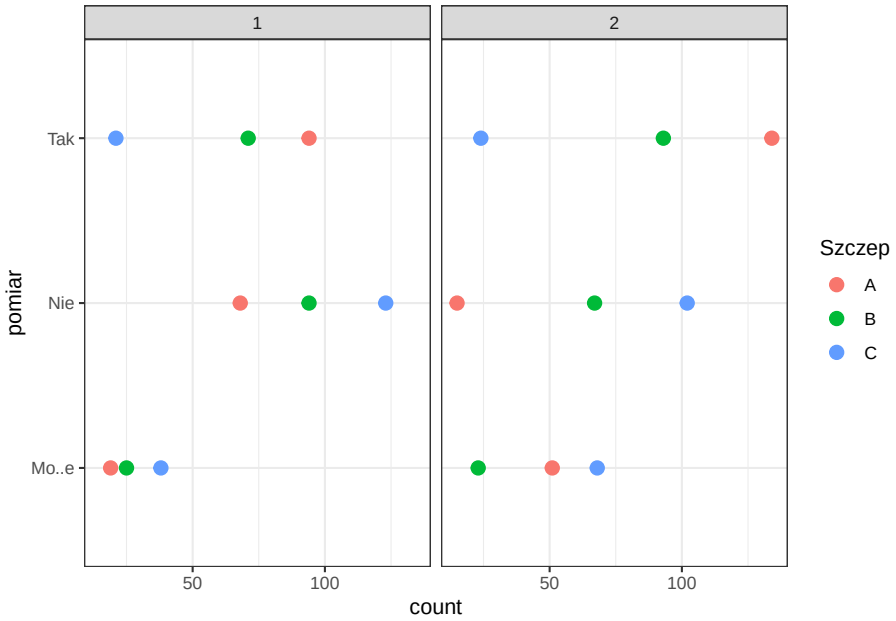
```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>
```

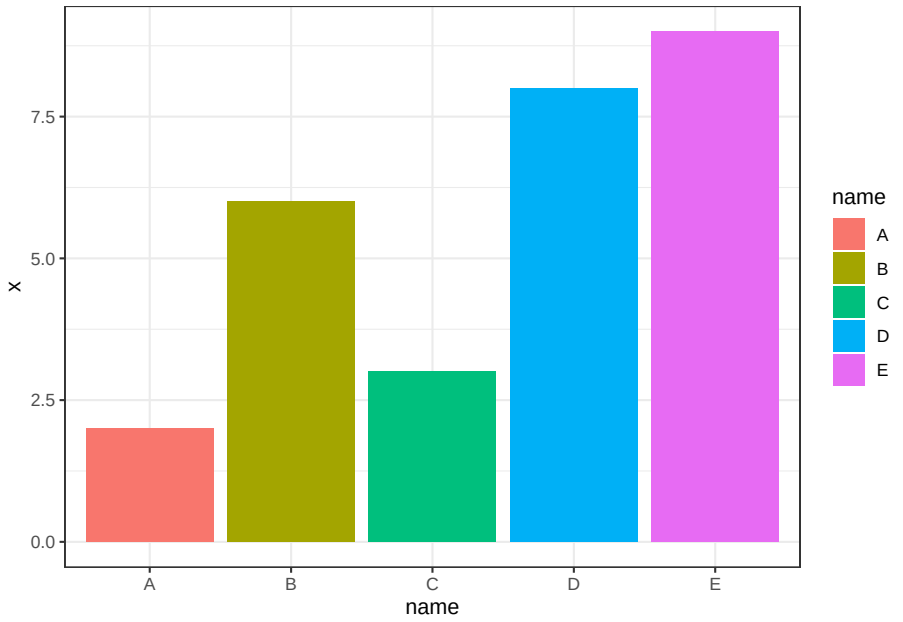
```
## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <c5>
```

```
## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Może' in 'mbcsToSbcs': dot substituted for <bc>
```



Nasze dane mogą też zawierać wysokości słupków. W takim wypadku należy użyć `geom_col`

```
x <- data.frame(x = c(2, 6, 3, 8, 9), name = c("A", "B", "C", "D", "E"))

p <- ggplot(data = x, aes(x = name, y = x))
p + geom_col(aes(fill = name))
```



4.4 Średnia, mediana itp na wykresie

4.4.1 Stat_summary

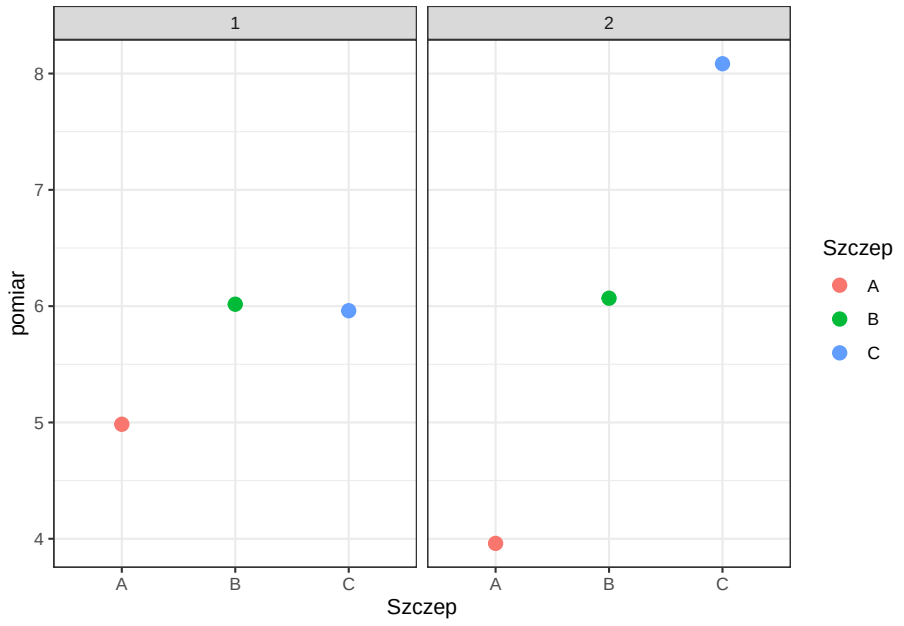
Możemy również potrzebować wykres z zaznaczoną średnią i przedziałem ufności albo błędem standardowym.

W takim wypadku możemy sami policzyć te wartości, a następnie pokazać je używając `geom_pointrange` albo `geom_crossbar`.

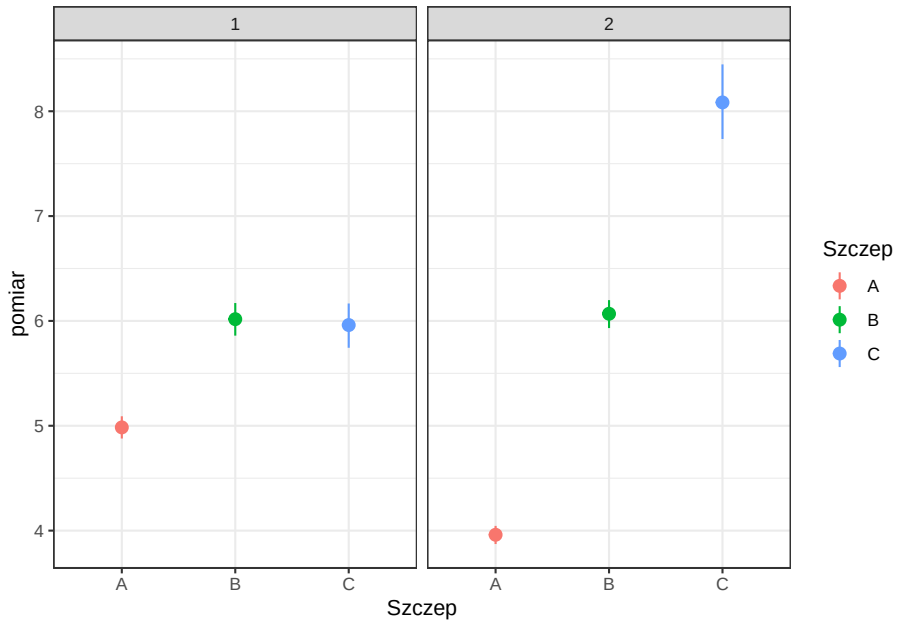
Można też wykorzystać `stat_summary`, który policzy je za nas.

W `stat_summary` najważniejszym argumentem jest funkcja jaką zastosujemy do podsumowania danych. Może to być albo fun.y - należy podać funckję, której wynikiem jest jedna wartość np. `mean`, `median`, `sd` albo fun.data - należy podać funkcję, której wynikiem jest więcej wartości np. `mean_cl_boot` i `mean_cl_normal` policzą średnią i przedział ufności. Należy też dobrać odpowiedni geom: dla jednej wartości np. point albo line, dla większej: linerange, crossbar, pointrange, errorbar.

```
p <- ggplot(data=dane1, aes(x=Szczep, y=pomiar, color=Szczep))
# Wykres tylko z wartością średnią
p + stat_summary(fun = "mean", geom = "point", size = 3)+
  facet_wrap(~ warunki, ncol = 2)
```




```
# Wykres ze średnią i przedziałem ufności policzonym metodą bootstrap
p + stat_summary(fun.data = "mean_cl_boot")+
  facet_wrap(~ warunki, ncol = 2)
```



4.4.2 Summary z użyciem dplyr

Do samodzielnego policzenia średnich możemy wykorzystać pakiet dplyr. Pozwala on m.in. na szybkie tworzenie podsumowań danych ze względu na zmienne np. różne szczepy.

Najpierw dane są dzielone na grupy, następnie każda grupa jest poddawana działaniu pewnej funkcji lub kilku funkcji, a wynik jest zapisywany do nowej ramki danych.

W pakiecie dplyr pierwszym krokiem jest podział na grupy - `group_by` i potem przekazania wyniku do funkcji `summarize`.

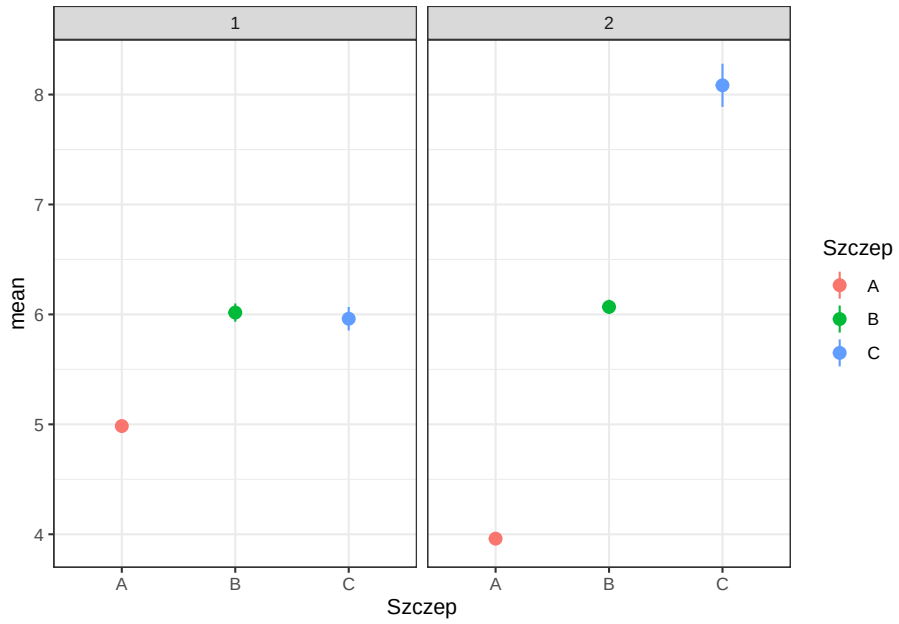
```
library(dplyr)

summ <- dane1 %>% group_by(Szczep, warunki) %>%
  summarize(mean = mean(pomiar), odch = sd(pomiar),
            blad = odch/sqrt(length(pomiar)),
            lower = mean-blad,
            upper = mean+blad)
```

```
## `summarise()` has grouped output by 'Szczep'. You can override using the `.groups` a
```

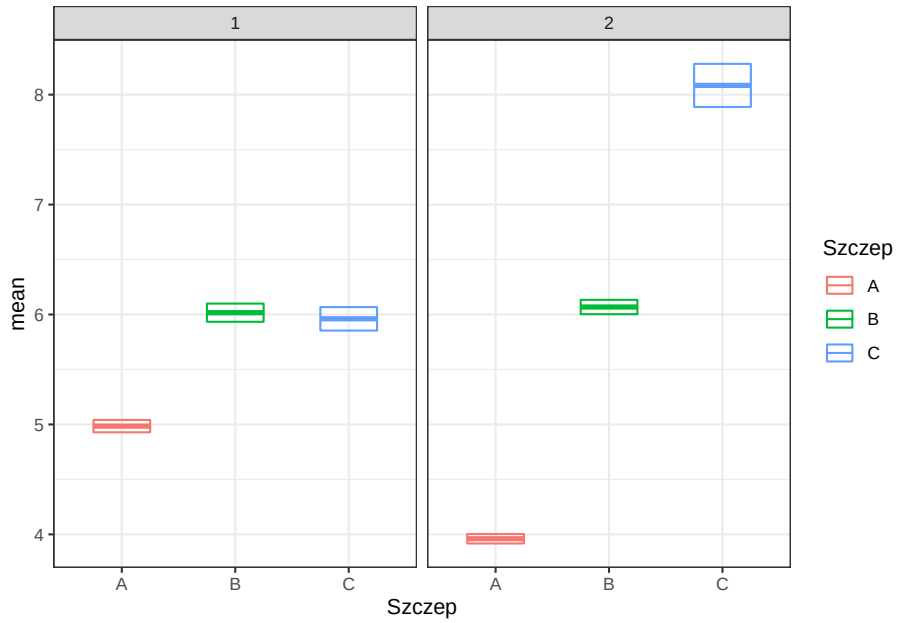
```
summ
```

```
## # A tibble: 6 x 7
## # Groups:   Szczep [3]
##   Szczep warunki  mean  odch   blad lower upper
##   <chr>    <int> <dbl> <dbl>  <dbl> <dbl> <dbl>
## 1 A            1  4.98 0.973 0.0562  4.93  5.04
## 2 A            2  3.96 0.748 0.0432  3.92  4.00
## 3 B            1  6.02 1.43  0.0828  5.93  6.10
## 4 B            2  6.07 1.13  0.0654  6.00  6.13
## 5 C            1  5.96 1.85  0.107   5.85  6.07
## 6 C            2  8.08 3.40  0.196   7.89  8.28
```

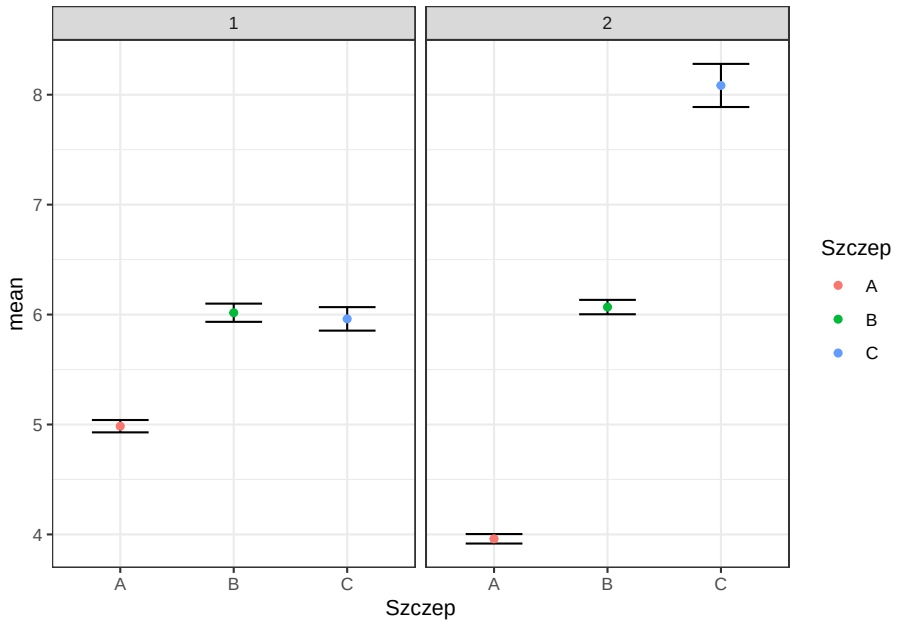
```
p <- ggplot(data = summ, aes(x = Szczep, y = mean, ymin = lower, ymax = upper,
                             color = Szczep)) + facet_wrap(~ warunki, ncol = 2)

# Wykres ze średnią i błędem standardowym
p + geom_pointrange()
```



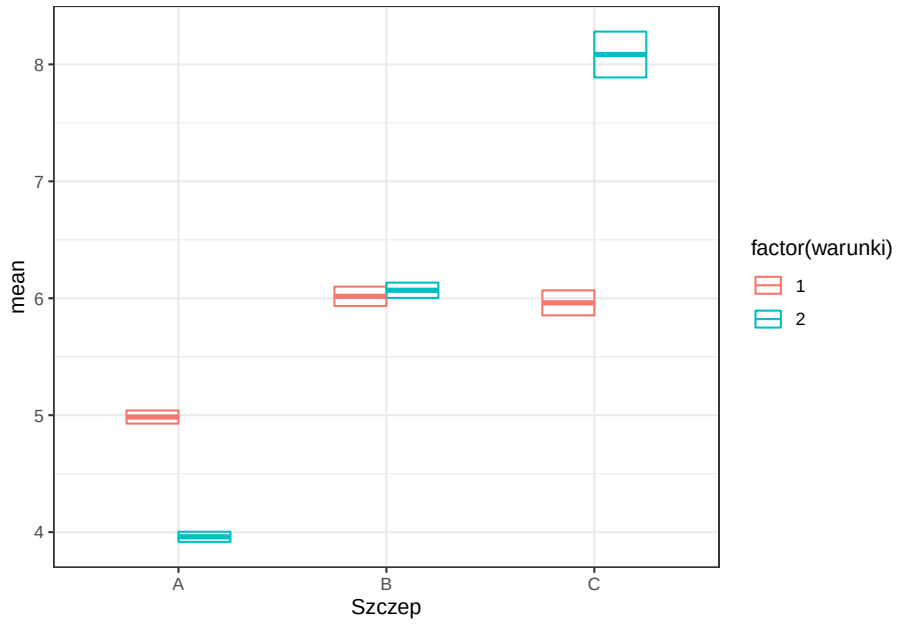

```
p + geom_crossbar(width = 0.5)
```



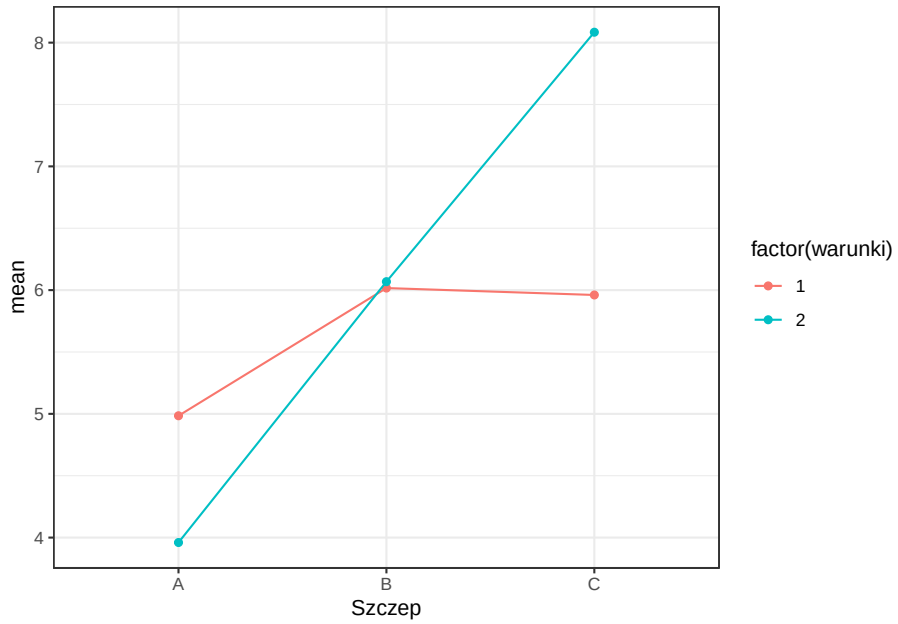
```
p + geom_errorbar(width = 0.5, color = "black") + geom_point()
```




```
# Z wykresem przedstawiającym średnie można postąpić tak samo jak z boxplotem
p <- ggplot(data = summ, aes(x = Szczep, y = mean, ymin = lower, ymax = upper,
                             color=factor(warunki)))
p + geom_crossbar(width = 0.5, position = "dodge")
```




```
# Jeżeli średnie połączymy za pomocą linii otrzymamy wykres interakcji opisany w dalszej części
p <- ggplot(data = summ, aes(x = Szczep, y = mean, color = factor(warunki)))
p + geom_point() + geom_line(aes(group = warunki))
```



4.4.3 Słupki błędów

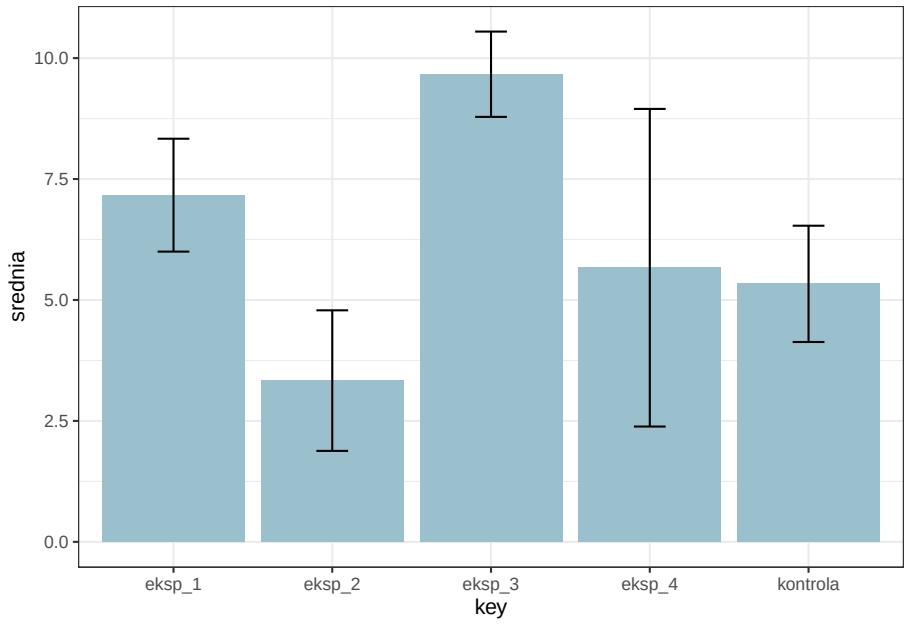
W publikacja często spotyka się wykresy słupkowe z dodanym słupkiem błędu. Podobny efekt w ggplot2 można uzyskać przy pomocy `geom_errorbar` albo `geom_linerange`. Należy w `aes` podać dolną i górną granicę słupka. Jednak jeżeli słupki pochodzą np. z trzech powtórzeń danego eksperymentu to może lepiej byłoby je zaznaczyć w postaci kropek z przedziałem ufności niż rysować słupki.

Wartość średnią i ewentualny błąd standardowy albo przedział ufności możemy policzyć sami albo wykorzystać w tym celu `stat_summary`.

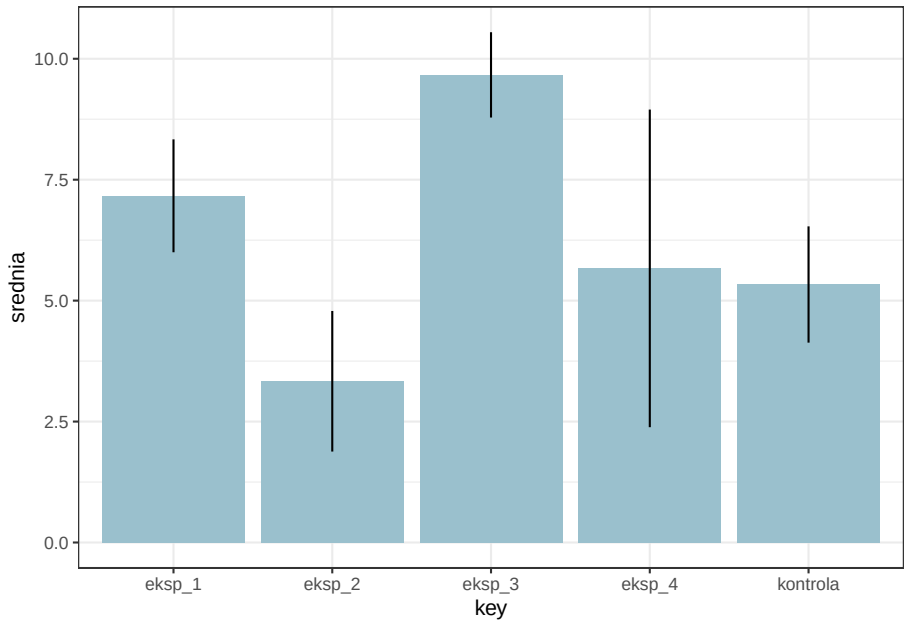
```
# Przykładowe dane

dane <- data.frame(kontrola = c(3,7,6), eksp_1 = c(5,9,7.5), eksp_2 = c(3,1,6), eksp_3

# przechodzimy do formatu tidy
library(tidyr)

dane %>% pivot_longer(cols = everything(), names_to = 'key', values_to = 'value') -> da
head(dane)
```

```
## # A tibble: 6 x 2
##   key      value
##   <chr>    <dbl>
## 1 kontrola     3
## 2 eksp_1       5
## 3 eksp_2       3
## 4 eksp_3      10
## 5 eksp_4       1
## 6 kontrola     7
```

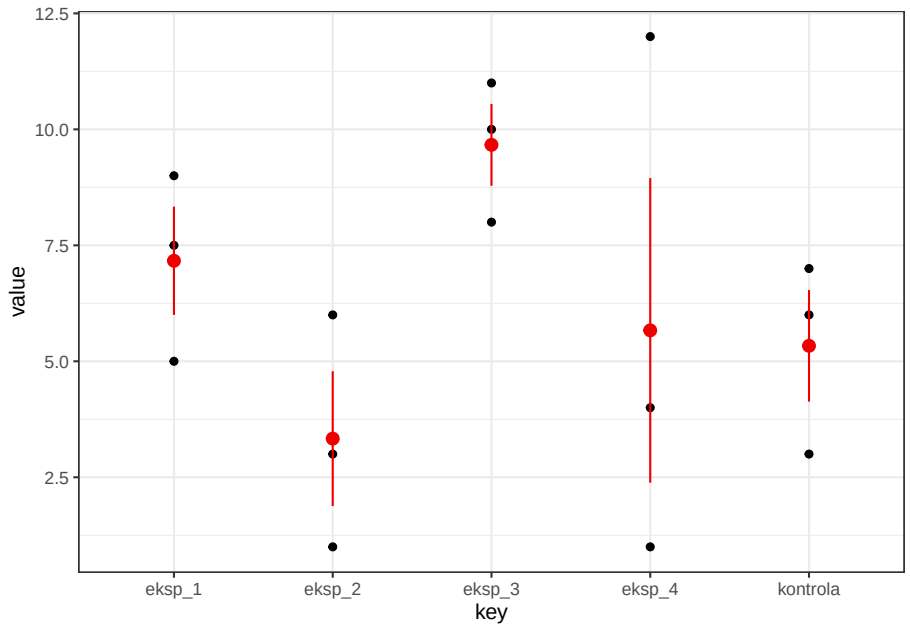
```
# liczymy średnią i np. błąd standardowy

podsumowanie <- dane %>% group_by(key) %>%
  summarize(                        srednia = mean(value),
                                    odchylenie = sd(value),
                                    dolny = srednia-(odchylenie/sqrt(length(value))),
                                    gorny = srednia+(odchylenie/sqrt(length(value))))

# Wykres słupkowy z zaznaczonym błędem standardowym
p <- ggplot(data = podsumowanie)
p + geom_col(aes(x = key, y = srednia), fill = "lightblue3")+
  geom_errorbar(aes(ymin = dolny, ymax = gorny, x = key), width = 0.2)
```



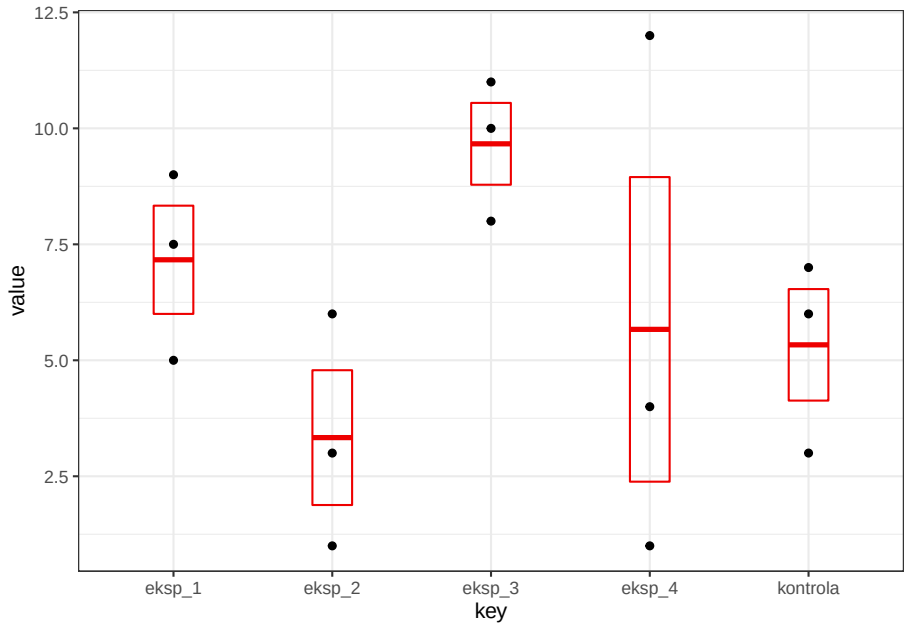

```
p + geom_col(aes(x = key, y = srednia), fill = "lightblue3")+
  geom_linerange(aes(ymin = dolny, ymax = gorny, x = key))
```



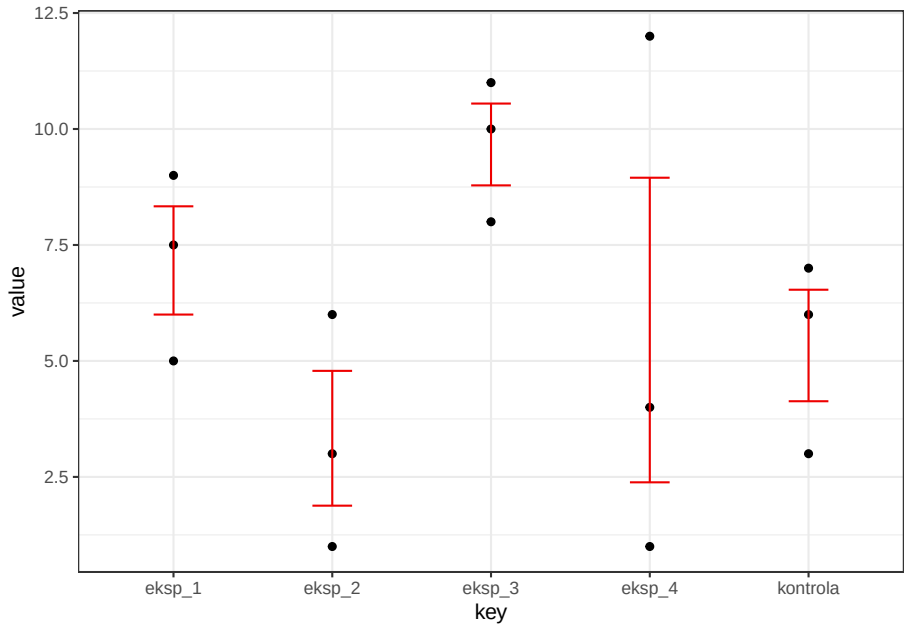
```
# Wykres punktowy wyników z zaznaczonym błędem standardowym
p + geom_point(data = dane, aes(x = key, y = value))+
  geom_pointrange(aes(x = key, ymin = dolny, ymax = gorny, y = srednia), col = "red2")
```




```
p + geom_point(data = dane, aes(x = key, y = value))+
  geom_crossbar(aes(x = key, ymin = dolny, ymax = gorny, y = srednia), col = "red2", wi
```




```
p + geom_point(data = dane, aes(x = key, y = value))+
  geom_errorbar(aes(x = key, ymin = dolny, ymax = gorny, y = srednia), col = "red2", width = 0.25
```

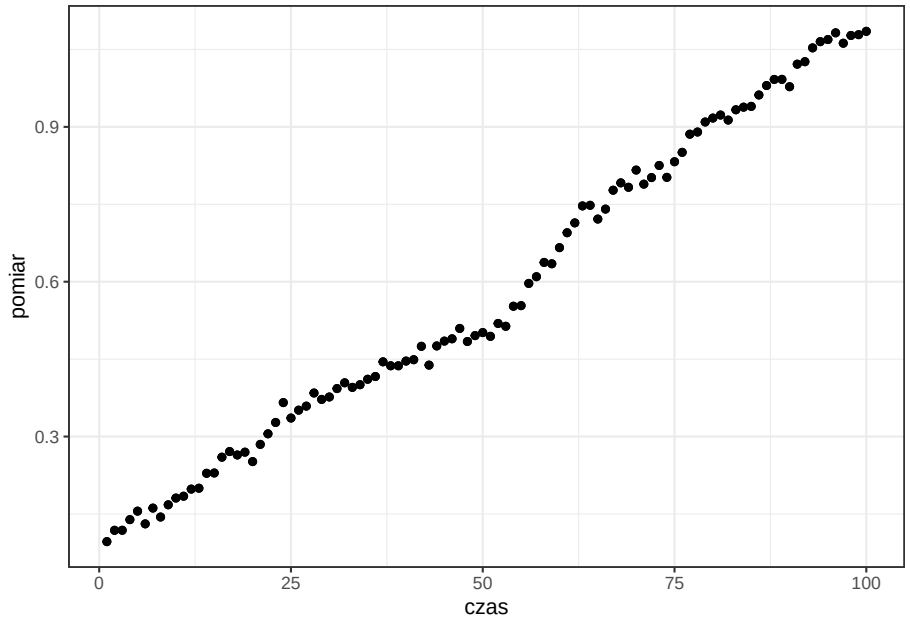


4.5 Wykres punktowy i liniowy

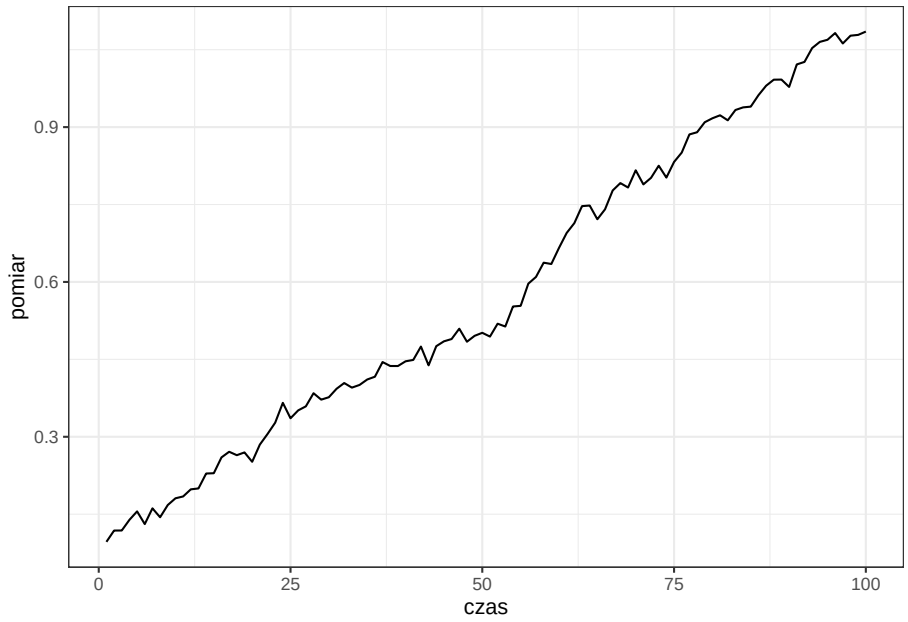
Zacznijmy od pojedynczego pomiaru szczepu A z dane3

Dla `geom_point` albo `geom_line` konieczne jest w aes podanie x i y.

```
dane3_1 <- filter(dane3, Szczep == "A" , powt == "1")

p <- ggplot(data = dane3_1, aes(x = czas, y = pomiar))
p + geom_point()
```




```
p + geom_line()
```

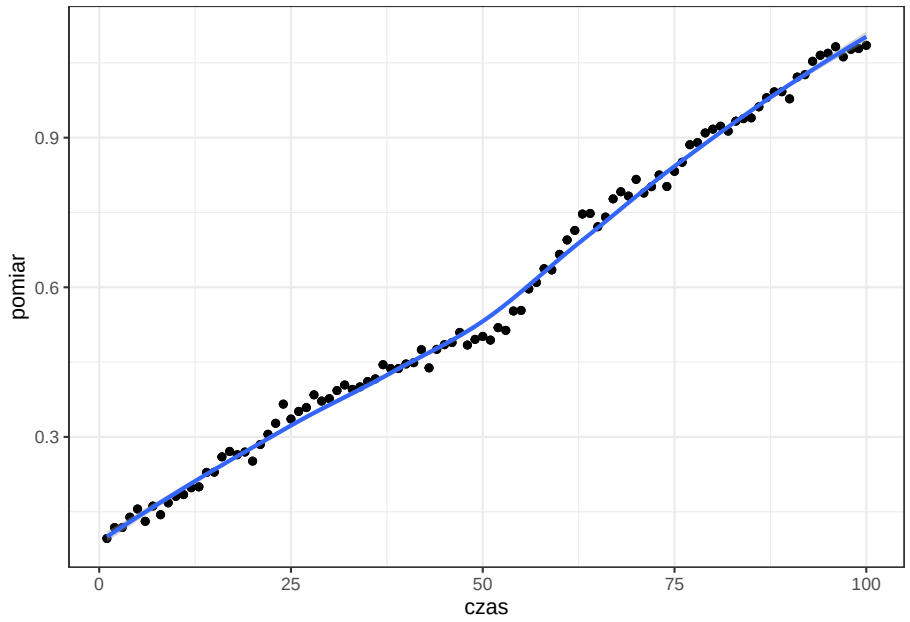


4.5.1 Dopasowanie linii trendu do wykresu punktowego

Zamiast rysować linię, możemy dopasować do wyników linię trendu (może być też inna niż liniowa np. logarytmiczna, kwadratowa, ograniczają nas tylko umiejętności pisania formuł w R ;)). Dopasowanie wykona `stat_smooth`. Domyślnie dopasuje linię do danych korzystając z własnego algorytmu (loess - lokalne wygładzanie wielomianami niskich stopni ;)), który ma za zadanie uzyskać jak najlepsze dopasowanie do danych, zaznaczy również przedział ufności. Możemy narzucić własną metodę i formułę.

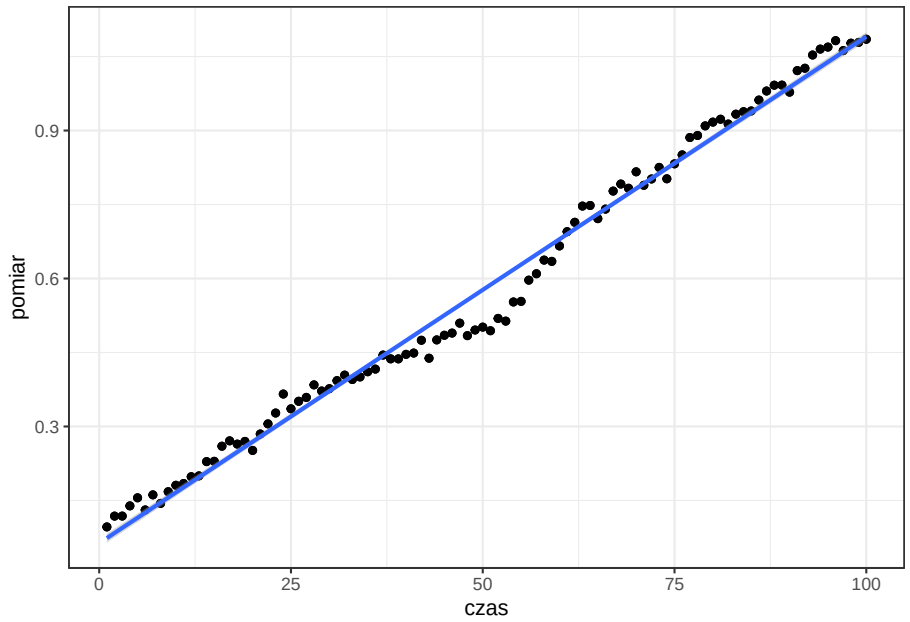
```
# metoda loess
p + geom_point() + stat_smooth()
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



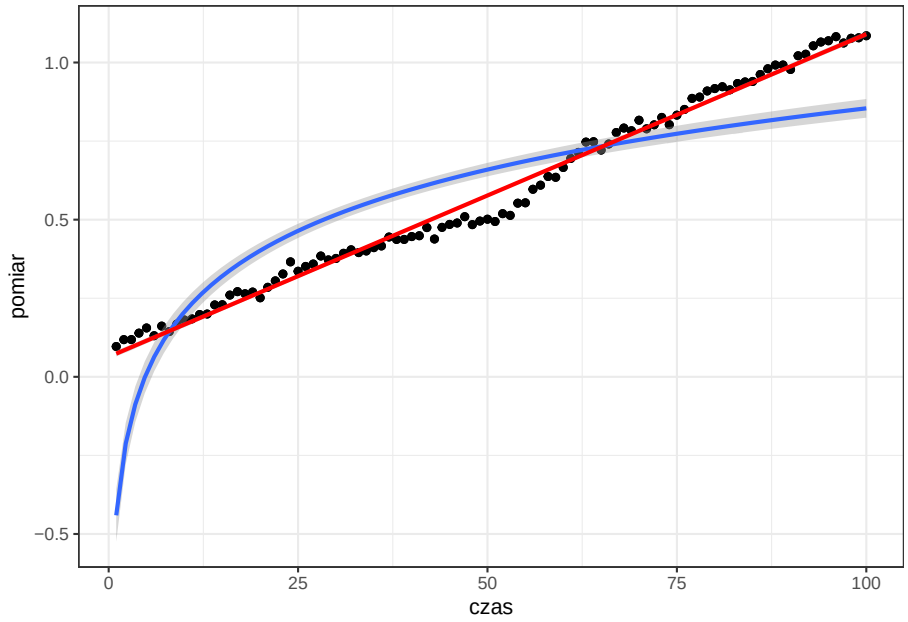

```
# metoda lm - dopasowanie do linii prostej
p + geom_point() + stat_smooth(method = "lm")
```

```
## `geom_smooth()` using formula 'y ~ x'
```



```
# metoda lm z podaną formułą
p + geom_point() + stat_smooth(method = "lm", formula = y~log(x))+
  stat_smooth(method = "lm", color = "red")
```

```
## `geom_smooth()` using formula 'y ~ x'
```



Sprawdzenie dopasowania można wykonać funkcją `lm`. Jej `summary` podaje wartośc R^2, istotność, parametry wzoru(coefficients).

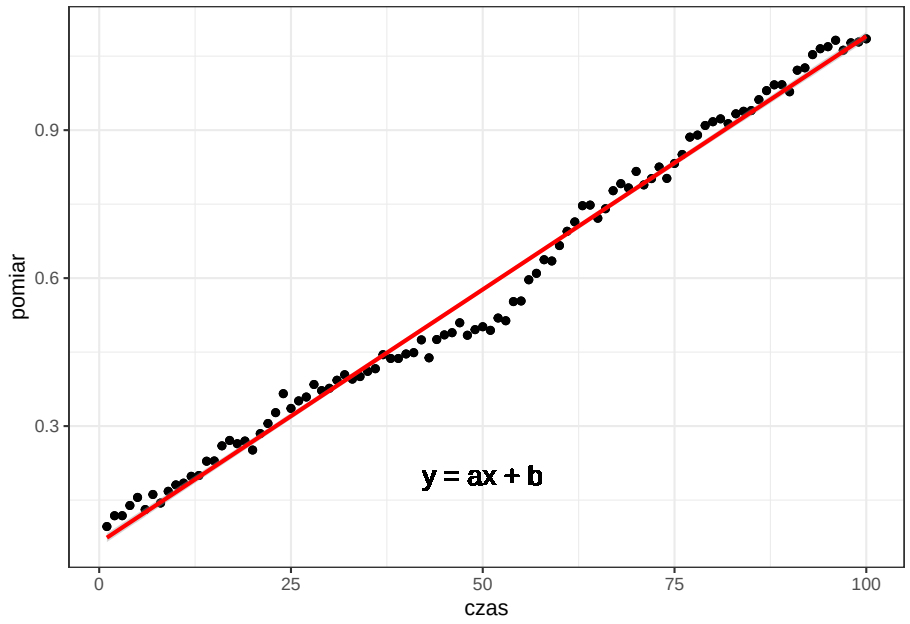
```
summary(lm(pomiar~czas, data=dane3_1))
```

```
##
## Call:
## lm(formula = pomiar ~ czas, data = dane3_1)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.094113 -0.014337  0.009274  0.023692  0.055815
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 6.352e-02  4.631e-03   13.72   <2e-16 ***
## czas        1.027e-02  7.962e-05  128.97   <2e-16 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0325 on 198 degrees of freedom
## Multiple R-squared:  0.9882, Adjusted R-squared:  0.9882
## F-statistic: 1.663e+04 on 1 and 198 DF,  p-value: < 2.2e-16
```

```
# dodanie tekstu do wykresu np. równanie opisujące linię trendu
p + geom_point() + stat_smooth(method = "lm", color = "red")+
  geom_text(data = NULL, x = 50, y = 0.2, label = "y = ax + b", size = 5)
```

```
## `geom_smooth()` using formula 'y ~ x'
```

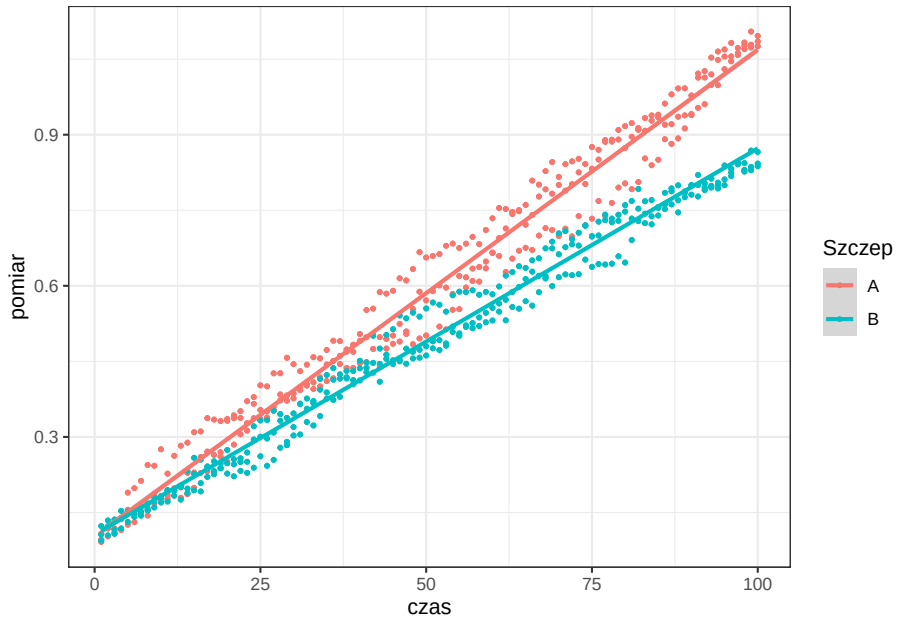


Możemy zastosować `stat_smooth` dla wszystkich danych.

`geom_point` można rozróżniać pod względem: color, shape, size, a `geom_line` pod względem: color i linetype.

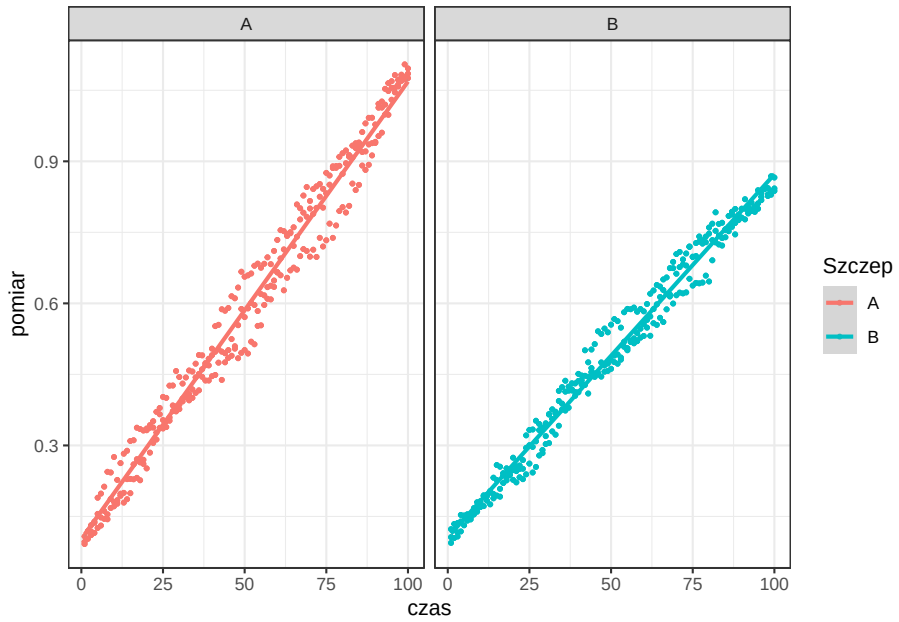
```
p <- ggplot(data=dane3, aes(x = czas, y = pomiar, color = Szczep))
p + geom_point(size = 0.75) + stat_smooth(method = "lm")
```

```
## `geom_smooth()` using formula 'y ~ x'
```



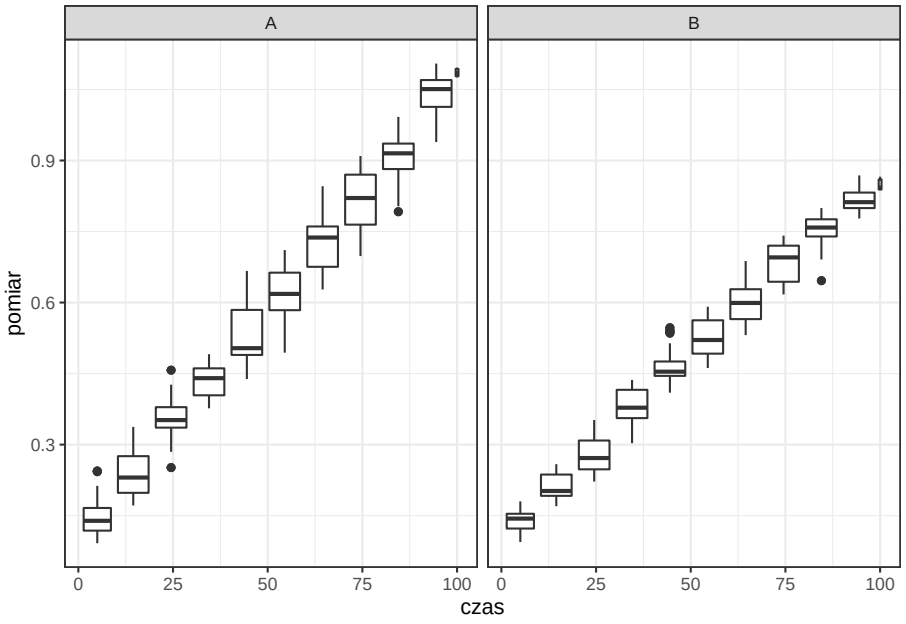

```
p + geom_point(size = 0.75) + stat_smooth(method = "lm") + facet_wrap(~Szczep)
```

```
## `geom_smooth()` using formula 'y ~ x'
```



Do przedstawienia takich danych można też użyć boxplot.

```
p <- ggplot(data=dane3, aes(x=czas, group = plyr::round_any(czas, 10, floor), y=pomiar)
p + geom_boxplot()+facet_wrap(~Szczep)
```



4.5.2 Jak poradzić sobie z nadmiarem punktów na wykresie (overplotting)?

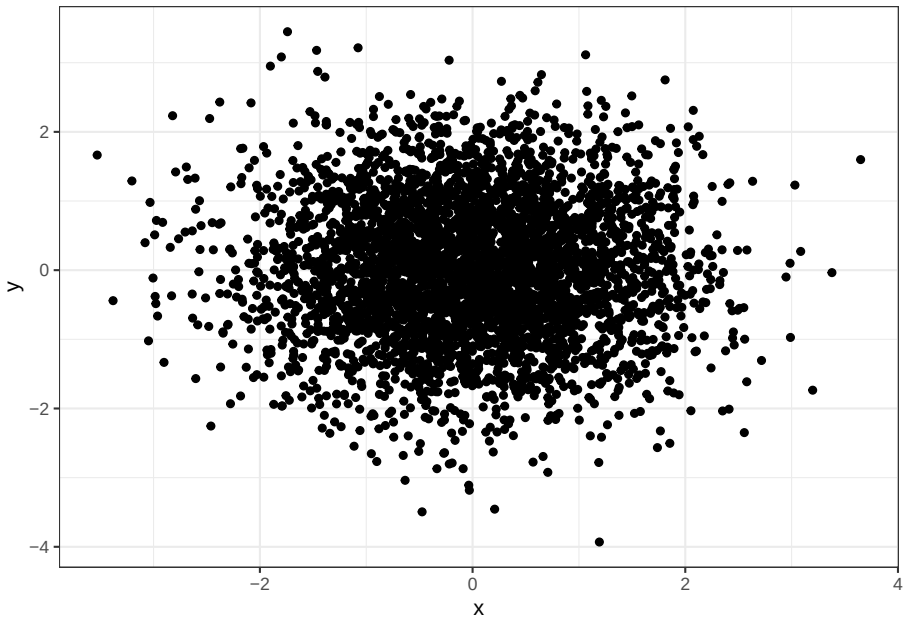
Jednym z problemów podczas stosowania wykresów punktowych może być zbyt duża liczba punktów powodująca zacieranie się informacji.

W ggplot2 możemy skorzystać z kilku sposobów poradzenia sobie z "overplotting". W przypadku danych ciągłych można wykorzystać parametr `alpha` pozwalający na ustawienie półprzezroczystych punktów albo wykorzystać punkty niewypełnione w środku - `shape = 1:14`.

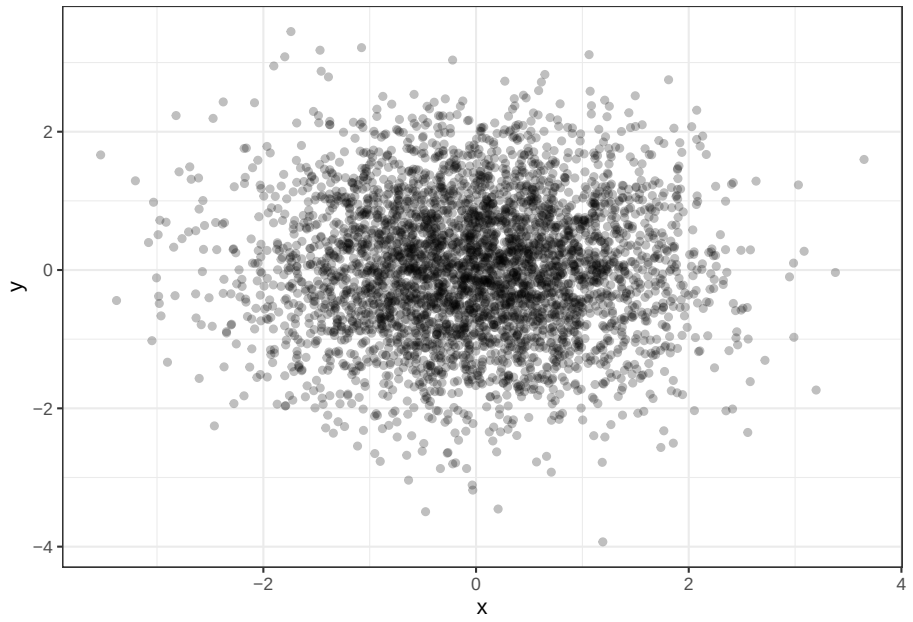
Innym sposobem może być wykorzystanie `geom_density2d`, który wylicza gęstość w układzie dwuwymiarowym i zaznacza liniami na wykresie albo `geom_bin2d` , który pozwala zrobić dwuwymiarowy histogram, możliwa kotrola `binwidth` (trzeba podać wektor dwóch wartości).

Dla danych dyskretnych pomóc może wykorzystanie `position="jitter"`, która pozwala losowo rozrzucić punkty albo zastosowanie parametru alpha oraz `stat_sum` - wielkość punktów zależy od ilości nakładających się punktów.

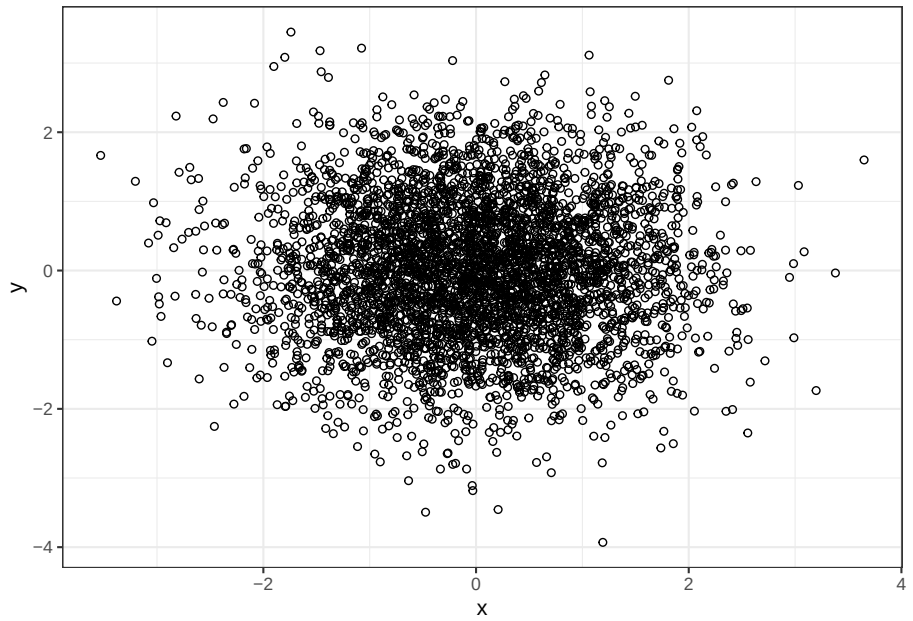
```
# przykładowe dane ciągłe

dane <- data.frame(x = rnorm(4000), y = rnorm(4000))

p <- ggplot(dane, aes(x = x,y = y))
p + geom_point()
```



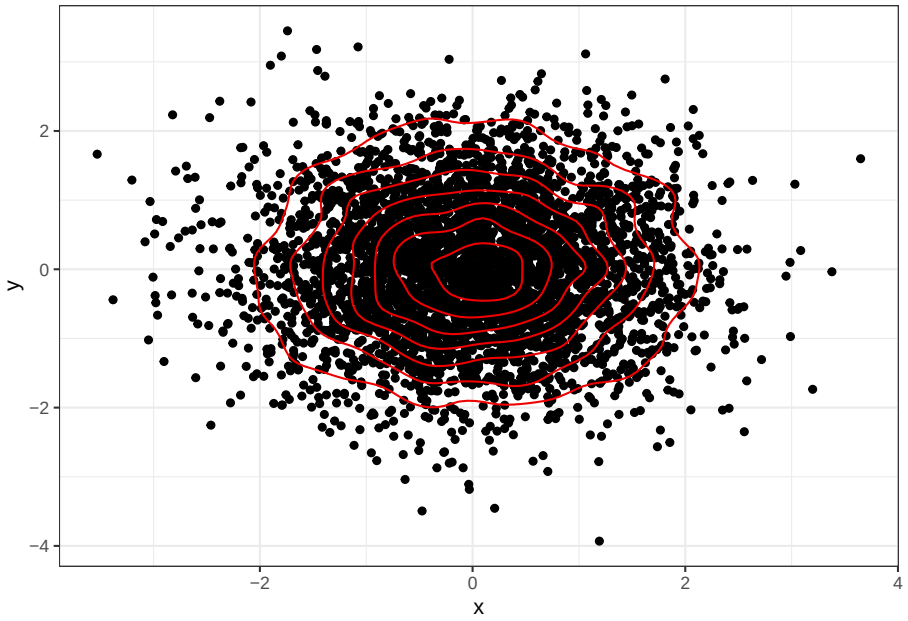

```
# zastosowanie alpha
p + geom_point(alpha = 0.25)
```



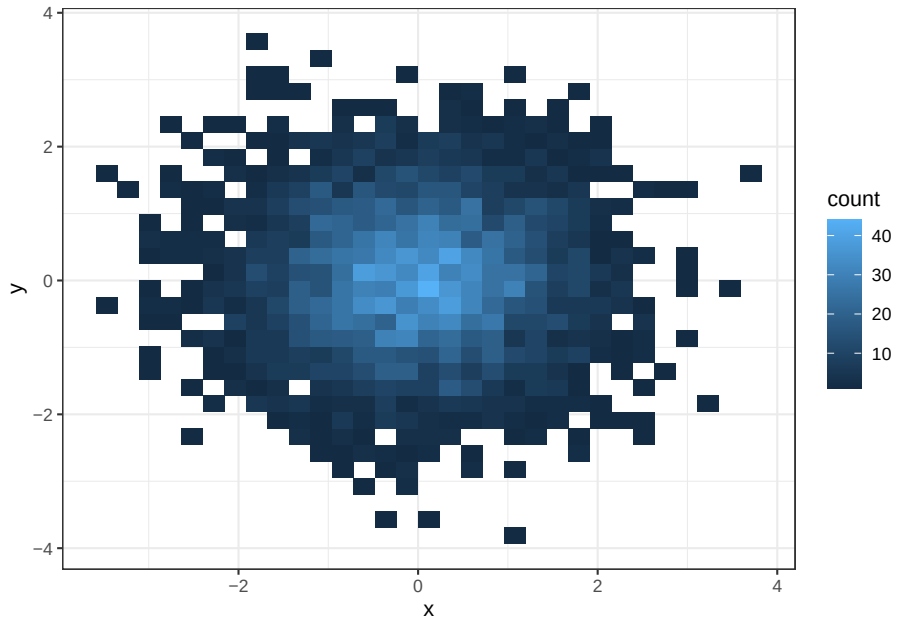

```
# puste punkty
p + geom_point(shape = 1)
```



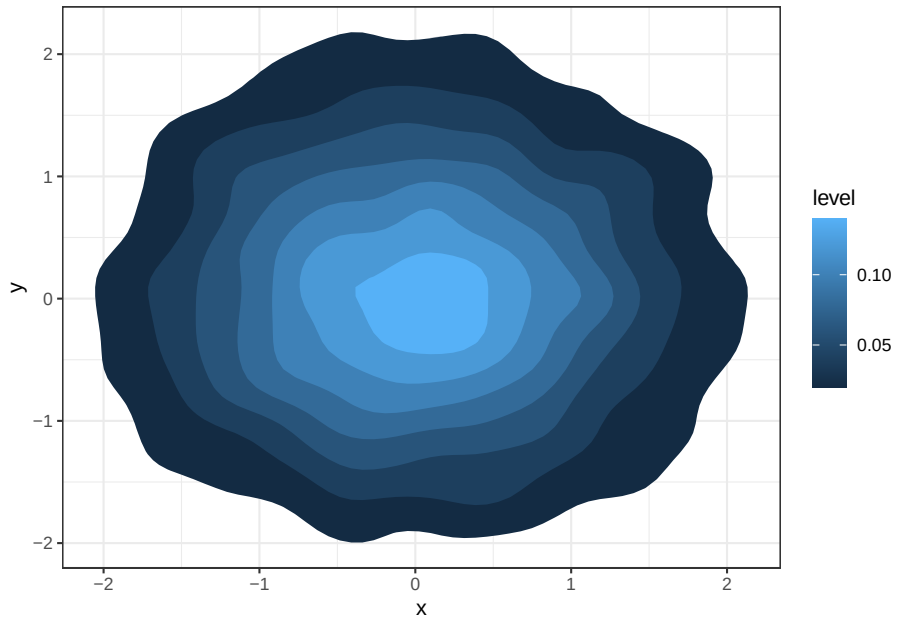
```
# geom_density2d albo geom_bin2d()
p + geom_point()+geom_density2d(color = "red2")
```



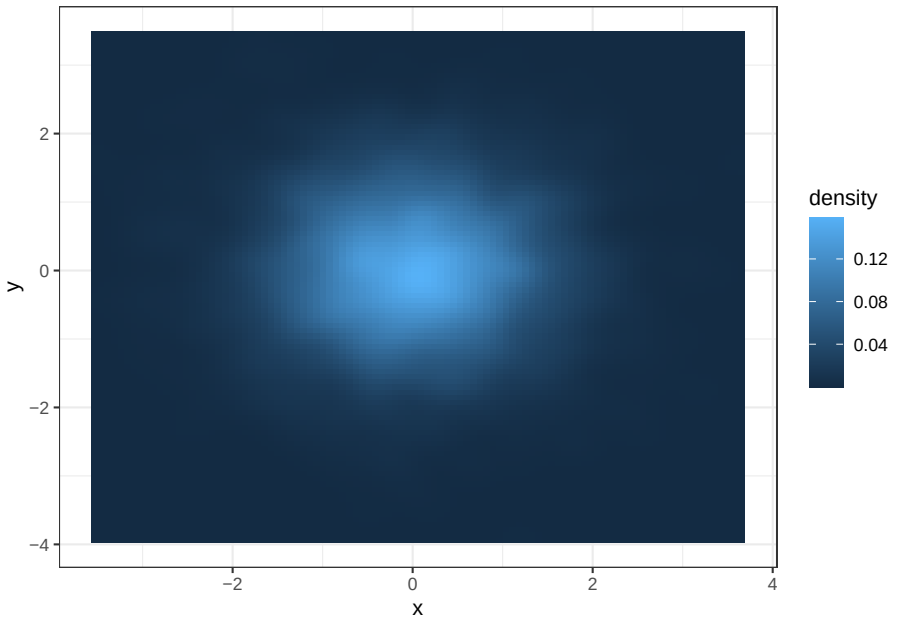

```
p + geom_bin2d()
```



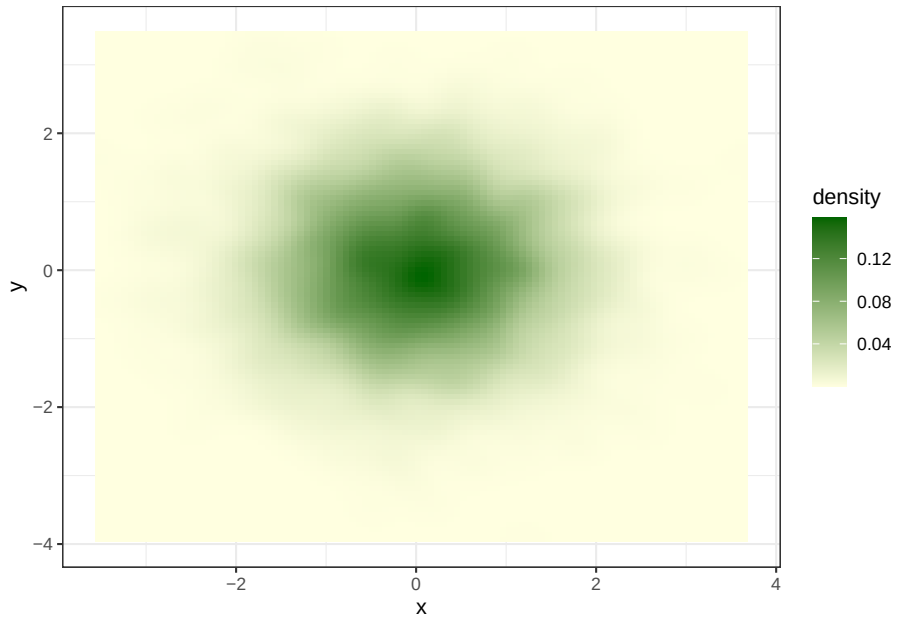

```
# można również użyć stat_density2d z innym geomem niż linia

p + stat_density2d(geom = "polygon", aes(fill  =  ..level..))
```



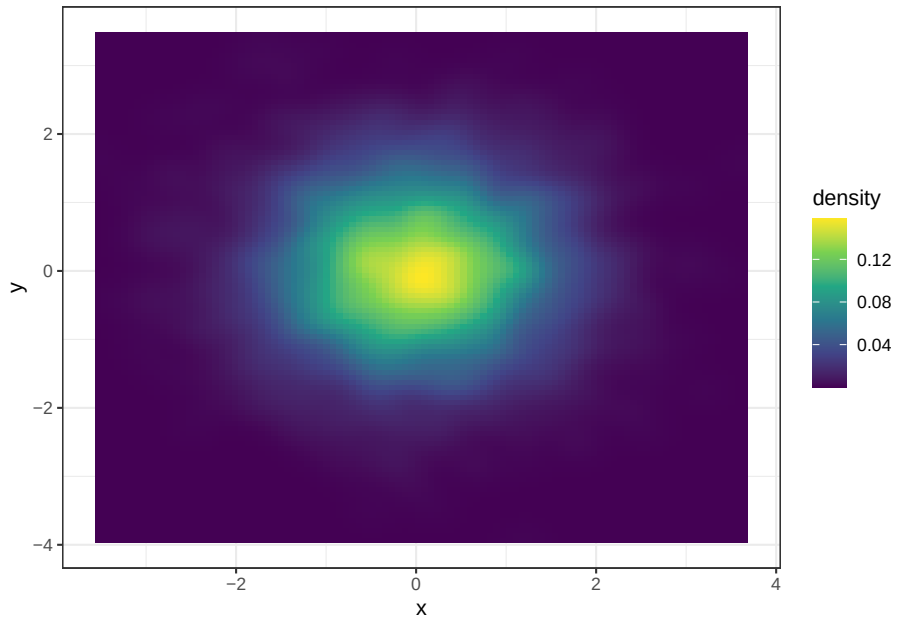
```
# linie widoczne na wykresach są wynikiem konwersji do pdf, nie będzie ich na wykresach zapisanyc
p + stat_density2d(geom = "tile", contour = FALSE, aes(fill = ..density..))
```



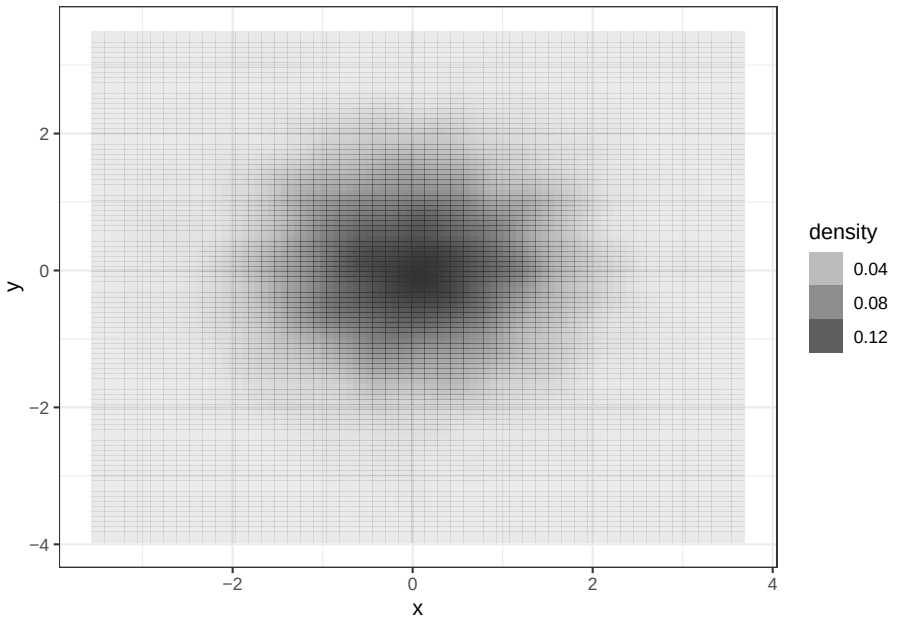

```
p + stat_density2d(geom = "tile", contour = FALSE, aes(fill = ..density..))+
  scale_fill_gradient(low = "lightyellow", high = "darkgreen")
```



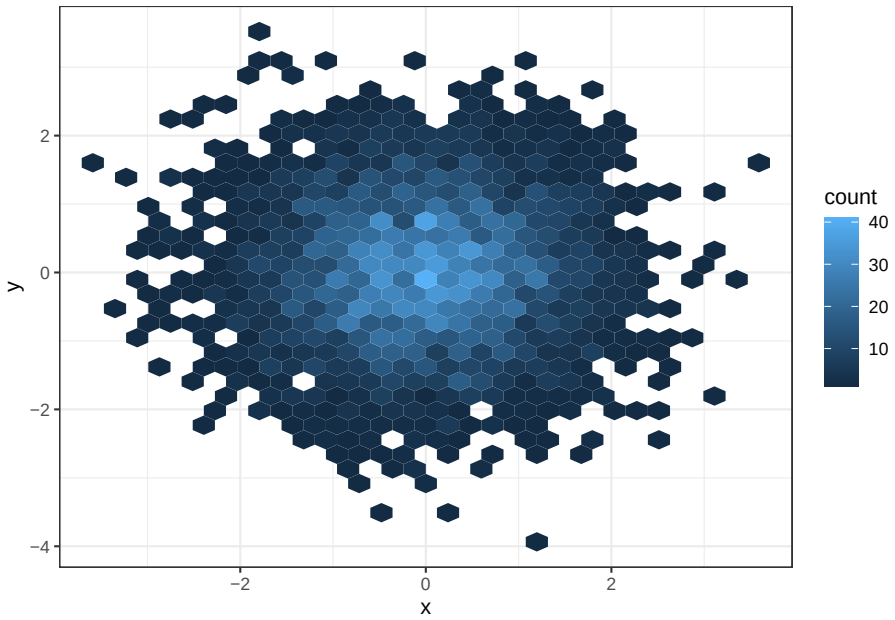

```
p + stat_density2d(geom = "tile", contour = FALSE, aes(fill = ..density..))+
  scale_fill_viridis_c()
```



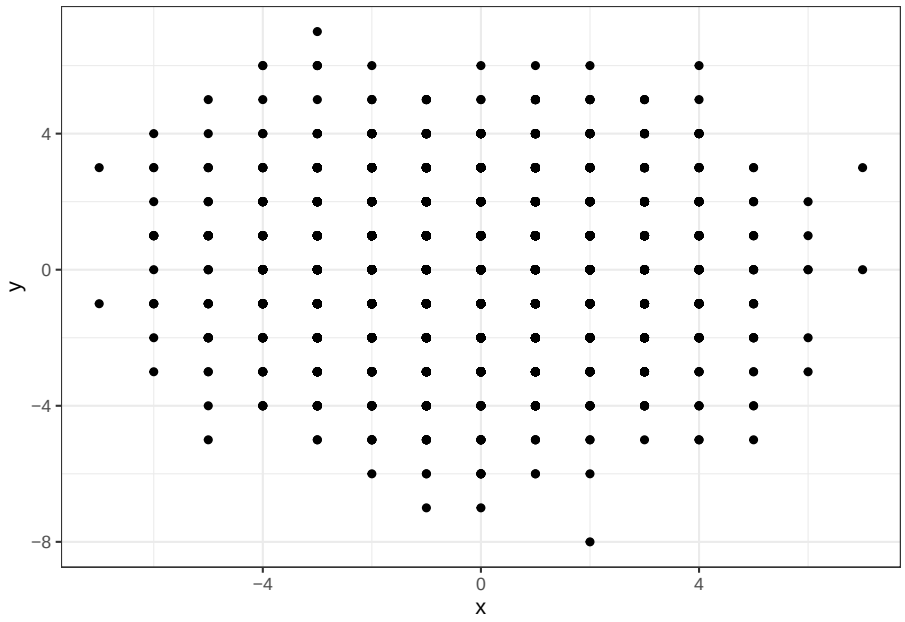
```
p + stat_density2d(geom = "tile", contour = FALSE, aes(alpha = ..density..))
```



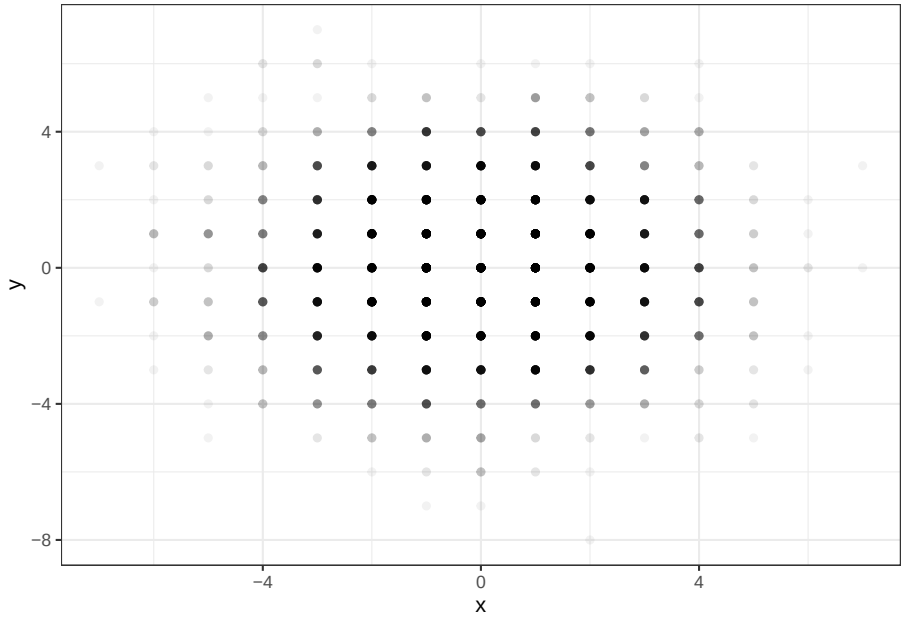

```
# zamiast kwadratów można użyć sześciokątów, wymaga zainstalowanie pakietu hexbin

library(hexbin)
p + stat_binhex()
```



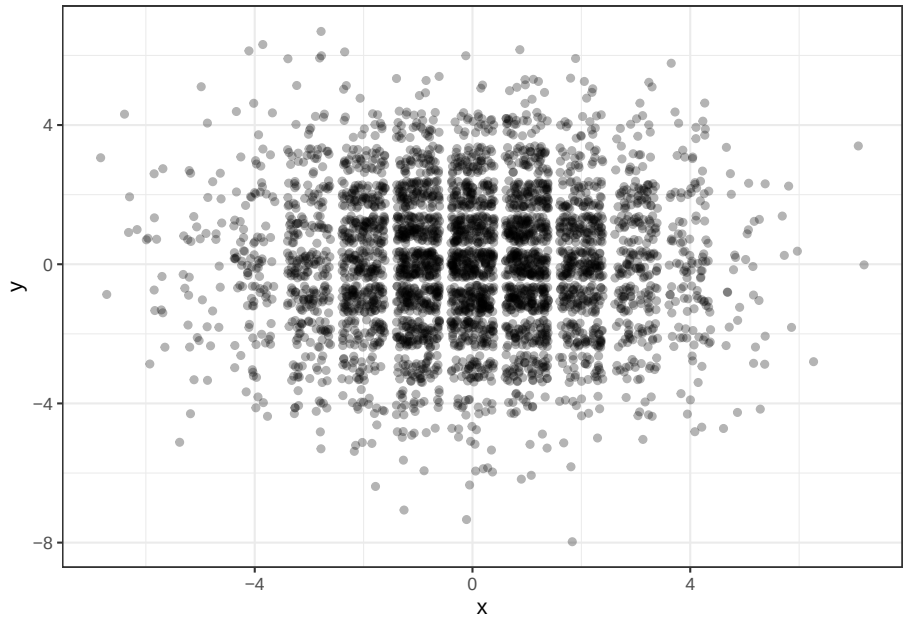

```
# dane dyskretne
dane <- round(2*dane)

p <- ggplot(dane, aes(x = x,y = y))
p + geom_point()
```



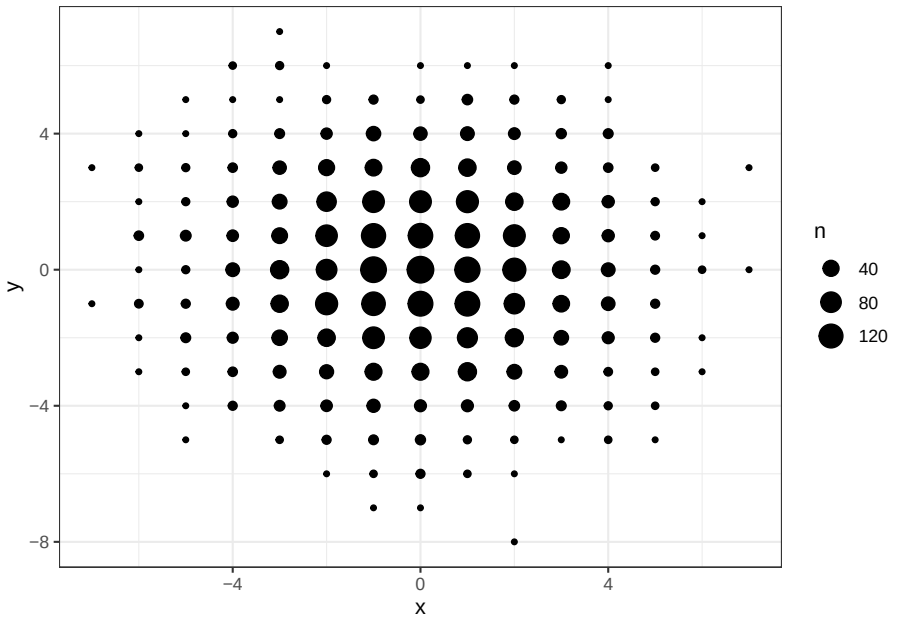

```
p + geom_point(alpha = 0.05)
```



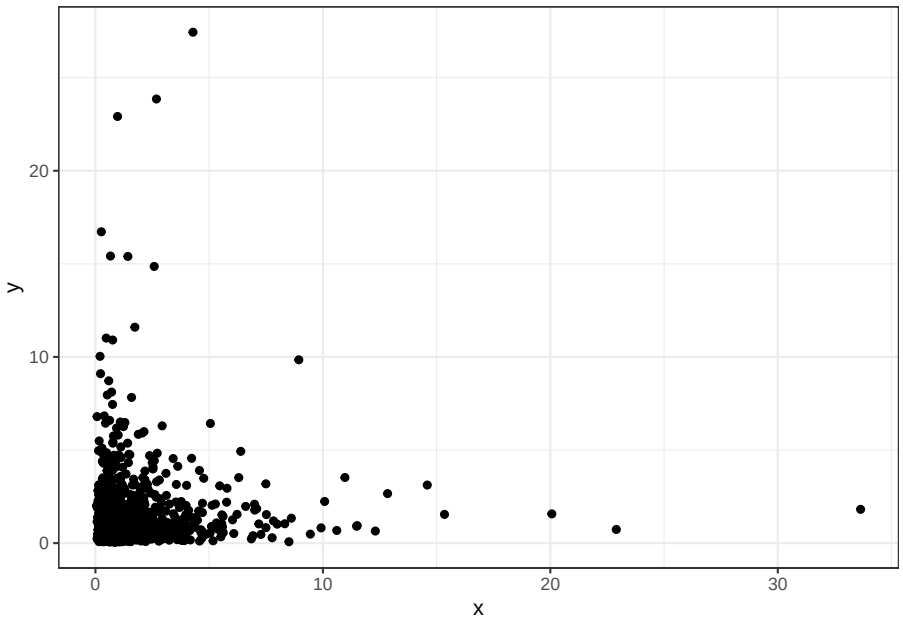
```
# wykorzystanie position = "jitter" i alpha
p + geom_point(position = "jitter", alpha = 0.3)
```



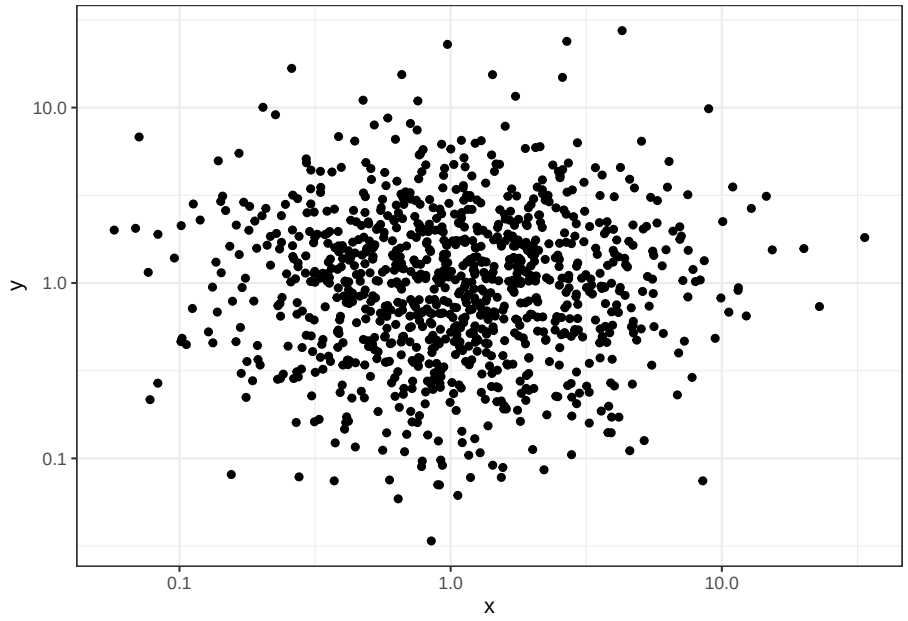

```
# z wykorzystaniem stat_sum
p + stat_sum()
```




```
# czasem pomóc może też zmiana skali na logarytmiczną
dane <- data.frame(x = rlnorm(1000), y = rlnorm(1000))
p <- ggplot(dane, aes(x = x,y = y))
p + geom_point()
```



```
p + geom_point()+scale_y_log10()+scale_x_log10()
```



4.5.3 Wykres wstążka (ribbon)

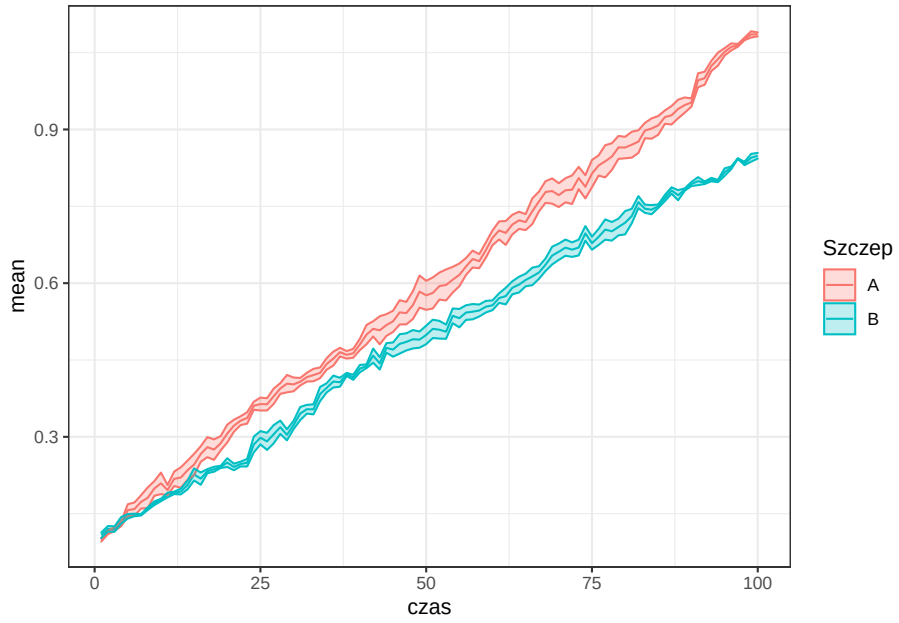
Użycie `geom_ribbon` wymaga wstępnego podsumowania danych np. przy użyciu dplyr - analogicznie jak dane1. W tym wypadku zmiennymi, pod względem których będziemy dzielić dane na grupy będą szczep i czas.

Półprzezroczystą wstążkę uzyskujemy dzięki parametrowi alpha.

```
library(dplyr)

summ <- dane3 %>% group_by(Szczep, czas) %>%
  summarize(mean = mean(pomiar),
            sd = sd(pomiar),
            blad = sd/sqrt(length(pomiar)),
            lower = mean-blad,
            upper = mean+blad)
```

```
## `summarise()` has grouped output by 'Szczep'. You can override using the `.groups` a
```

```
p <- ggplot(data = summ, aes(x = czas, y = mean, color = Szczep, fill = Szczep))
p + geom_line() + geom_ribbon(aes(ymin = lower, ymax = upper), alpha = 0.25)
```



4.6 Kolory, osie, panele

4.6.1 Porównywanie - fill, color, shape

Porównanie kilku szczepów można zrobić w ten sam sposób, wybierając w `aes` - `fill=Szczep` albo `color=Szczep` (w zależności od geomu, fill pasuje do histogram i density, color do density, line, point, boxplot itd.)

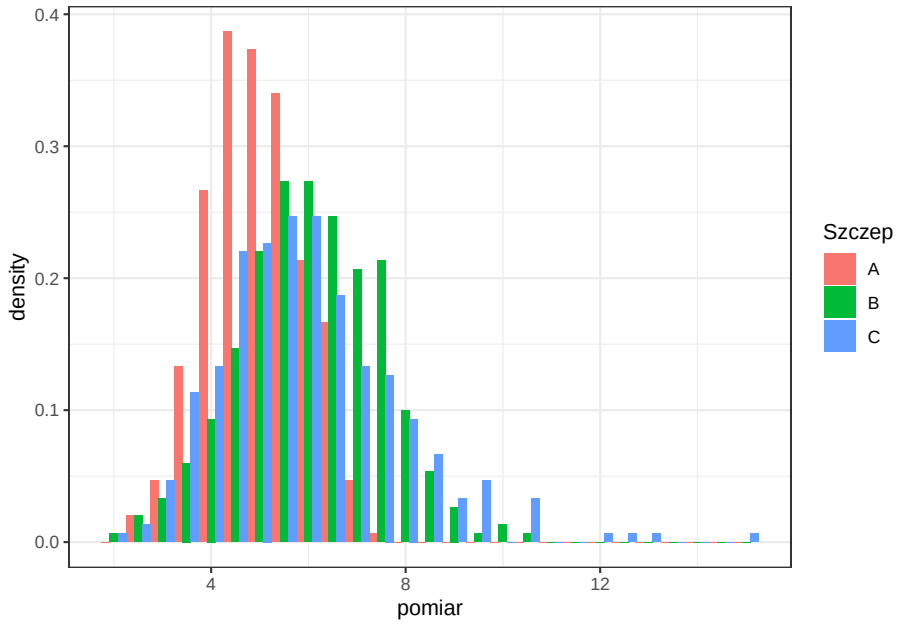
W histogramie domyślnie dostaniemy słupki ustawione jeden na drugim, jeżeli chcemy słupki ustawione obok siebie należy ustawić parametr `position="dodge"`.

W przypadku boxplot dla kilku cech można pod x podstawić Szczep - otrzymamy wykres zawierający po jednym boxplocie dla jednego szczepu.

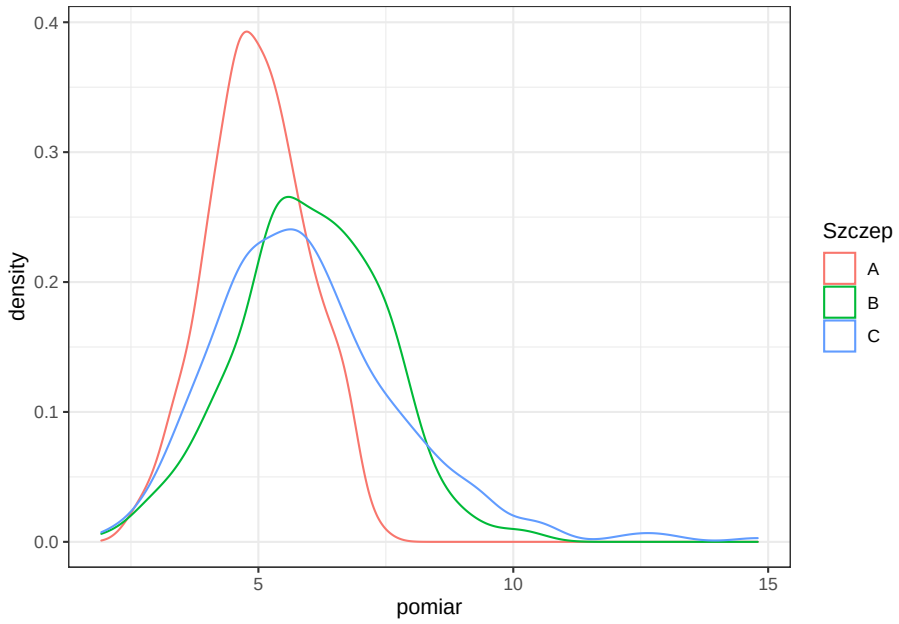
Wybieram dane tylko dla warunków 1.

```
dane1_2 <- dane1 %>% filter(warunki == 1)

p <- ggplot(data = dane1_2, aes(x = pomiar))
```

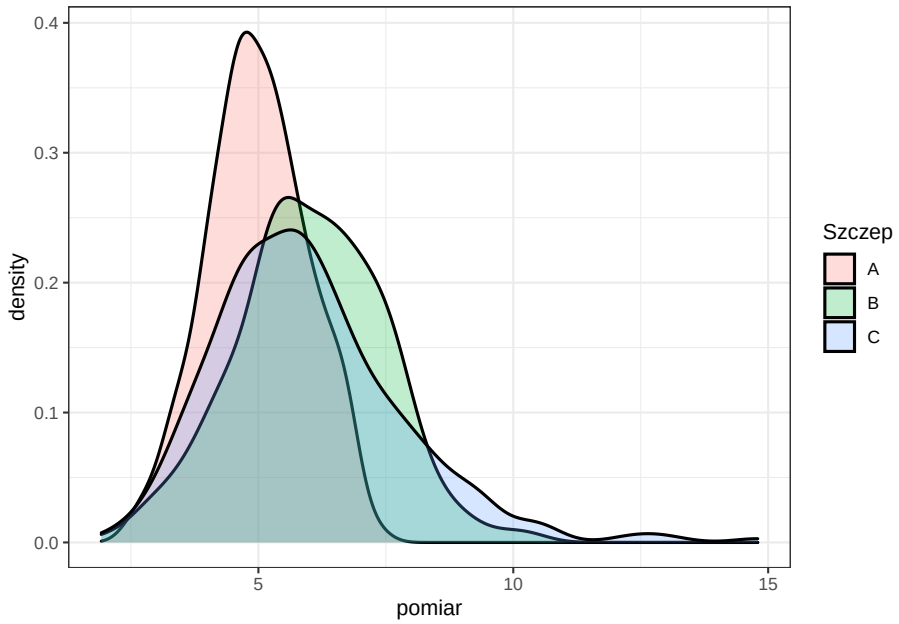
```
# Histogram dla trzech szczepów
p + geom_histogram(aes(fill = Szczep, y = ..density..), binwidth = 0.5,
                   position="dodge")
```



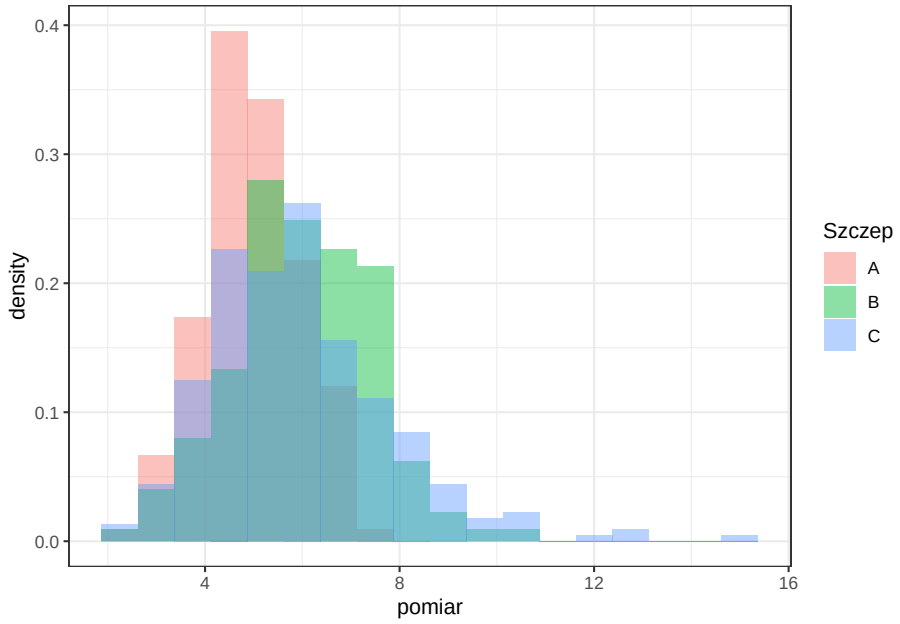

```
# Gęstość dla trzech szczepów rozróżniona przez color
p + geom_density(aes(color = Szczep))
```



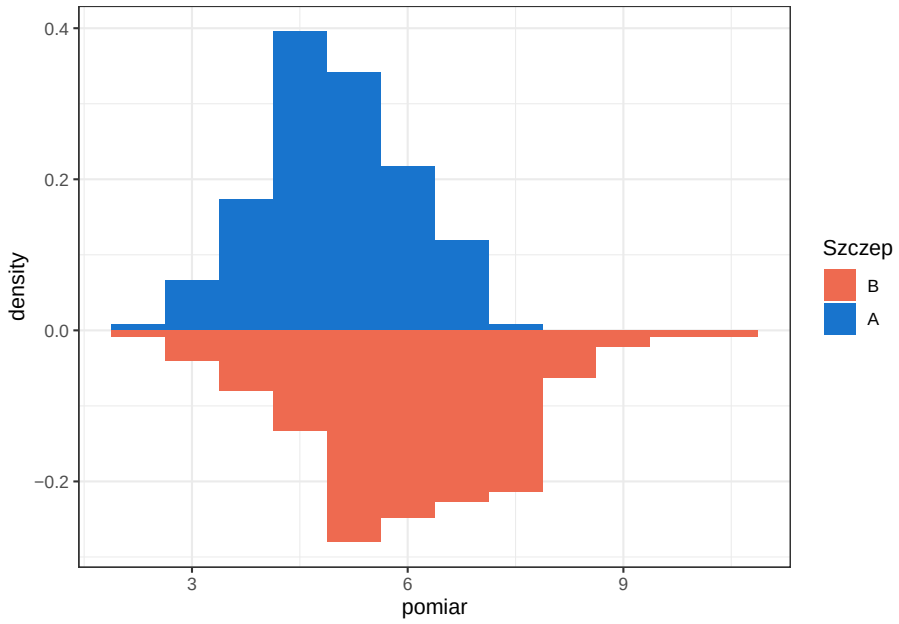

```
# Gęstość rozróżniona przez fill, półprzezroczystość uzyskujemy parametrem alpha
p + geom_density(aes(fill = Szczep), alpha = 0.25, size = 0.75)
```



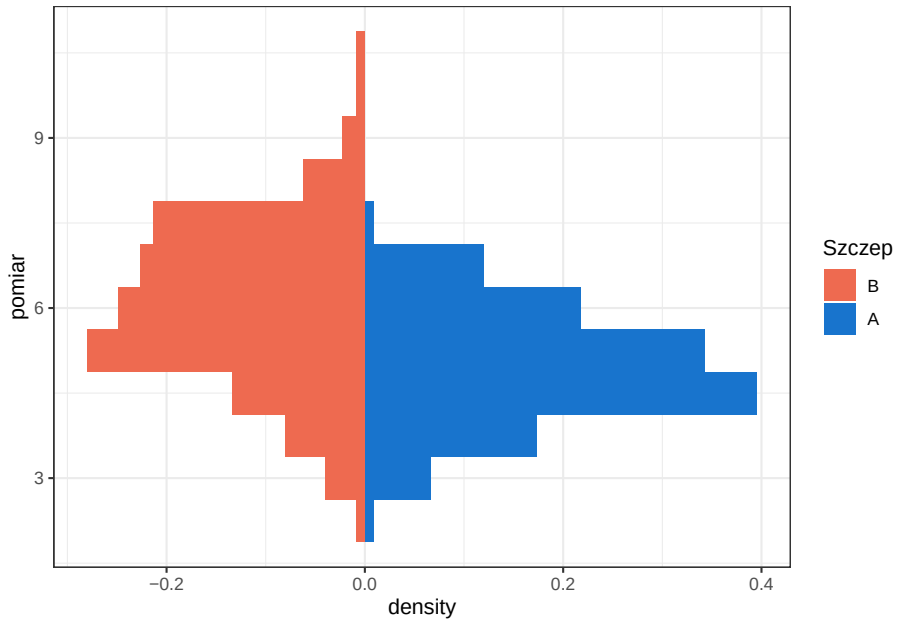
```
# Histogramy nałożone na siebie - wymaga ustawienia position = "identity"
p <- ggplot(data=dane1_2, aes(x = pomiar))
p + geom_histogram(aes(y = ..density.., fill = Szczep), binwidth = 0.75, alpha = 0.45,
```



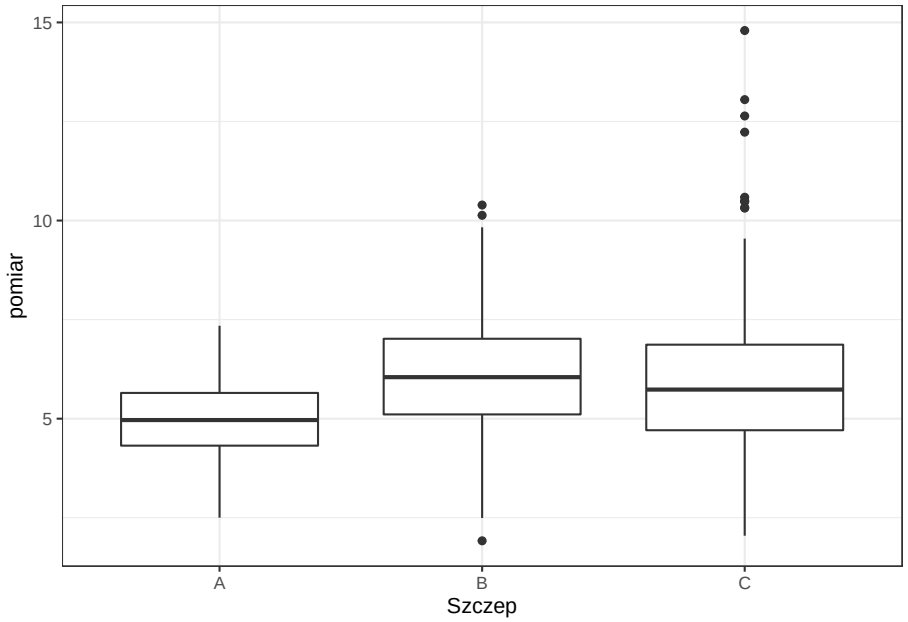

```
# Histogram "back to back" wymaga ustawienia w drugim geom_histogram ujemnego y
dane1_3 <- dane1 %>% filter(Szczep == "B", warunki == 1)
p <- ggplot(data = dane1_1, aes(x = pomiar))
p <- p + geom_histogram(aes(y = ..density.., fill = "dodgerblue3"), binwidth = 0.75)+
  geom_histogram(data = dane1_3, aes(x = pomiar, y = -..density.., fill = "coral2"),
                 binwidth = 0.75)+
  scale_fill_manual(name = "Szczep",
                    values = c("coral2" = "coral2", "dodgerblue3" = "dodgerblue3"),
                    labels = c("B","A"))
p
```


0.4
0.2
0.0
−0.2
density
3
6
9
pomiar
Szczep
B
A

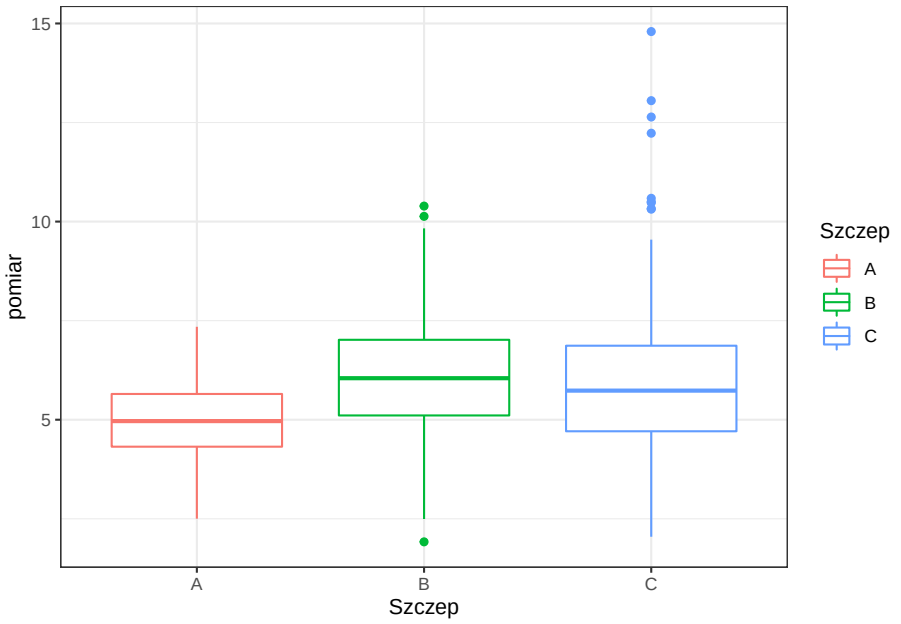
```
p + coord_flip()
```


9
6
3
pomiar
−0.2
0.0
0.2
0.4
density
Szczep
B
A

```
# Boxplot dla każdego szczepu
p <- ggplot(data = dane1_2, aes(x = Szczep, y = pomiar))
p + geom_boxplot()
```




```
# Boxploty też mogą być kolorowe ;)
p + geom_boxplot(aes(color = Szczep))
```



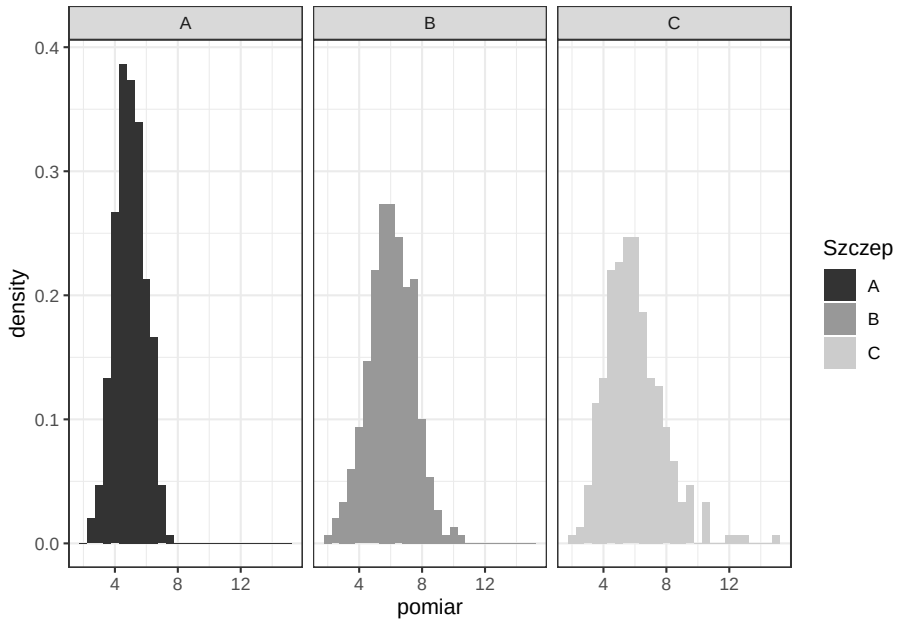
4.6.2 Zmiana skali kolorów

Możemy zmienić domyślne kolory wykresu korzystając z funkcji `scale_colour\fill_sth`. Rodzaj funkcji zależy od rodzaju danych - ilościowe albo jakościowe.

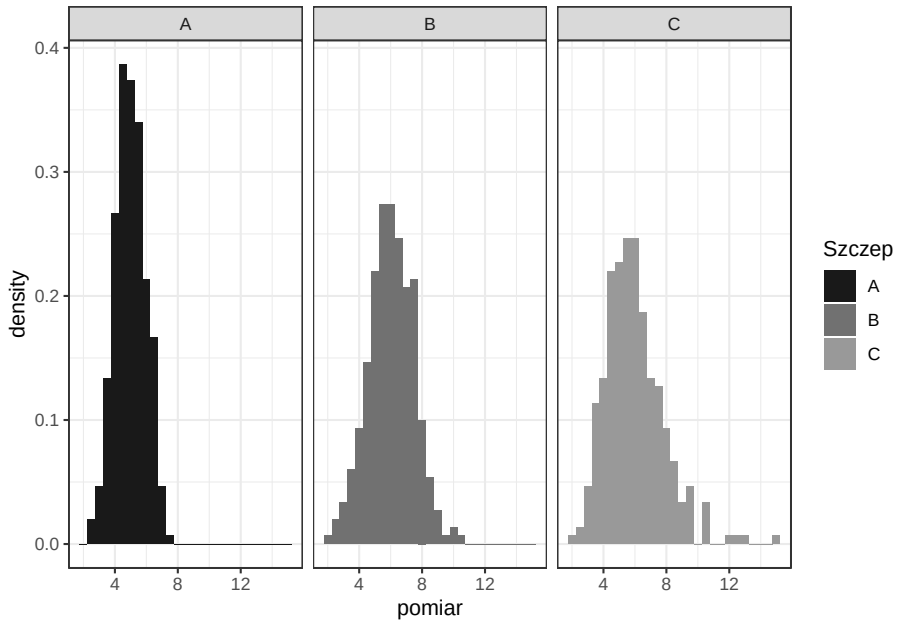
Dla danych jakościowych można użyć: `scale_color_grey` - odcienie szarości, `scale_colour_brewer` - zawiera zestawy kolorów ze strony Color Brewer, `scale_color_viridis_d` - zawiera palety viridis, `scale_color_hue` - pozwala na wybranie kolorów korzystając z palety HCL. Można też ustawić własny zestaw kolorów korzystając z nazw kolorów R albo np. przez RGB, korzystając z `scale_colour_manual`

Dla danych ilościowych można wybrać: `scale_color_gradient` - gradient między dwoma kolorami, `scale_color_gradient` - gradient pomiędzy 3 kolorami, `scale_color_gradientn` - gradient pomiędzy dowolną liczbą kolorów albo `scale_color_viridis_c`

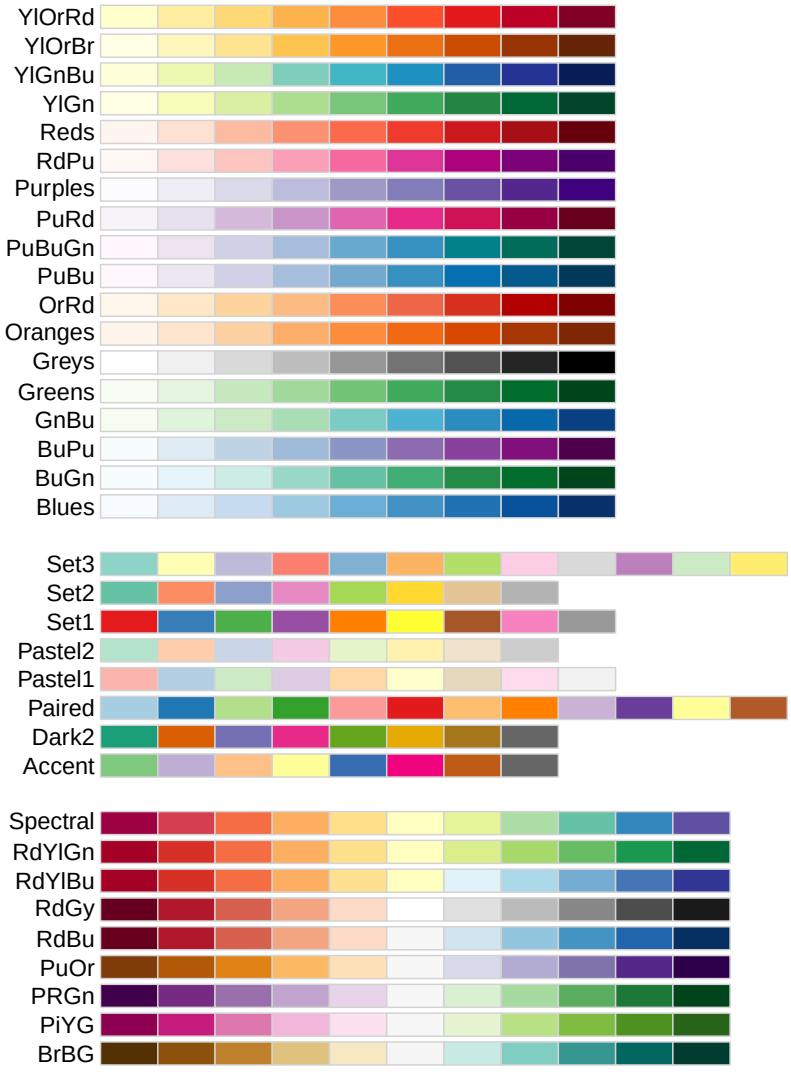
```
# Zmienne jakościowe

# Odcienie szarości
p <- ggplot(data = dane1_2, aes(x = pomiar))
p + geom_histogram(aes(fill = Szczep, y = ..density..), binwidth = 0.5)+
  facet_wrap(~Szczep) + scale_fill_grey()
```



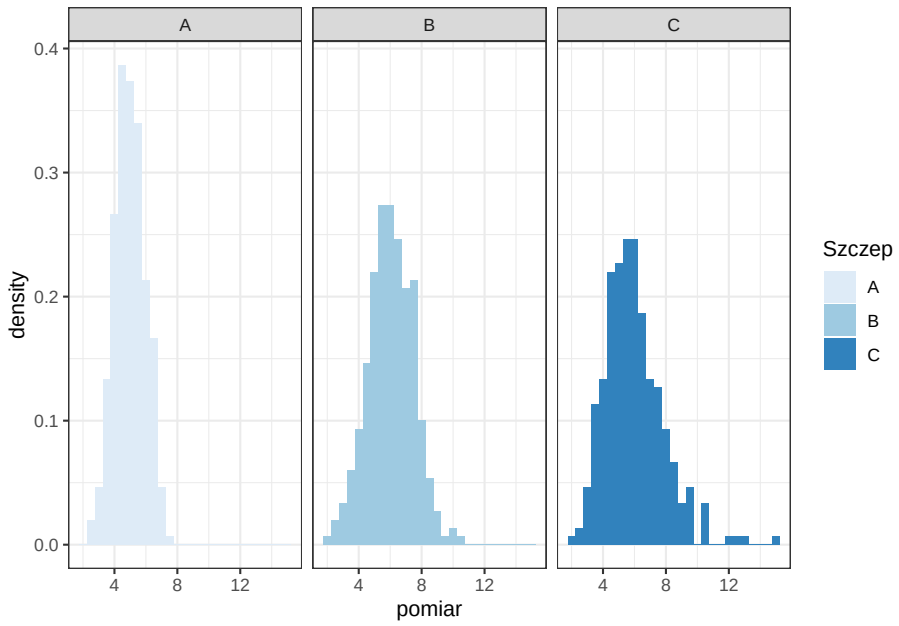

```
# można wybrać początek i koniec skali

p + geom_histogram(aes(fill = Szczep, y = ..density..), binwidth = 0.5)+
  facet_wrap(~Szczep) + scale_fill_grey(start = 0.1, end = 0.6)
```



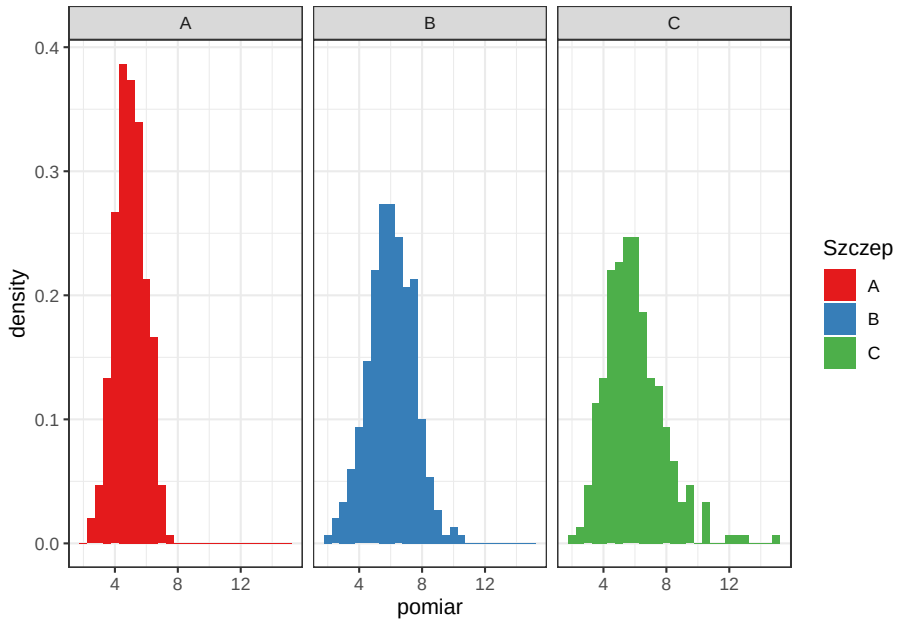
```
# skala ColorBrewer
library(RColorBrewer)
display.brewer.all()
```


YlOrRd
YlOrBr
YlGnBu
YlGn
Reds
RdPu
Purples
PuRd
PuBuGn
PuBu
OrRd
Oranges
Greys
Greens
GnBu
BuPu
BuGn
Blues
Set3
Set2
Set1
Pastel2
Pastel1
Paired
Dark2
Accent
Spectral
RdYlGn
RdYlBu
RdGy
RdBu
PuOr
PRGn
PiYG
BrBG

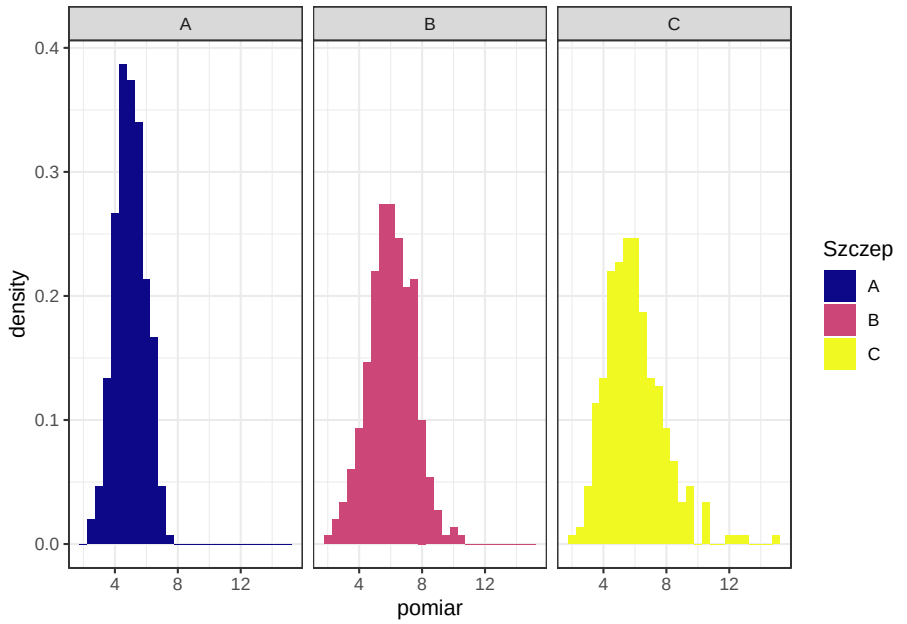
```
# z użyciem ColorBrewer

p <- ggplot(data = dane1_2, aes(x = pomiar))
p + geom_histogram(aes(fill = Szczep, y = ..density..), binwidth = 0.5)+
  facet_wrap(~Szczep) + scale_fill_brewer()
```



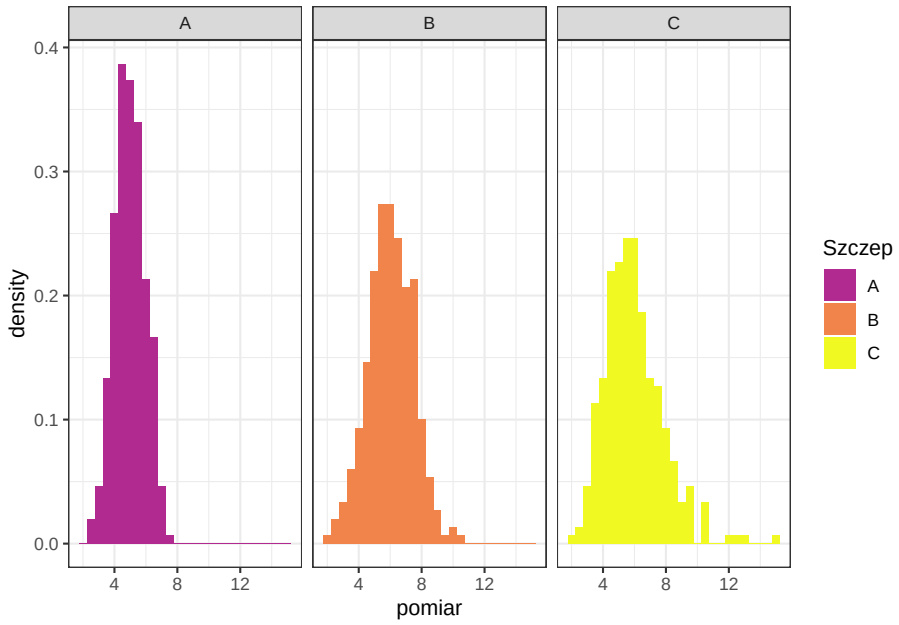

```
p + geom_histogram(aes(fill = Szczep, y = ..density..), binwidth = 0.5)+
  facet_wrap(~Szczep) + scale_fill_brewer(palette = "Set1")
```



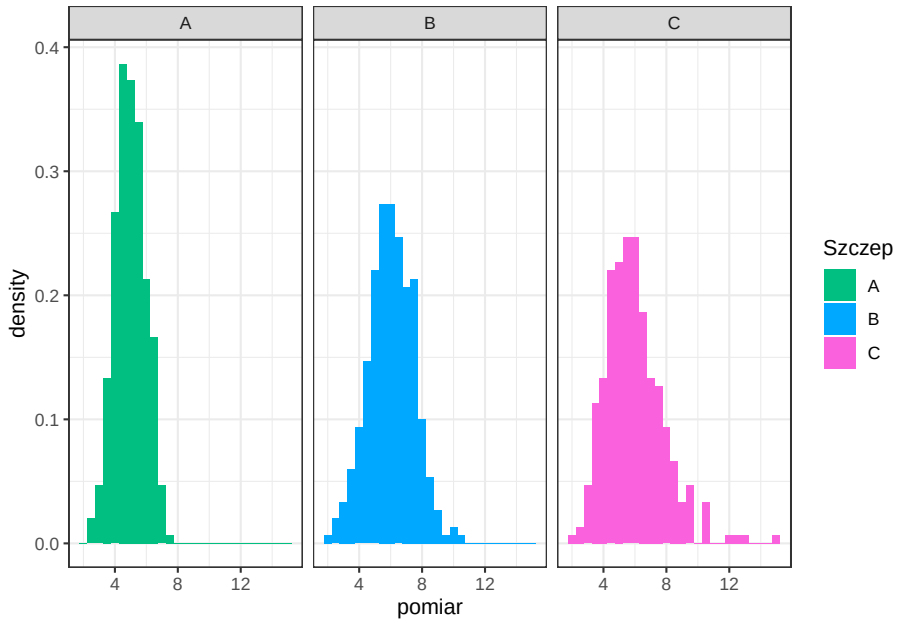

```
# z użyciem viridis, argument option przyjmuje wartości od A do H

p + geom_histogram(aes(fill = Szczep, y = ..density..), binwidth = 0.5)+
  facet_wrap(~Szczep) + scale_fill_viridis_d(option = 'C')
```



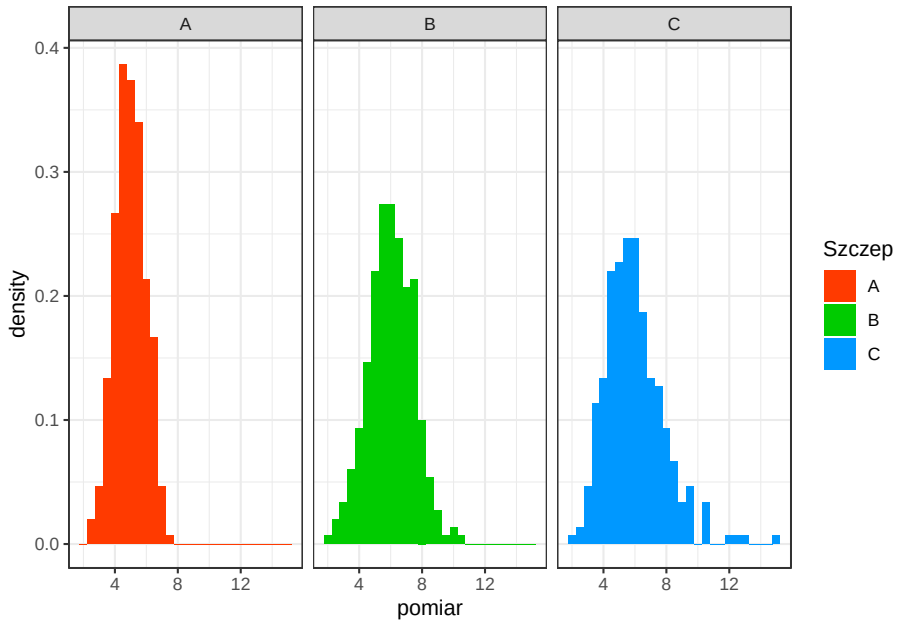
```
# zakres palety można modyfikować przy pomocy argumentów begin i end

p + geom_histogram(aes(fill = Szczep, y = ..density..), binwidth = 0.5)+
  facet_wrap(~Szczep) + scale_fill_viridis_d(option = 'C', begin = 0.4)
```



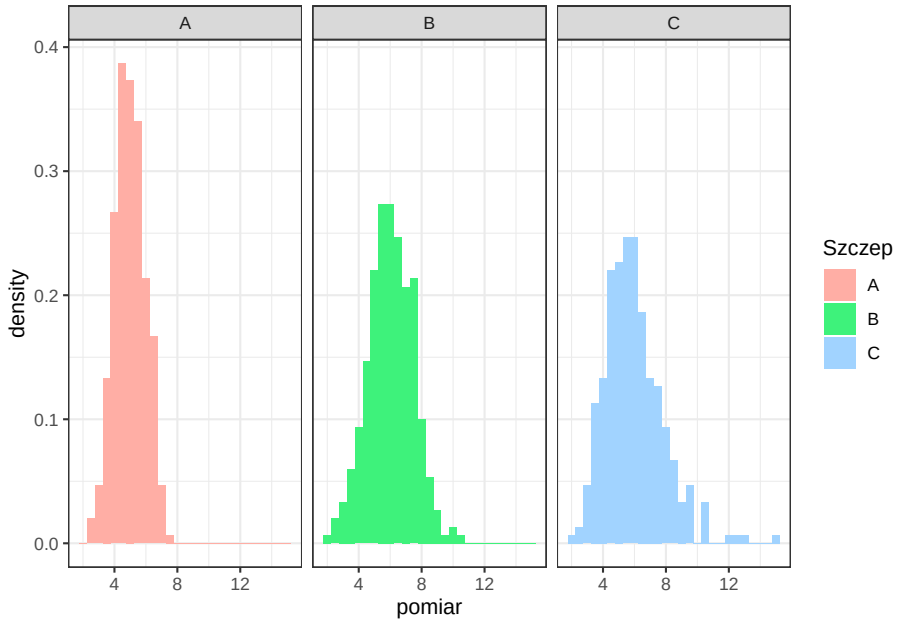

```
# skala z użyciem palety HCL - ustawiamy trzy argumenty: h(zakres barw), c(intensywność) i l(jas

p + geom_histogram(aes(fill = Szczep, y = ..density..), binwidth = 0.5)+
  facet_wrap(~Szczep) + scale_fill_hue(h = c(160,320))
```



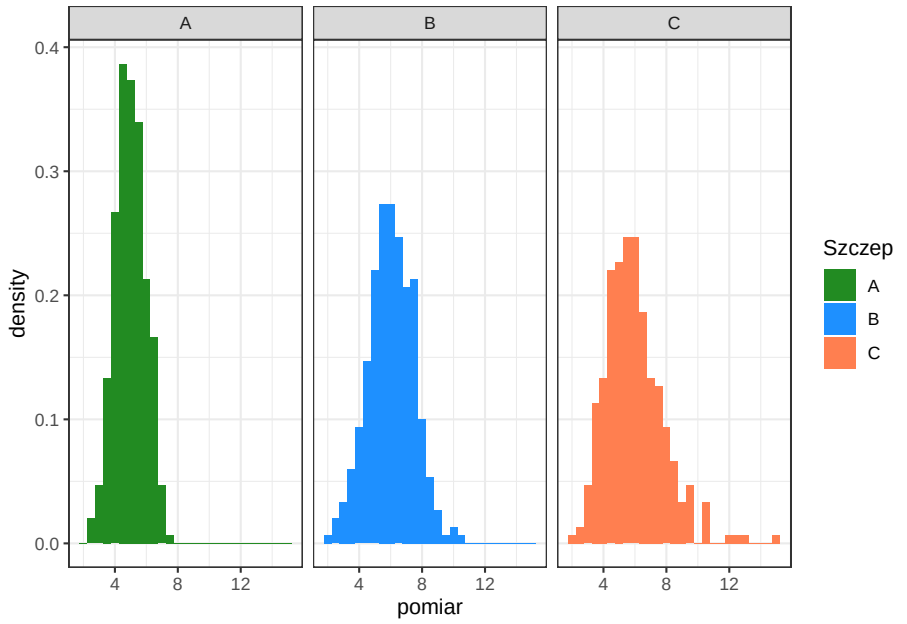

```
p + geom_histogram(aes(fill = Szczep, y = ..density..), binwidth = 0.5)+
  facet_wrap(~Szczep) + scale_fill_hue(c = 200)
```



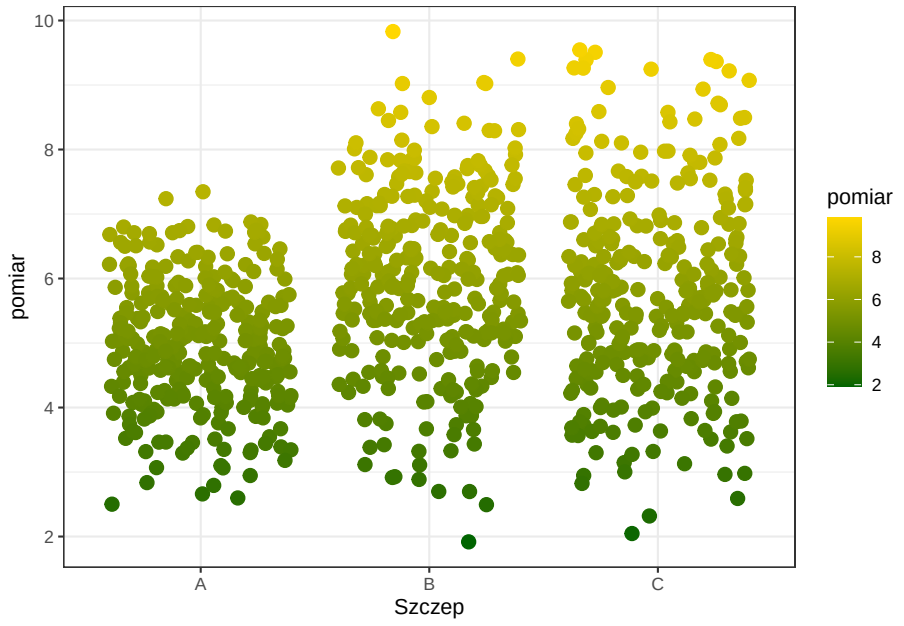
```
p + geom_histogram(aes(fill = Szczep, y = ..density..), binwidth = 0.5)+
  facet_wrap(~Szczep) + scale_fill_hue(l = 85)
```



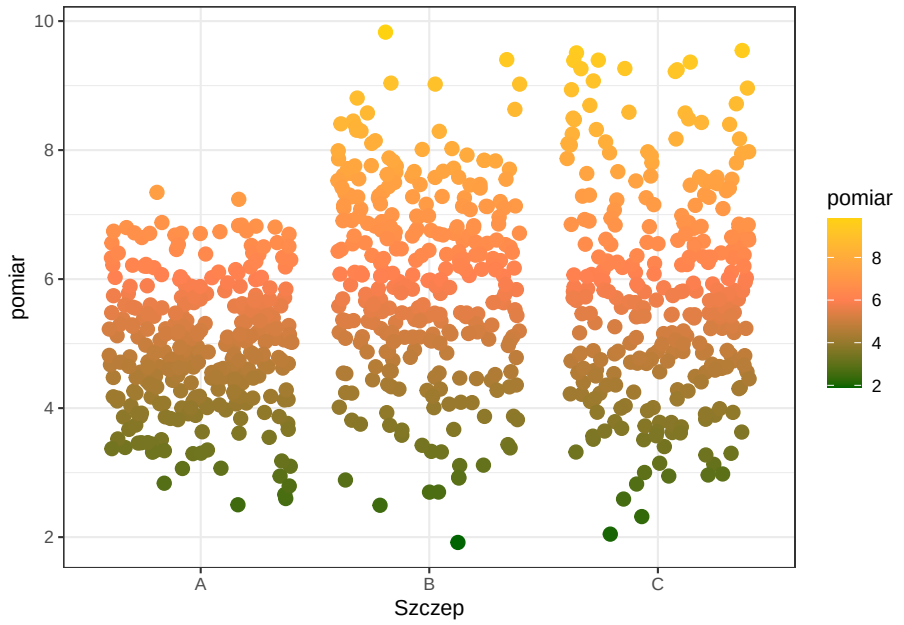

```
# skala manualna

p + geom_histogram(aes(fill = Szczep, y = ..density..), binwidth = 0.5)+
  facet_wrap(~Szczep) + scale_fill_manual(values = c("forestgreen", "dodgerblue", "coral"))
```



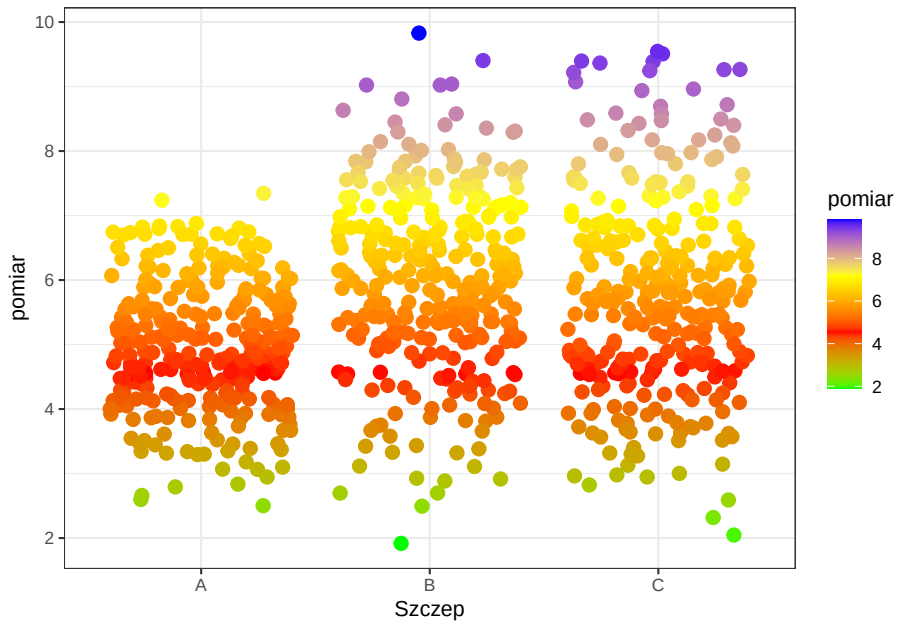

```
# Zmienne ilościowe

# domyślny gradient
p <- ggplot(subset(dane1_2, pomiar < 10))
p  <- p + geom_jitter(aes(x = Szczep, y = pomiar, color = pomiar), size = 3)

# dwa kolory
p + scale_color_gradient(low = "darkgreen", high = "gold")
```



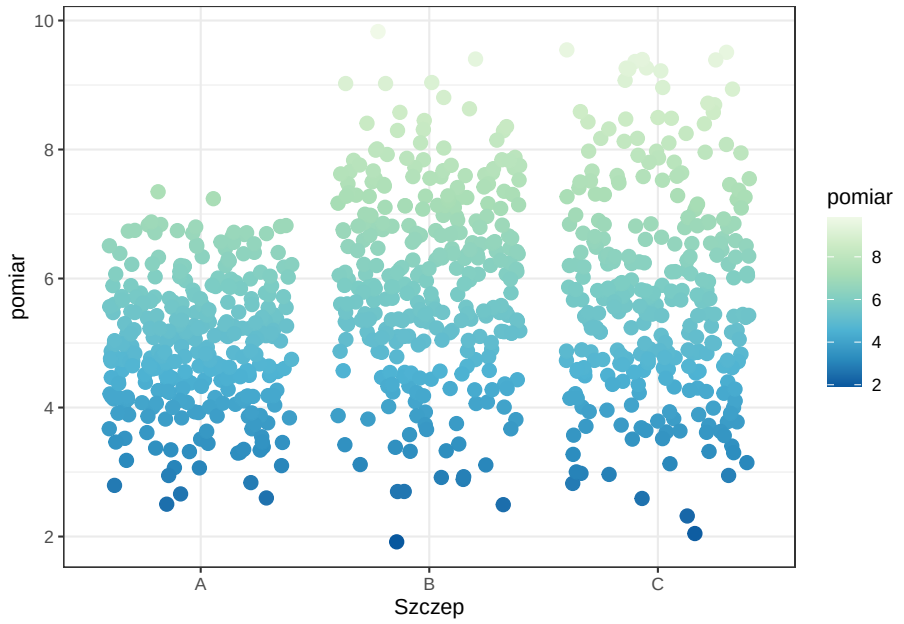

```
# trzy kolory - domyślnie założy, że wartość środkowa to zero, jeżeli jest inaczej to trzeba to p
p + scale_color_gradient2(low = "darkgreen", mid = "coral", high = "gold", midpoint = 6)
```



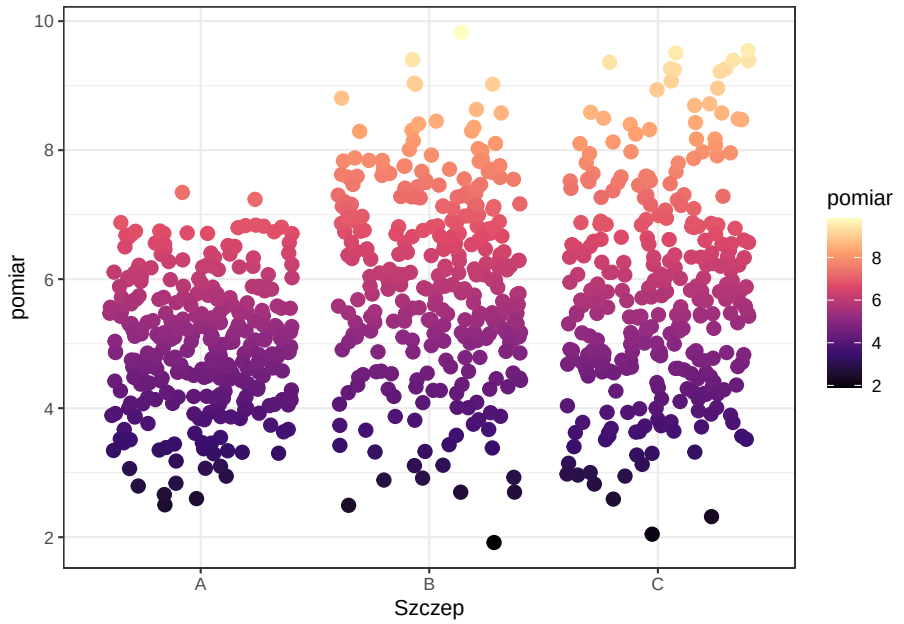
```
# więcej kolorów
p + scale_color_gradientn(colours = c("green", "red", "yellow", "blue"))
```



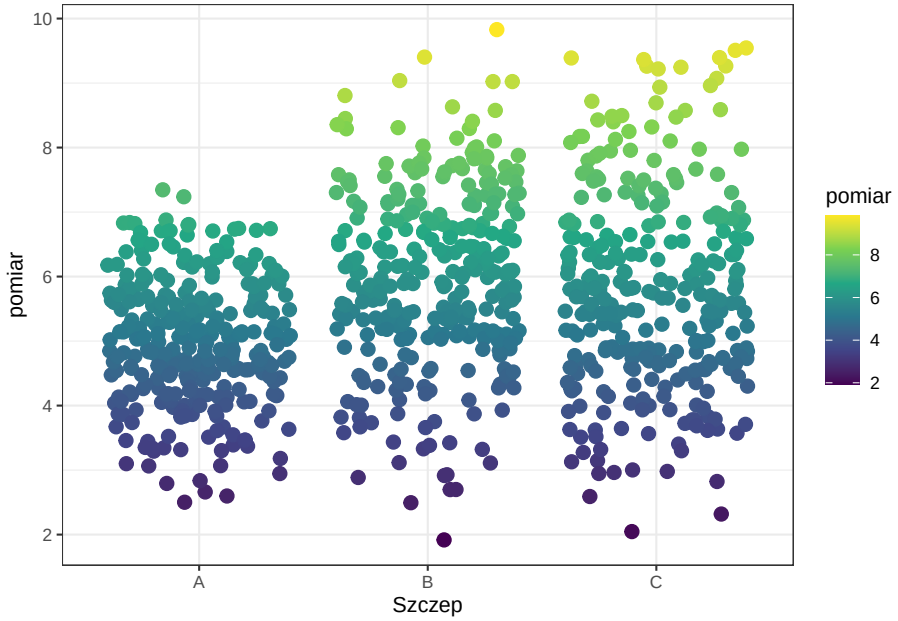

```
# ColorBrewer

p + scale_color_distiller(palette = 4)
```



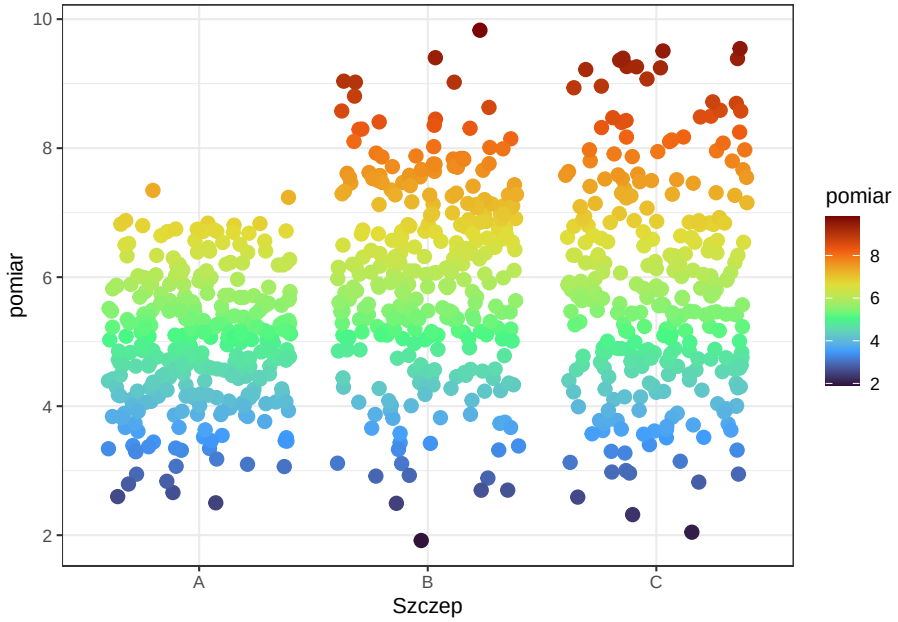

```
# viridis

p + scale_color_viridis_c(option = 'A')
```



```
p + scale_color_viridis_c(option = 'D')
```




```
p + scale_color_viridis_c(option = 'H')
```



Analogicznie do kolorów można ustawiać skale dotyczące kształtu punktów, stylu linii itp.

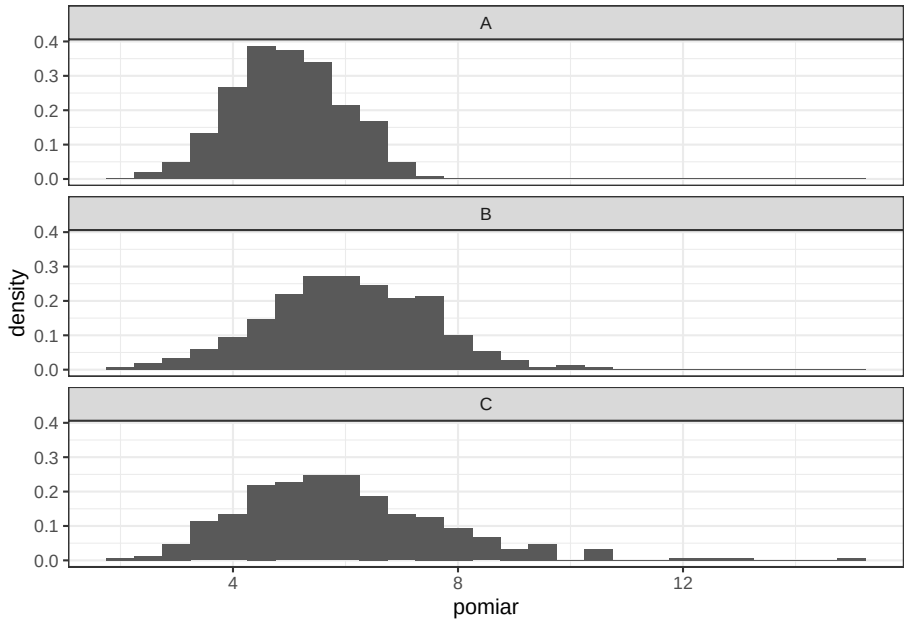
4.6.3 Podział wykresu na panele

Innym sposobem na rozróżnianie danych jest użycie paneli (facet).

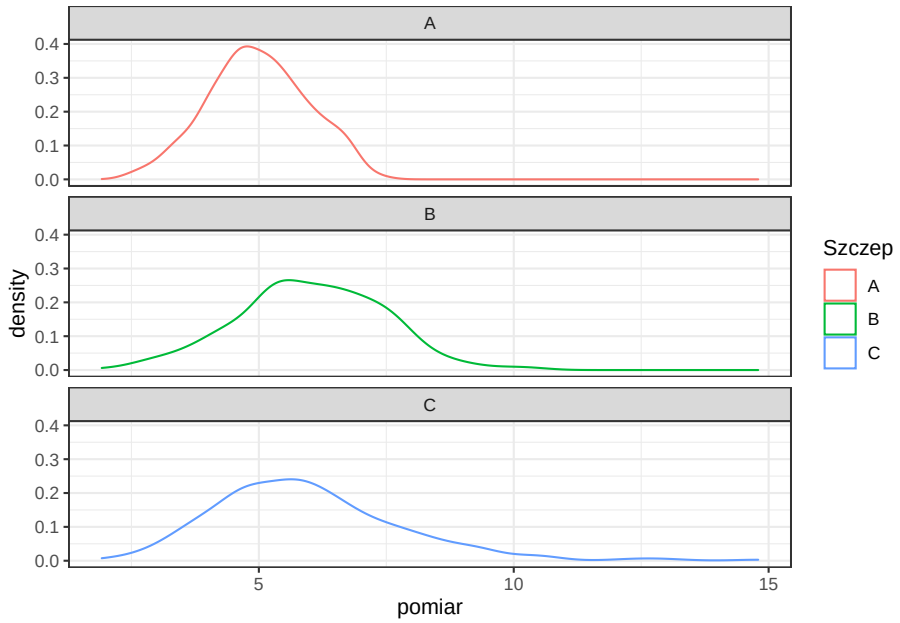
Dzielimy wykres na części pod względem zmiennej np. Szczepu używając `facet_wrap` albo `facet_grid`

`facet_grid` - rozmieści wykresy w tabeli według jednej lub dwóch zmiennych podanych w formule np. zmienna1 ~ zmienna2 `facet_wrap` - stworzy "wstążkę" wykresów, końcową liczbę kolumn/wiersz można ustawić argumentami ncol/nrow. W formule po + można dodawać kolejne zmienne np. ~ zmienna1 + zmienna2

```
p <- ggplot(data = dane1_2, aes(x = pomiar))
p + geom_histogram(aes(y = ..density..), binwidth = 0.5, position = "dodge")+
  facet_wrap(~Szczep, ncol = 1)
```



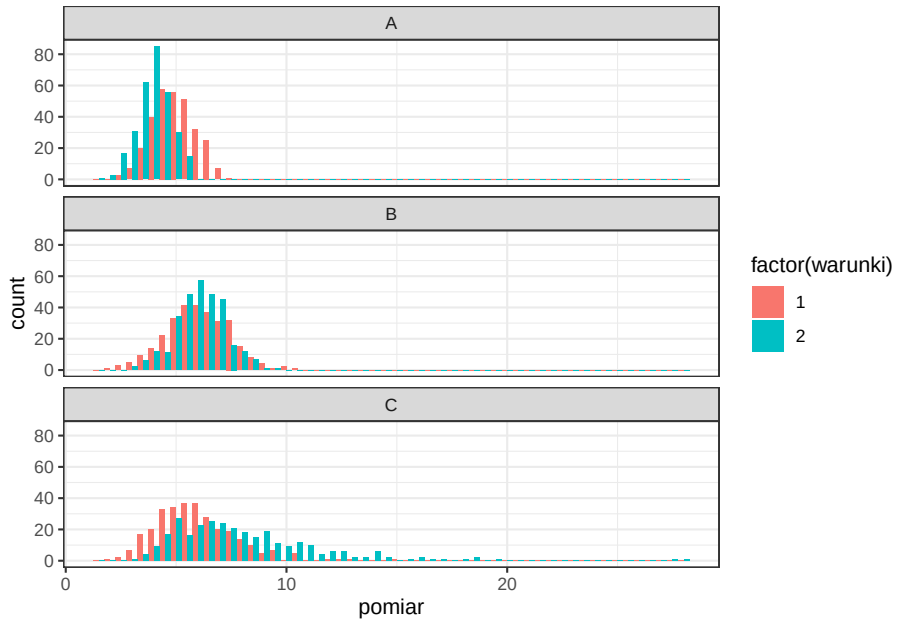

```
p + geom_density(aes(color = Szczep)) + facet_wrap(~Szczep, ncol = 1)
```



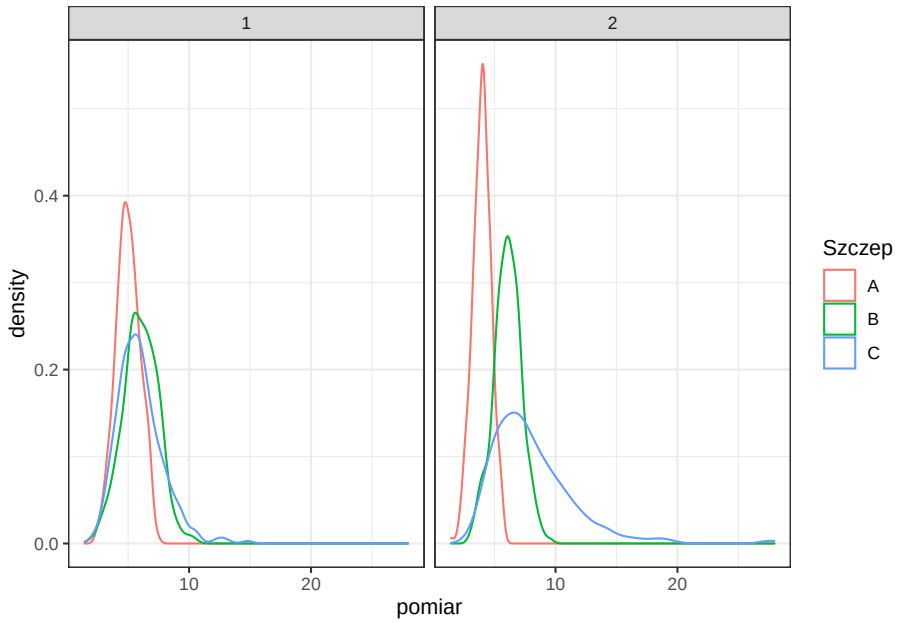
Oba sposoby można ze sobą dowolnie łączyć.

Analizujemy dane pod względem szczepu i warunków.

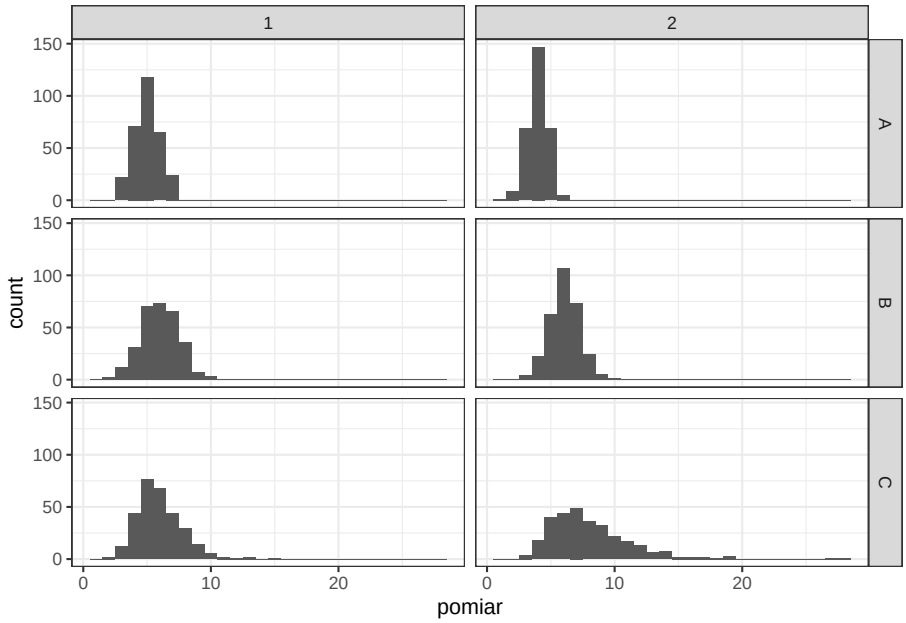
```
p <- ggplot(data = dane1, aes(x = pomiar))
p + geom_histogram(binwidth = 0.5, aes(fill = factor(warunki)), position = "dodge")+
  facet_wrap(~Szczep, ncol = 1)
```



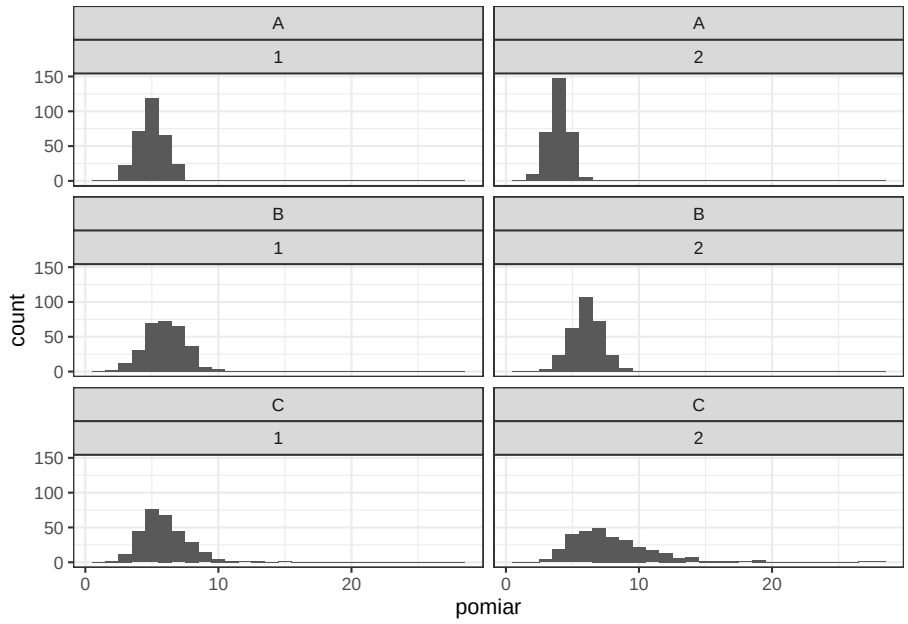

```
p + geom_density(aes(color = Szczep)) + facet_wrap(~ warunki, ncol = 2)
```



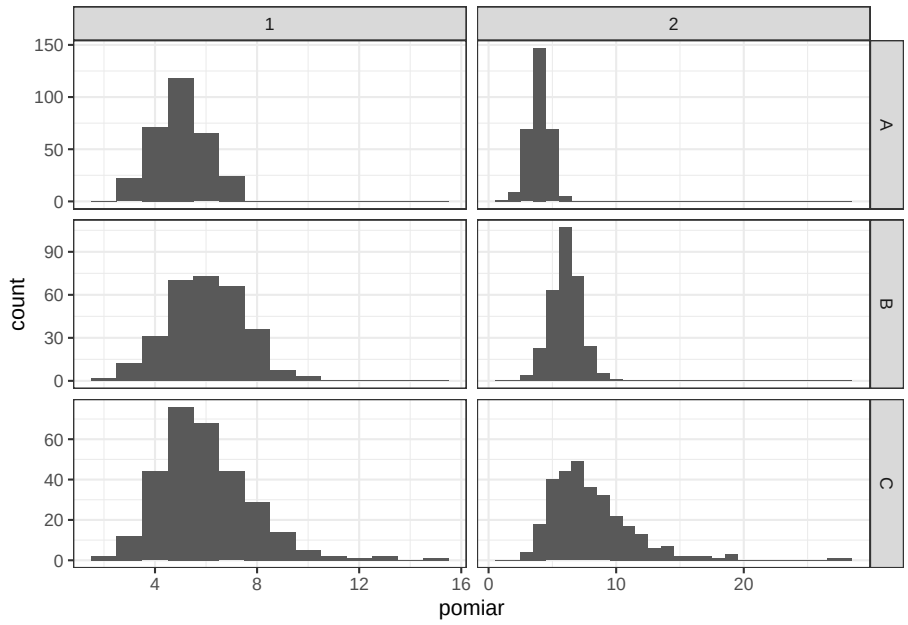
```
# Histogram podzielony pod względem szczepu i warunków - facet_grid
p + geom_histogram(binwidth = 1) + facet_grid(Szczep ~ warunki)
```



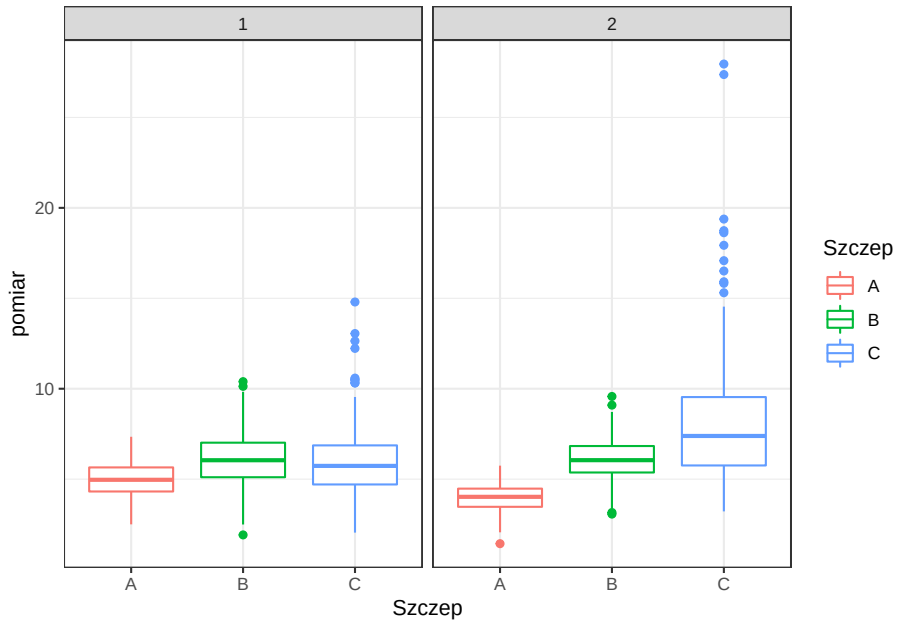

```
# Podobny efekt do facet_frid można osiągnąć stosując facet_wrap z więcej niż jednym w
# Kolejne warunki dodaje się znakiem +
p + geom_histogram(binwidth = 1) + facet_wrap(~Szczep + warunki, ncol = 2)
```



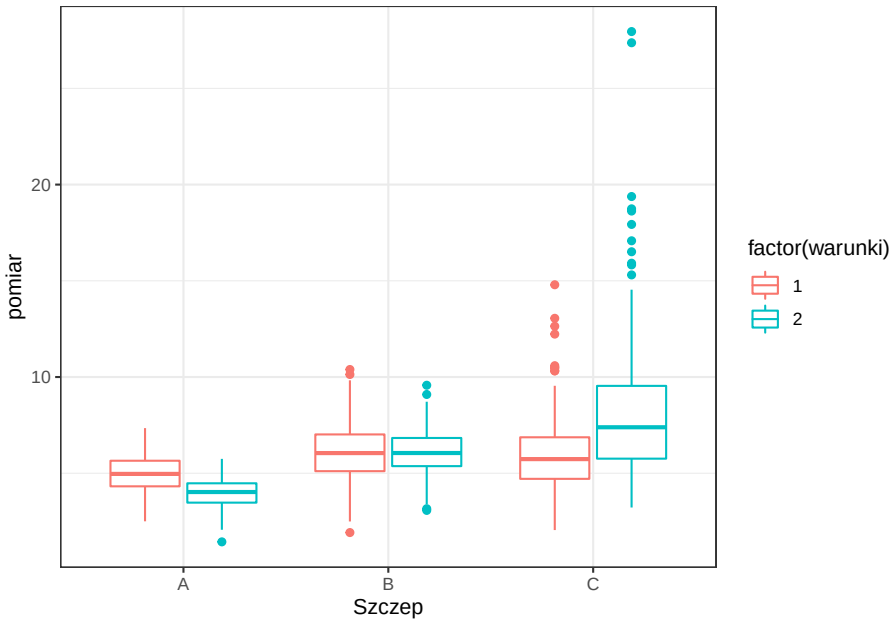

```
# Osie nie muszą być jednakowe dla wszystkich części
p + geom_histogram(binwidth = 1) + facet_grid(Szczep ~ warunki, scales = "free")
```



```
# Dzielić na części dzięki facet można każdy typ wykresu
p <- ggplot(data = dane1, aes(x = Szczep, y = pomiar))
p + geom_boxplot(aes(color = Szczep)) + facet_wrap(~ warunki)
```




```
# Boxploty można również rozmiescić według jednego czynnika, a pokolorować według drug
p + geom_boxplot(aes(color = factor(warunki)))
```



4.6.4 Zmiana skali, osi, obracanie wykresu

ggplot2 domyślnie dobiera takie parametry osi, żeby zmieściły się wszystkie dane, ale można je zmieniać używając `scale_x_continuous` albo `scale_y_continuous`.

Jeżeli chcemy tylko zmienić limity osi można to zrobić funkcją `xlim` i `ylim`.

Trzeba pamiętać, że po zmianie limitów osi najpierw zostaną usunięte wartości, które się nie mieszczą, a potem policzone statystyki, więc takie wykresy jak `geom_boxplot`, `stat_smooth`, `stat_summary` i inne mogą ulec zmianie. Jeżeli chcemy tego uniknąć należy zamiast osi zmieniać układ współrzędnych - `coord_cartesian(xlim, ylim)`.

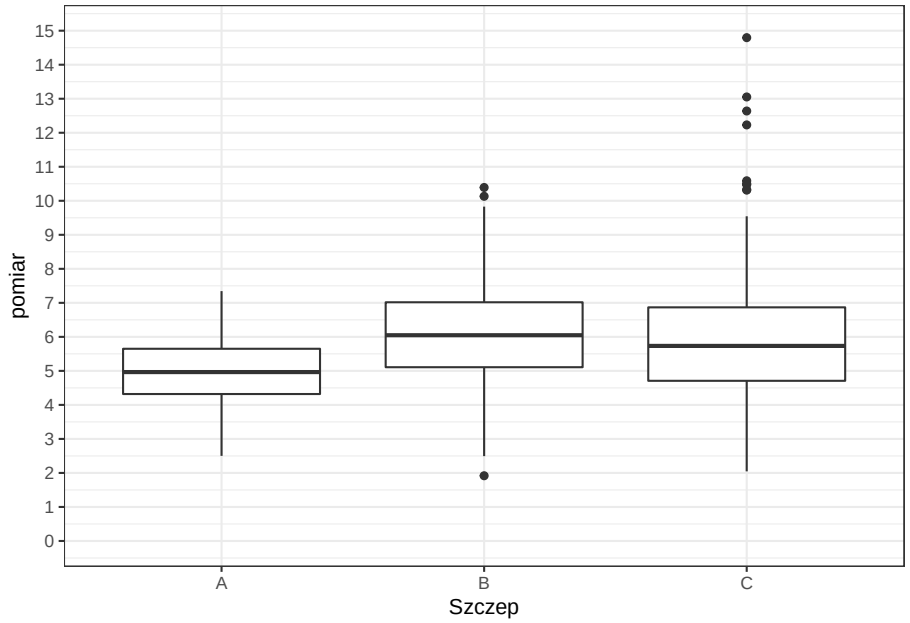
Przy ich pomocy możemy zmienić np. parametry: * limits - miejsce startu i końca osi np `limits = c(1,10)` * name - nazwa osi * breaks - miejsca "tick marks" * labels - nazwy "tick marks"

Podstawowe transformacje osi to `scale_y_log10`, `scale_y_reverse`, `scale_y_sqrt`.

Oś % - należy wpisać `labels=percent` w scale oraz załadować pakiet scales.

Jeżeli chcemy obrócić wykres o 90 stopni możemy użyć funkcji `coord_flip`.

```
# Zmiana osi na przykładzie boxplot
p <- ggplot(data = dane1_2, aes(x = Szczep, y = pomiar))
p <- p + geom_boxplot()

# Ustawienie startu i końca osi oraz miejsc podziału
p + scale_y_continuous(limits = c(0,15), breaks = 0:15)
```




```
# Miejsca podziału nie muszą być w równych odstępach i mogą być dowolnie nazwane
p + scale_y_continuous(limits = c(0,15), breaks = c(1,4,7,15),
                       labels = c("mało", "lepiej", "ok", "za dużo"),
                       name = "Coś")
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Coś' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Coś' in 'mbcsToSbcs': dot substituted for <9b>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Coś' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Coś' in 'mbcsToSbcs': dot substituted for <9b>
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Coś' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Coś' in 'mbcsToSbcs': dot substituted for <9b>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Coś' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Coś' in 'mbcsToSbcs': dot substituted for <9b>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Coś' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Coś' in 'mbcsToSbcs': dot substituted for <9b>
```

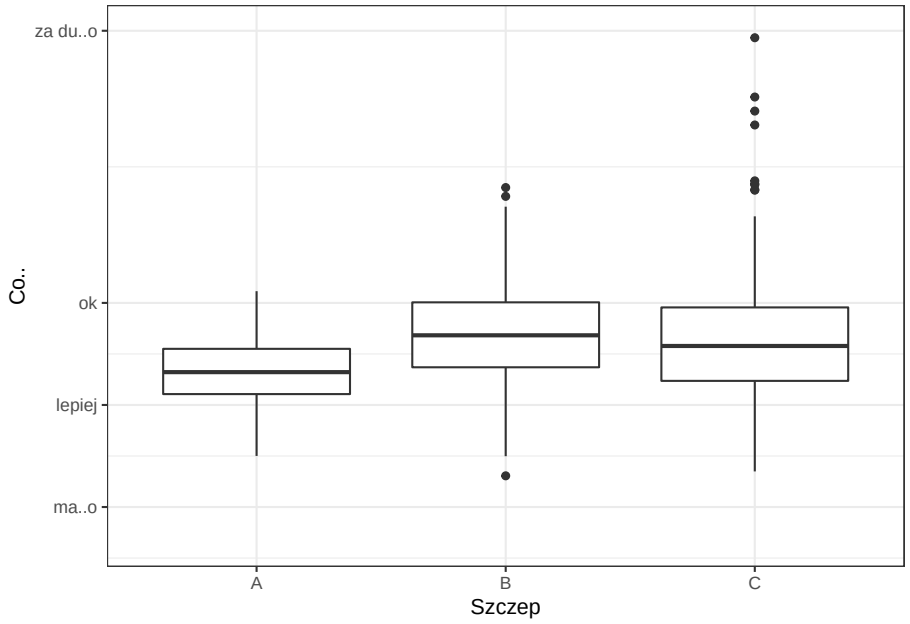
```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <82>
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <bc>
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <82>
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <82>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <bc>
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Coś' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Coś' in 'mbcsToSbcs': dot substituted for <9b>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Coś' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Coś' in 'mbcsToSbcs': dot substituted for <9b>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Coś' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Coś' in 'mbcsToSbcs': dot substituted for <9b>
```




```
# Wykres obrócony
p + scale_y_continuous(limits = c(0,15), breaks = c(1,4,7,15),
                       labels = c("mało", "lepiej", "ok", "za dużo"),
                       name="Coś") + coord_flip()
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Coś' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Coś' in 'mbcsToSbcs': dot substituted for <9b>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Coś' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Coś' in 'mbcsToSbcs': dot substituted for <9b>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <82>
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Coś' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Coś' in 'mbcsToSbcs': dot substituted for <9b>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Coś' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Coś' in 'mbcsToSbcs': dot substituted for <9b>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Coś' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Coś' in 'mbcsToSbcs': dot substituted for <9b>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <82>
```

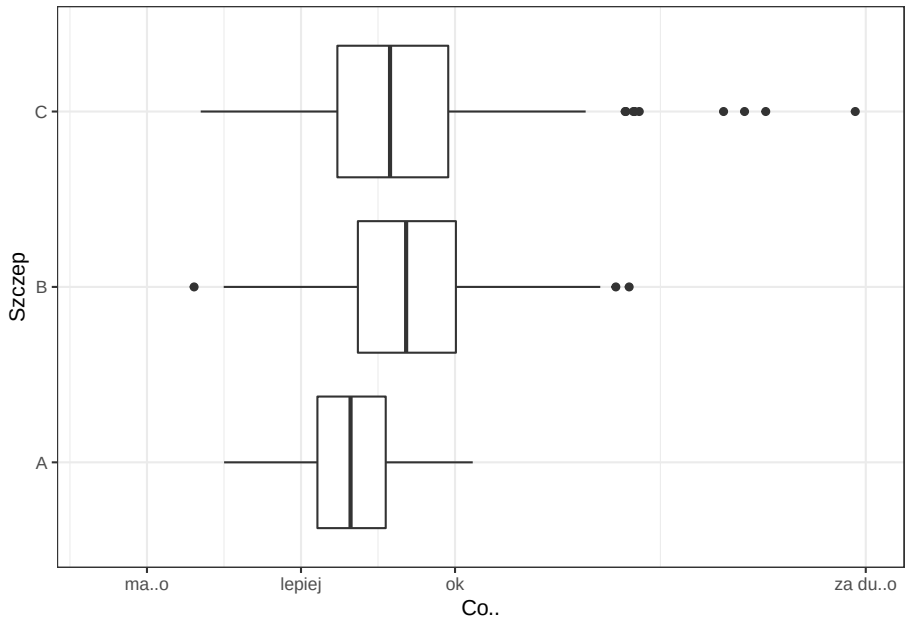
```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <bc>
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <82>
```

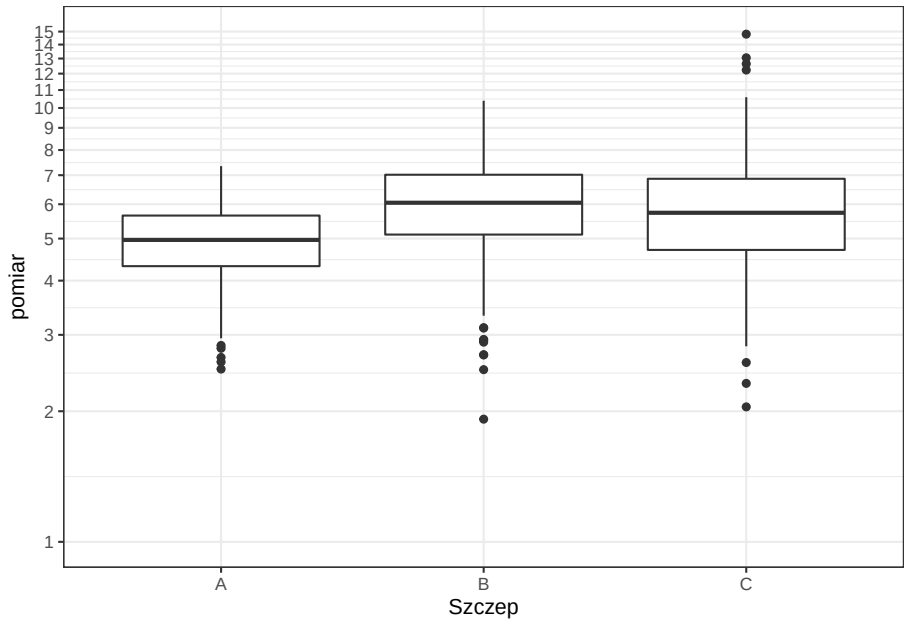
```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'mało' in 'mbcsToSbcs': dot substituted for <82>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'za dużo' in 'mbcsToSbcs': dot substituted for <bc>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Coś' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Coś' in 'mbcsToSbcs': dot substituted for <9b>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Coś' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Coś' in 'mbcsToSbcs': dot substituted for <9b>
```

```
## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Coś' in 'mbcsToSbcs': dot substituted for <c5>
```

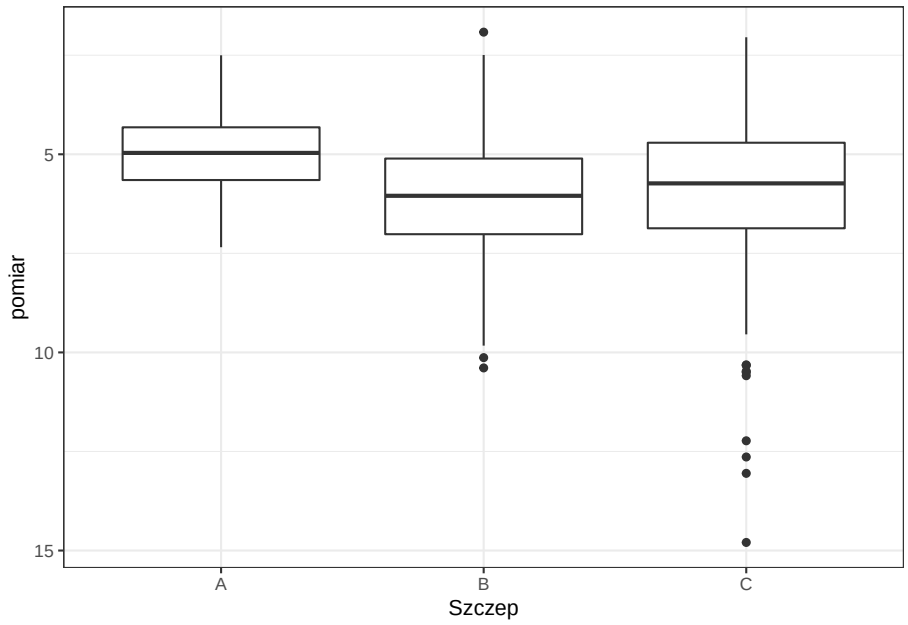
```
## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Coś' in 'mbcsToSbcs': dot substituted for <9b>
```



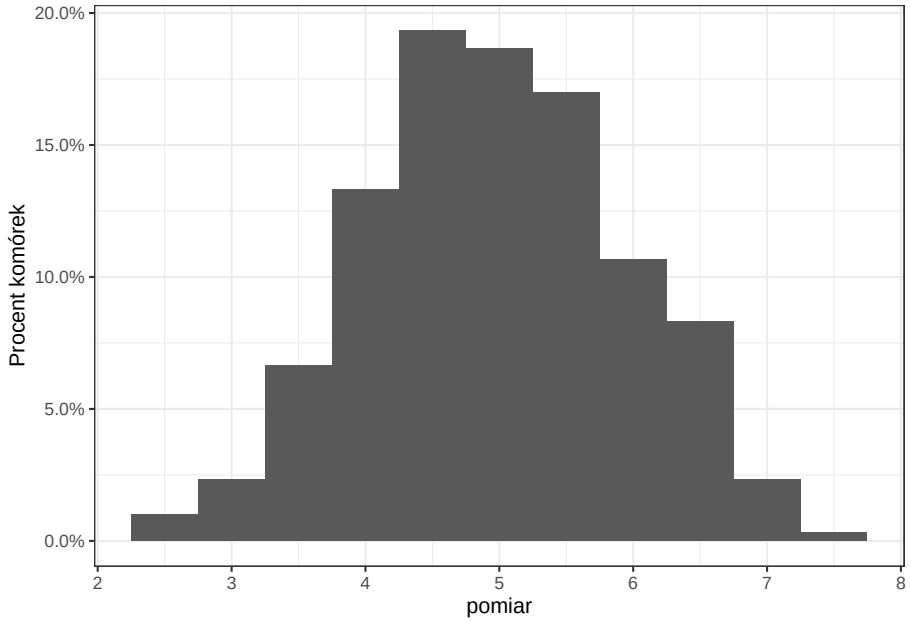

```
# Skala logarytmiczna
p + scale_y_log10(limits = c(1,15), breaks = 1:15)
```




```
# Wykres "do góry nogami"
p + scale_y_reverse()
```



```
library(scales)
# Oś procentowa
p <- ggplot(data = dane1_1, aes(x = pomiar))
p + geom_histogram(binwidth = 0.5, aes(y = ((..count..)/sum(..count..))))+
  scale_y_continuous(labels = percent, name = "Procent komórek")
```

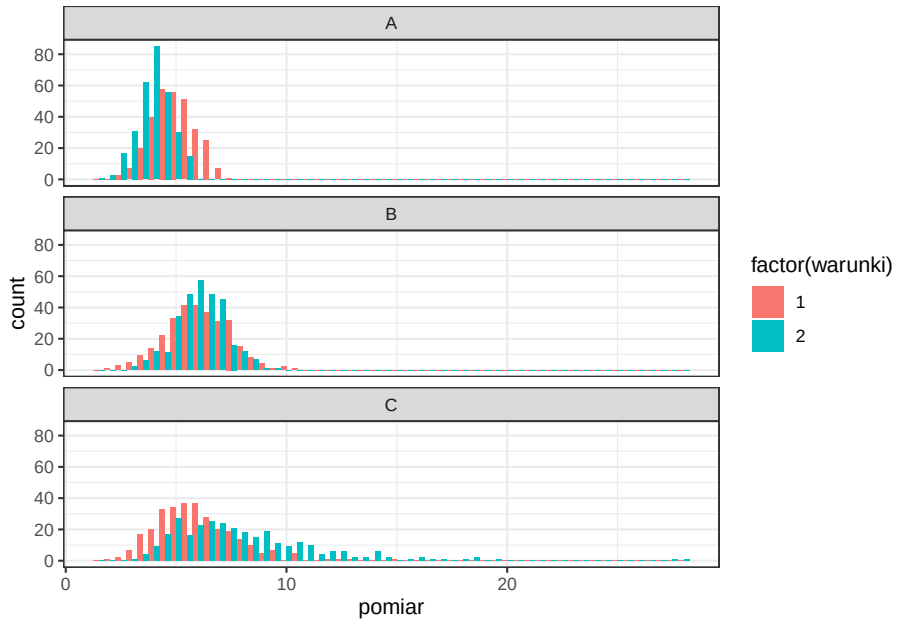


4.7 Motyw (theme)

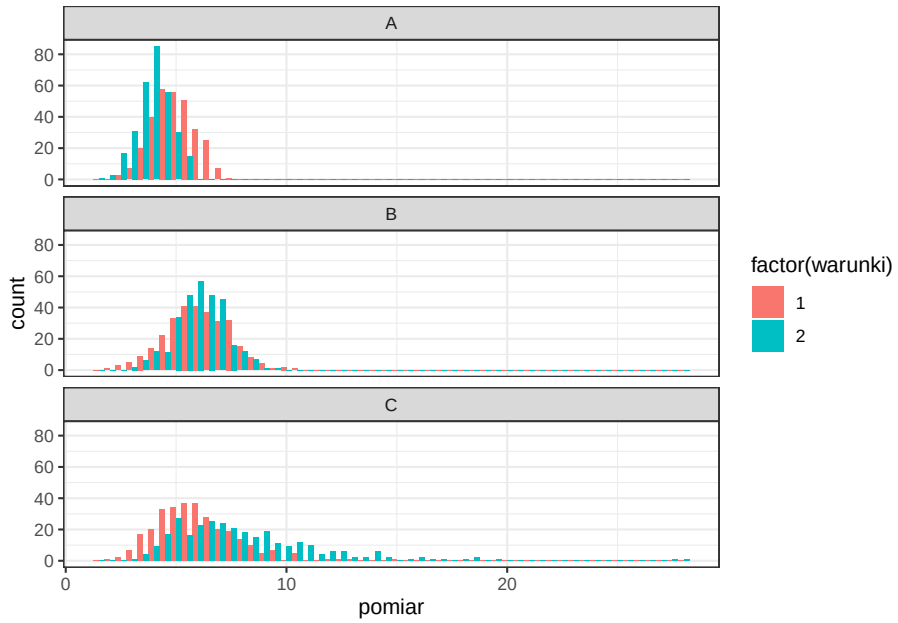
W pakiecie ggplot2 jest dostępnych kilka różnych motywów. Domyślnie ustawiony jest `theme_grey`, inne dostępne to `theme_bw`, `theme_minimal`, `theme_classic`, `theme_linedraw`, `theme_light`. W pakiecie ggthemes znajdują się dodatkowe wersje motywów, nawet (o zgrozo) `theme_excel` ;) Inny pakiet zawierający gotowe motywy to ggthemr - można go pobrać z GitHub.

Tytuł do wykresu możemy dodać korzystając z funkcji `ggtitle`. Nazwy osi też można szybko zmienić przy pomocy `xlab` i `ylab`.

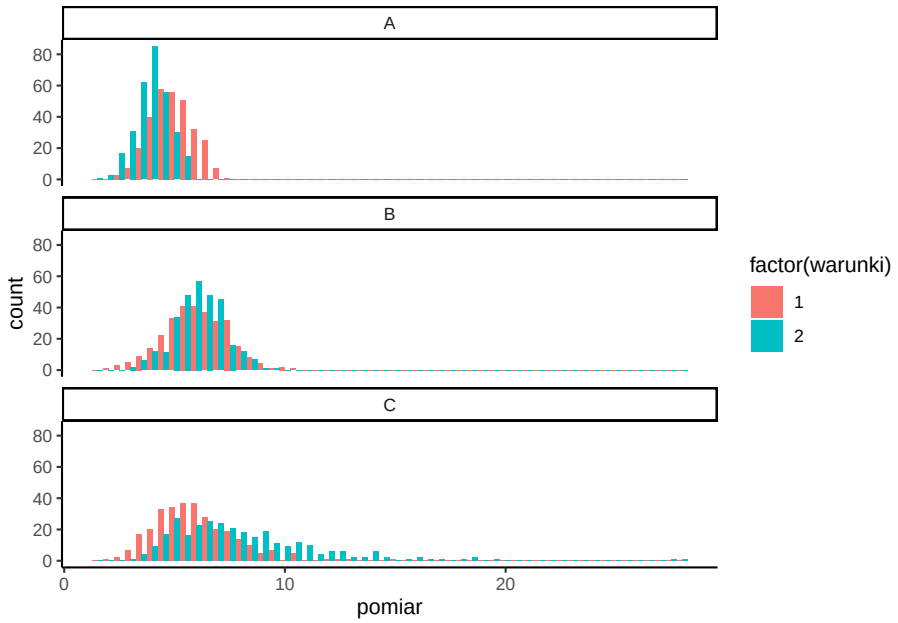
```
p <- ggplot(data = dane1, aes(x = pomiar))
p <- p + geom_histogram(binwidth = 0.5, aes(fill = factor(warunki)), position = "dodge"
  facet_wrap(~Szczep, ncol = 1)
p
```


A
B
C
count
pomiar
0
10
20
0
20
40
60
80
factor(warunki)
1
2

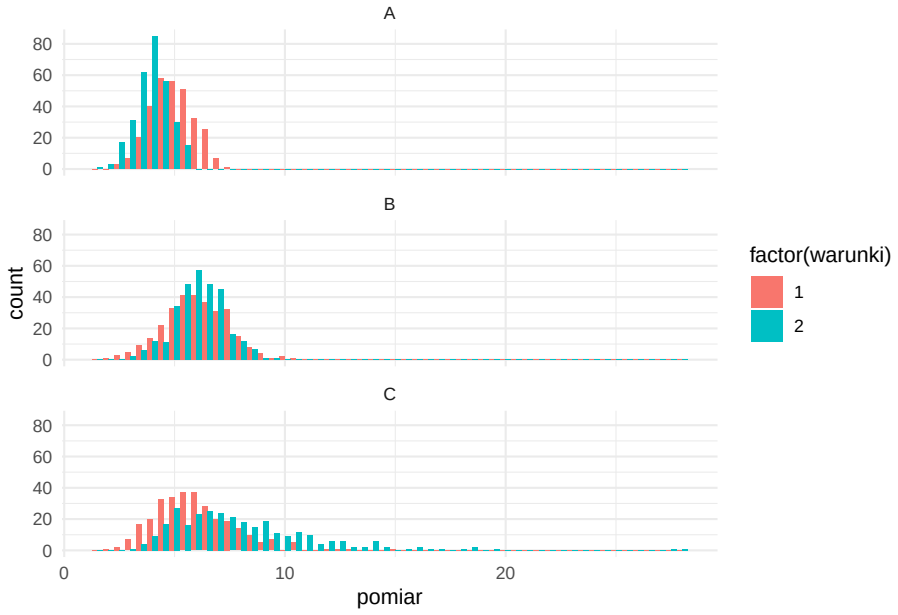
```
p + theme_bw()
```


A
B
C
count
pomiar
0
10
20
0
20
40
60
80
factor(warunki)
1
2

```
p + theme_classic()
```




```
p + theme_minimal()
```



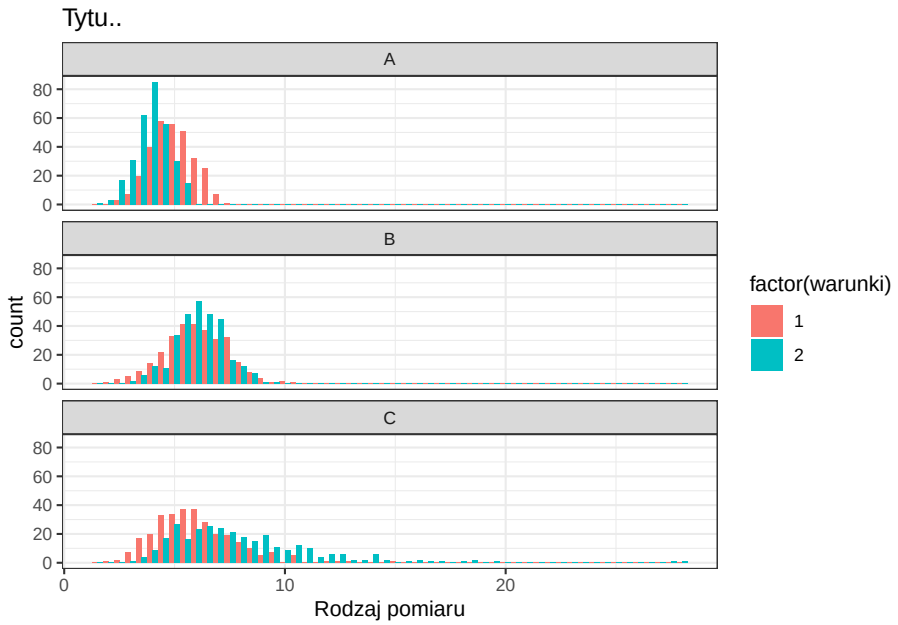
```
p + ggtitle("Tytuł") + xlab("Rodzaj pomiaru")
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Tytuł' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Tytuł' in 'mbcsToSbcs': dot substituted for <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Tytuł' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Tytuł' in 'mbcsToSbcs': dot substituted for <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Tytuł' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Tytuł' in 'mbcsToSbcs': dot substituted for <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Tytuł' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Tytuł' in 'mbcsToSbcs': dot substituted for <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Tytuł' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Tytuł' in 'mbcsToSbcs': dot substituted for <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Tytuł' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Tytuł' in 'mbcsToSbcs': dot substituted for <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Tytuł' in 'mbcsToSbcs': dot substituted for <c5>
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Tytuł' in 'mbcsToSbcs': dot substituted for <82>
```

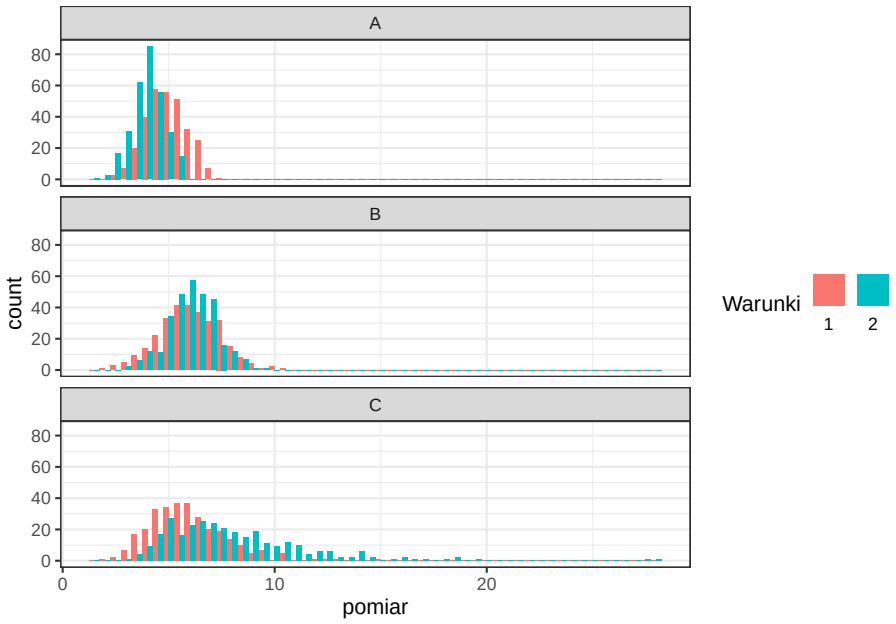
```
## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Tytuł' in 'mbcsToSbcs': dot substituted for <c5>
```

```
## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Tytuł' in 'mbcsToSbcs': dot substituted for <82>
```



Legendę wykresu można modyfikować przy pomocy `guide_legend` wewnątrz funkcji `scale_fill_discrete` , `scale_colour_discrete` itp. Dostępne parametry to m.in. title, title.position, label.position, direction, nrow i ncol legendy. Modyfikacje są możliwe też bezpośrednio w funkcji `theme` albo samodzielnie ustawiając przez `scale_color_manual`.

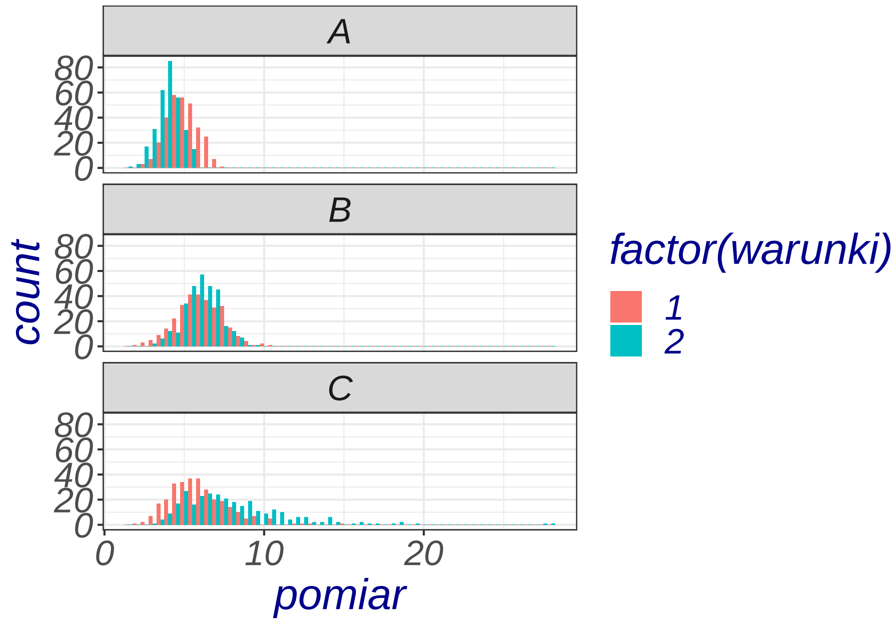
```
p + scale_fill_discrete(guide = guide_legend(title = "Warunki",
                                             title.position = "left",
                                             label.position = "bottom", ncol = 2))
```



Można również modyfikować osobno każdy element wykresu np. czcionkę, kolor, linie, tło itd. przy pomocy funkcji `theme`, dużo przykładów znajduję się na stronie.

Można modyfikować jednocześnie wszystkie elementy danego rodzaju np. tekst przy pomocy `text = element_text()` albo pojedyncze części wykresu np. tytuł - `plot.title=element_text()`.

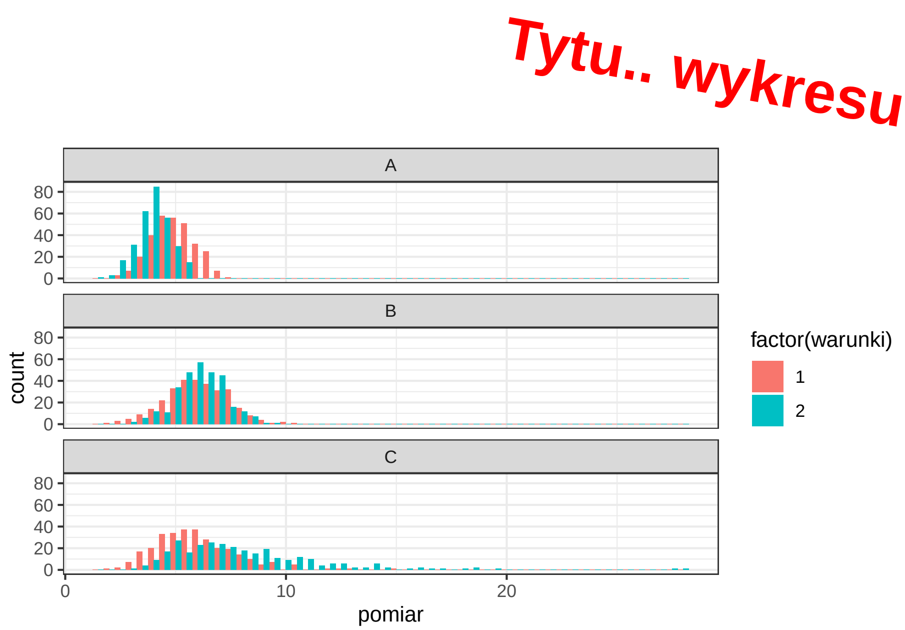
```
p + theme(text = element_text(size = 22, face = "italic", color = "darkblue"))
```



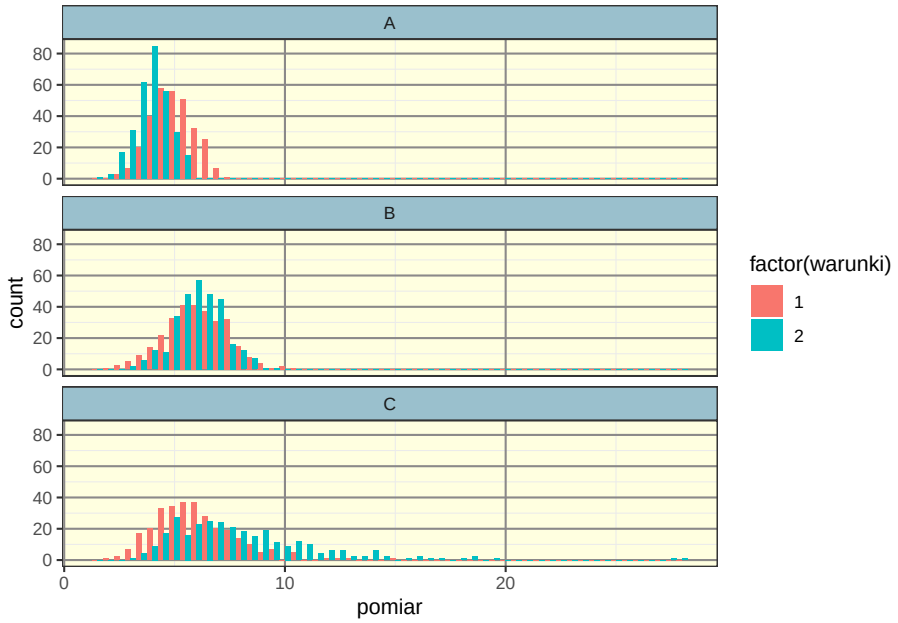

```
p + ggtitle("Tytuł wykresu")+theme(plot.title = element_text(color = "red",
                                                              face = "bold", size = 30, an
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Tytuł wykresu' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Tytuł wykresu' in 'mbcsToSbcs': dot substituted for <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Tytuł wykresu' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Tytuł wykresu' in 'mbcsToSbcs': dot substituted for <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Tytuł wykresu' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Tytuł wykresu' in 'mbcsToSbcs': dot substituted for <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Tytuł wykresu' in 'mbcsToSbcs': dot substituted for <c5>
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Tytuł wykresu' in 'mbcsToSbcs': dot substituted for <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Tytuł wykresu' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Tytuł wykresu' in 'mbcsToSbcs': dot substituted for <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Tytuł wykresu' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Tytuł wykresu' in 'mbcsToSbcs': dot substituted for <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Tytuł wykresu' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Tytuł wykresu' in 'mbcsToSbcs': dot substituted for <82>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Tytuł wykresu' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Tytuł wykresu' in 'mbcsToSbcs': dot substituted for <82>
```

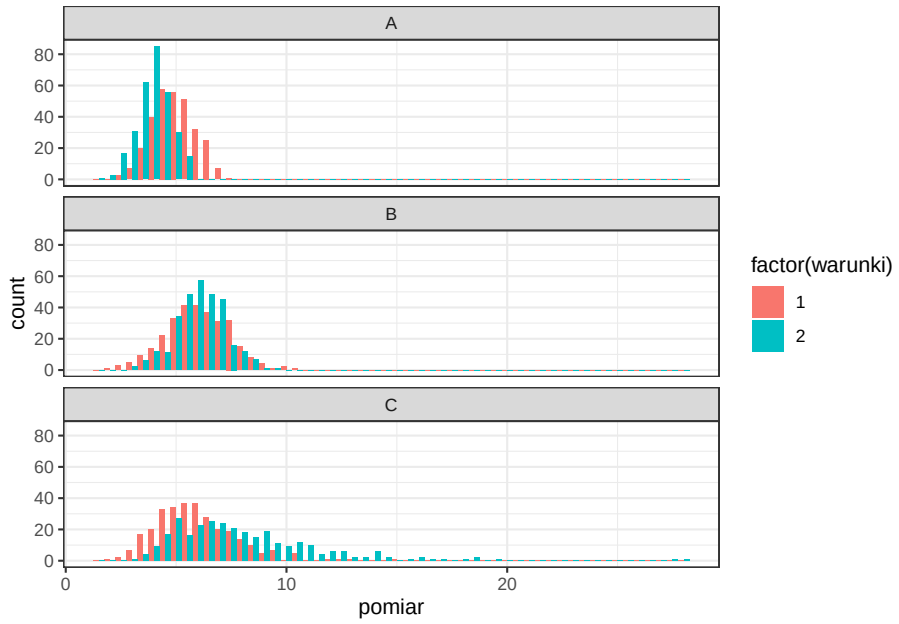


```
p + theme(panel.background = element_rect(fill = "lightyellow"),
          panel.grid.major = element_line(color = "snow4"),
          strip.background = element_rect(fill = "lightblue3"))
```

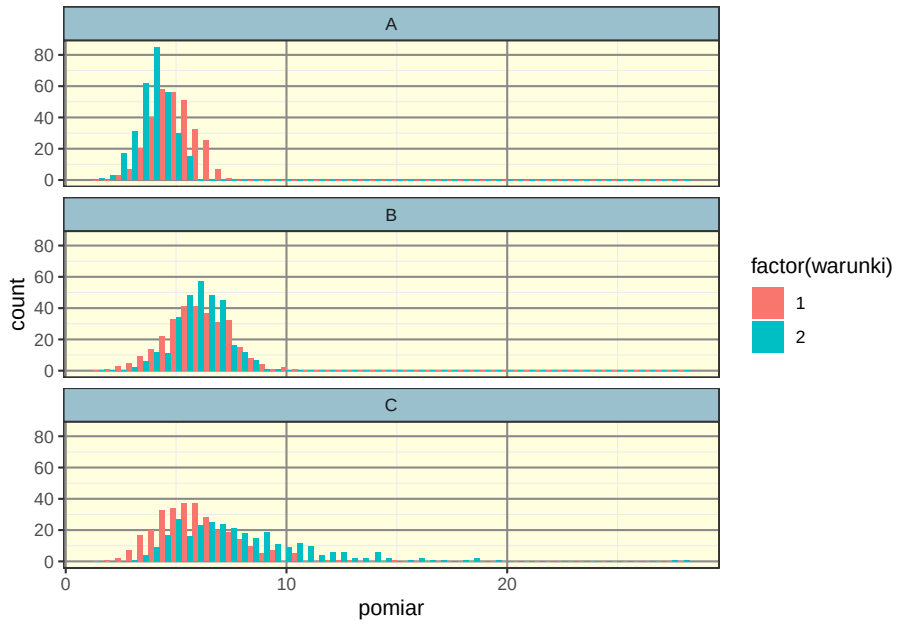


Jeżeli chcemy przygotować kilka pasujących do siebie wykresów możemy zapisać swój motyw i potem dodawać go do kolejnych wykresów.

```
motyw <- theme(panel.background = element_rect(fill = "lightyellow"),
               panel.grid.major = element_line(color = "snow4"),
               strip.background = element_rect(fill = "lightblue3"))

p
```


A
B
C
count
pomiar
factor(warunki)
1
2

```
p + motyw
```


A
B
C
count
pomiar
factor(warunki)
1
2

4.8 Różne

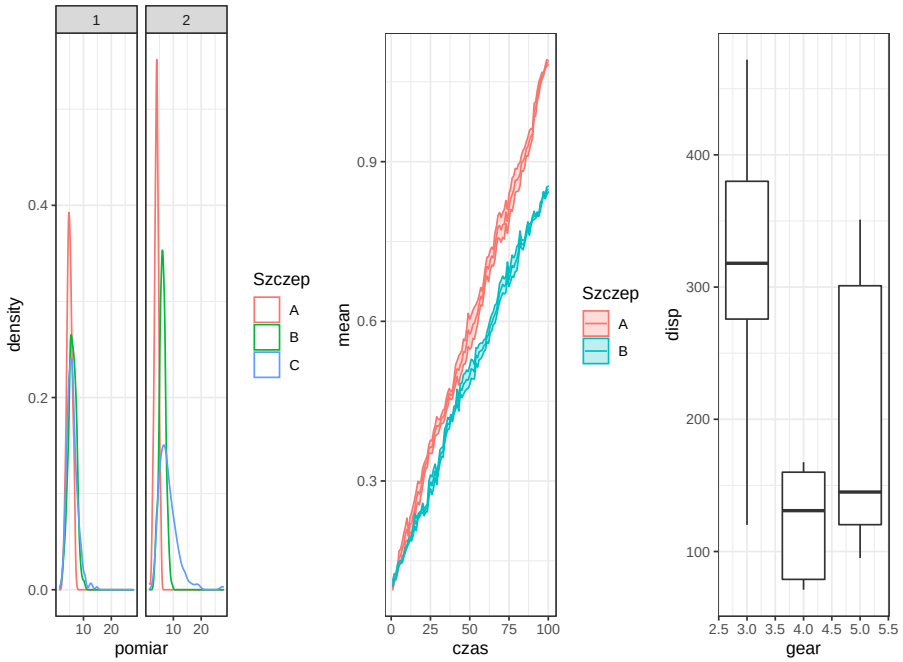
4.8.1 Łączenie wykresów

Pakiet ggplot2 jest oparty o system wyświetlania kontrolowany przez pakiet grid (inny niż grafika z podstawowego R). Korzystając z funkcji `viewport` można z dużą dokładnością rozmieścić kilka wykresów różnych rozmiarów obok siebie, jeden na drugim itp.

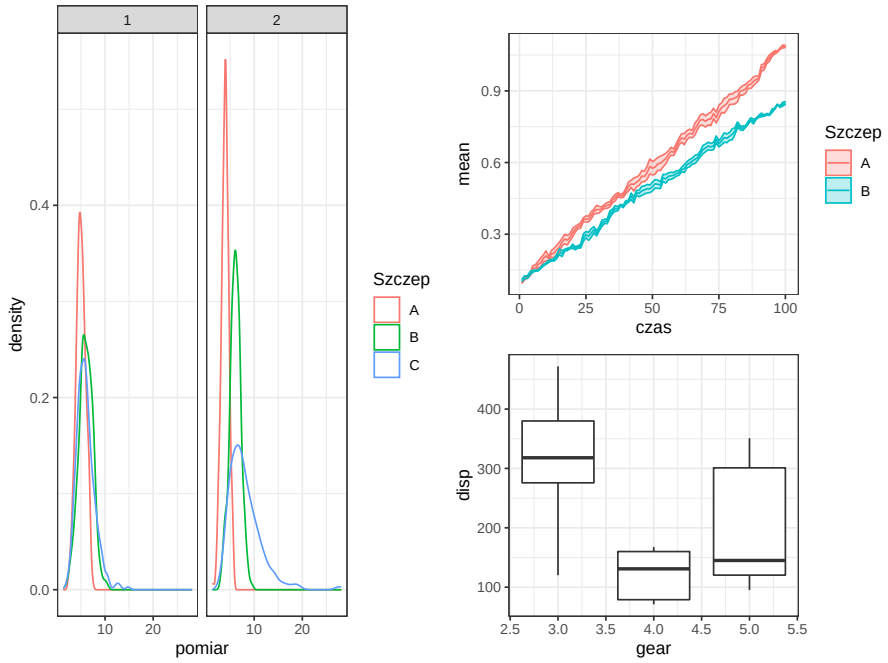
W funkcji `viewport` ustawiamy parametry `width` i `height` oznaczające wymiary wykresu. Wykres zajmujący całą powierzchnię ma wymiary 1x1 oraz `x` i `y` oznaczające współrzędne środka wykresu np. x=0.5, y=0.5 da wykres umiejscowiony na samym środku.

Dużo łatwiejszym sposobem jest wykorzystanie pakietu patchwork. Pozwala on na ułożenie wykresów na jednej stronie tlyko przy użyciu +, | i /. Można też dokładnie ustalać rozmieszczenie wykresów, więcej na stronie autora pakietu

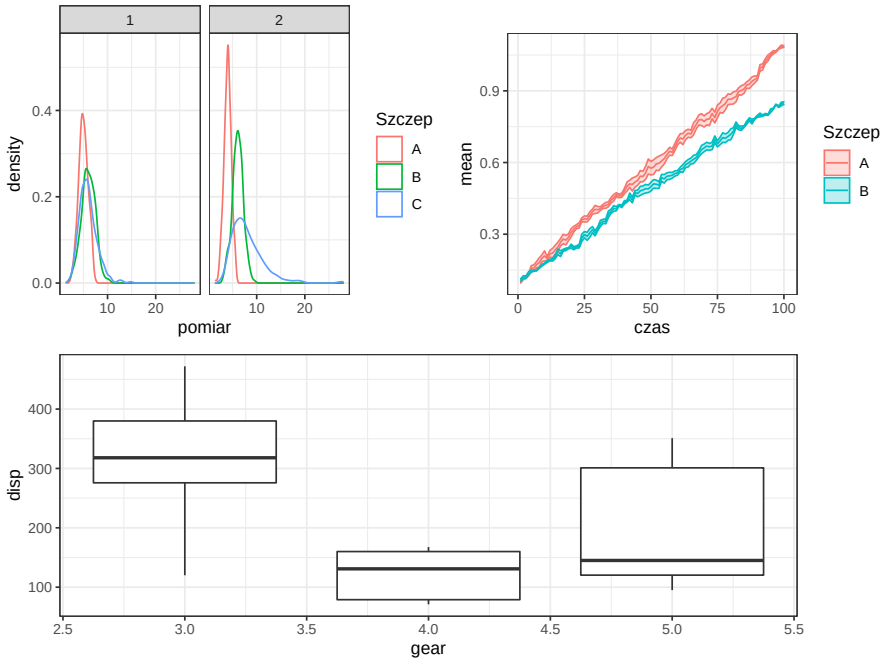
```
# Przypisujemy wykresy do zmiennych

p1 <- ggplot(data=dane1, aes(x=pomiar))
p1 <- p1 + geom_density(aes(color=Szczep))+facet_wrap(~warunki, ncol=2)

p2 <- ggplot(data=summ, aes(x=czas, y=mean, color=Szczep, fill=Szczep))
p2 <- p2 + geom_line()+geom_ribbon(aes(ymin=lower, ymax=upper),alpha=0.25)

p3 <- ggplot(mtcars) + geom_boxplot(aes(gear, disp, group = gear))

# z wykorzytsaniem patchwork

library(patchwork)

p1 + p2 + p3
```


1
2
density
0.4
0.2
0.0
10
20
pomiar
Szczep
A
B
C
mean
0.9
0.6
0.3
0
25
50
75
100
czas
Szczep
A
B
disp
400
300
200
100
2.5
3.0
3.5
4.0
4.5
5.0
5.5
gear

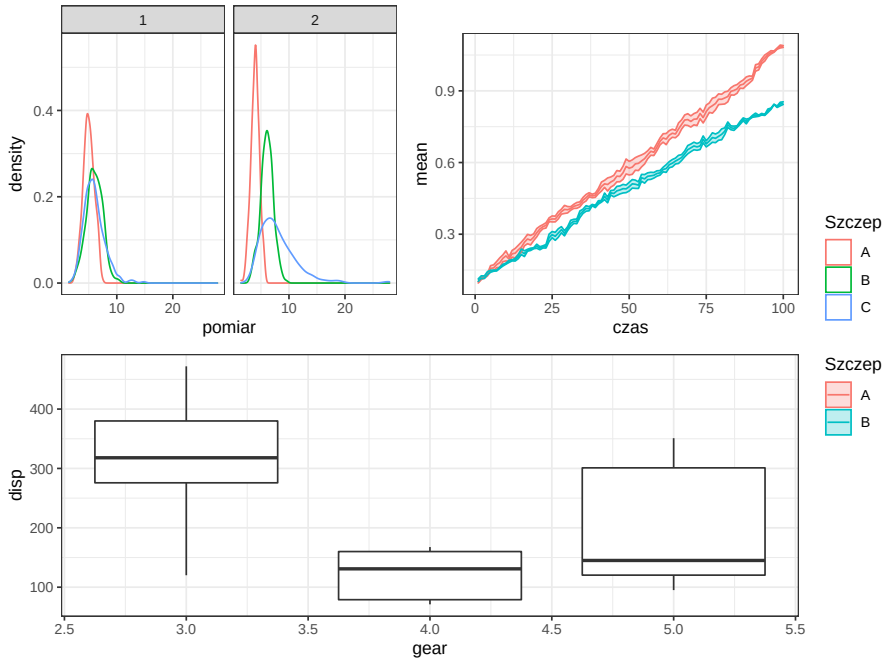
```
p1 | p2 / p3
```


1
2
density
0.0
0.2
0.4
10
20
pomiar
Szczep
A
B
C
mean
0.3
0.6
0.9
0
25
50
75
100
czas
Szczep
A
B
disp
100
200
300
400
2.5
3.0
3.5
4.0
4.5
5.0
5.5
gear

```
(p1 | p2 )/ p3
```

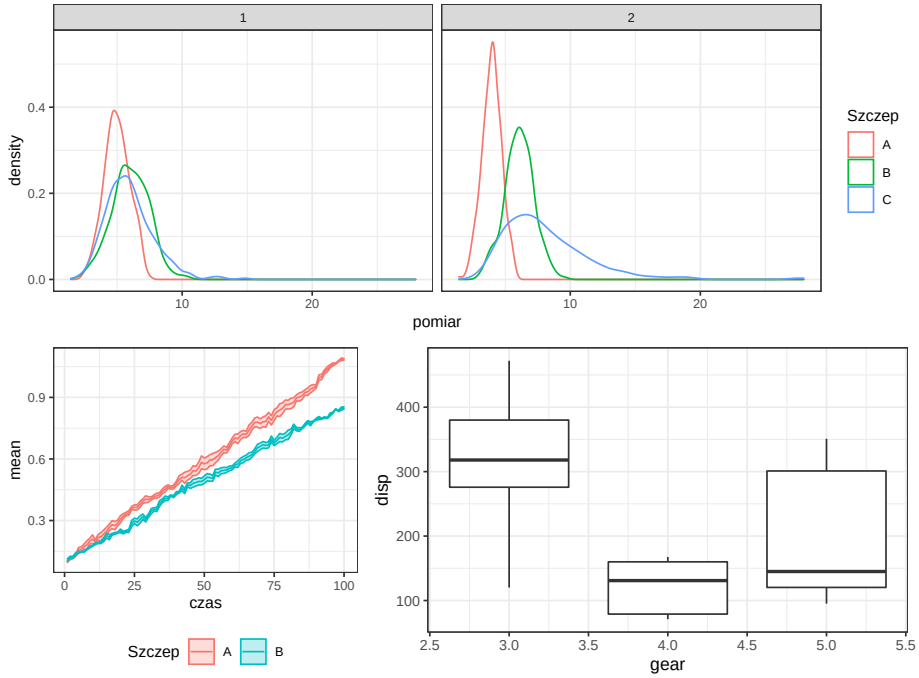



```
(p1 | p2 )/ p3 + plot_layout(guides = 'collect')
```




```
layout <- "
##BBBB
AACCCC
AACCCC
"

p1 + p2 + p3 +
  plot_layout(design = layout)

# Z wykorzystaniem pakietu grid

library(grid)

p1 <- p1 + theme(text=element_text(size=10))
p2 <- p2 + theme(text=element_text(size=10), legend.position="bottom")

# Ustawiamy parametry viewportów
vp1 <- viewport(width=1, height=0.5, x=0.5, y=0.75)
vp2 <- viewport(width=0.4, height=0.5, x=0.2, y=0.25)
vp3 <- viewport(width=0.6, height=0.5, x=0.7, y=0.25)

# Wyświetlamy wykresy w ospowiednich viewportach
print(p1, vp=vp1)
```

```
print(p2, vp=vp2)
print(p3, vp=vp3)
```



4.8.2 Ten sam wykres różne dane

Istnieje kilka sposobów na przygotowanie kilku takich samych wykresów, różniących się jedynie danymi.

Można oczywiście ręcznie podmienić wartość parametru data na inny albo skorzystać z wbudowanego w pakiet ggplot2 operatora - %+%.

Alternatywą jest też napisanie własnej funkcji przygotowującej konkretny wykres. Zaletą tego rozwiązania jest możliwość wpisania do funkcji odpowiednich argumentów dostosowujących wykres do konkretnej sytuacji.

```
# Przykładowe zestawy danych
a <- data.frame(x = rnorm(1000))
b <- data.frame(x = rlnorm(1000))
c <- data.frame(x = runif(1000, 0, 5))

# Przygotujemy wykres składajacy się z kilku elementów dla danych a
```

```
p <- ggplot(data = a, aes(x = x))
p <- p + geom_histogram(binwidth = 0.25, fill = "blue4", aes(y = (..count../sum(..coun
  scale_y_continuous(labels = percent, name = "Procent")+
  xlab("Wartość")+
  ggtitle("Przykładowy rozkład")+
  theme(panel.background=element_rect(fill = "white"), text = element_text(size = 14),
        axis.text = element_text(color = "red4"))
p
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c5>
```

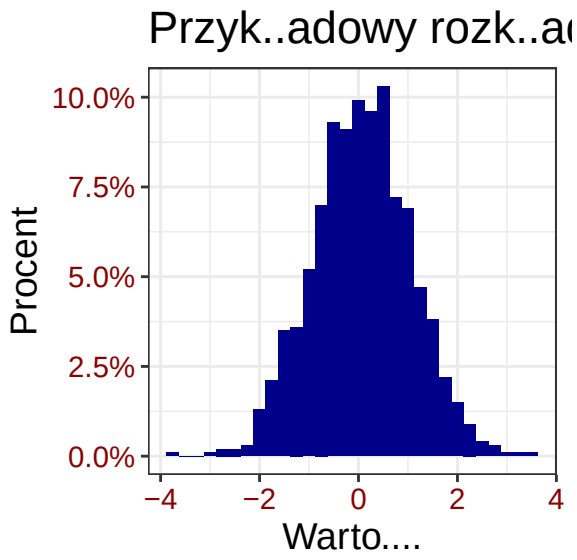
```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <9b>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c4>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <87>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <9b>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c4>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <87>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <9b>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c4>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <87>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <9b>
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c4>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <87>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <9b>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c4>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <87>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <9b>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c4>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <87>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <9b>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c4>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <87>
```

```
## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <9b>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c4>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <87>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>
```

```
## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>
```




```
# taki sam wykres dla danych b

p %+% b
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <9b>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c4>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <87>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <9b>
```

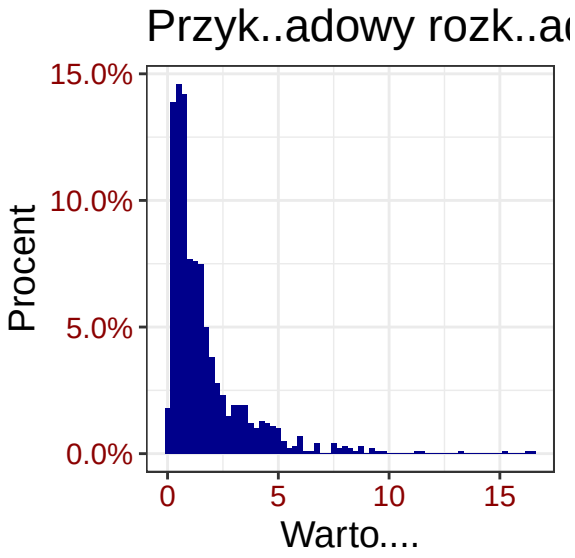
```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c4>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <87>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <9b>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c4>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <87>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <9b>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c4>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <87>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <9b>
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c4>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <87>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <9b>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c4>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <87>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <9b>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c4>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <87>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <9b>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c4>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <87>
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>
```

```
## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>
```




```
# i c ;)

p %+% c
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <9b>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c4>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <87>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <9b>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c4>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <87>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c5>
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <9b>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c4>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <87>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <9b>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c4>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <87>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <9b>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c4>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <87>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <9b>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c4>
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <87>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <9b>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c4>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <87>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c5>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <9b>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <c4>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Wartość' in 'mbcsToSbcs': dot substituted for <87>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>
```

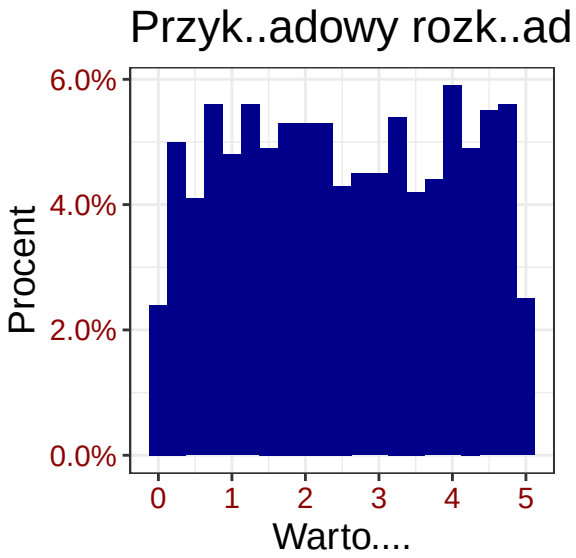
```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>
```

```
## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>
```

```
## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>
```

```
## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <c5>
```

```
## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Przykładowy rozkład' in 'mbcsToSbcs': dot substituted for
## <82>
```



4.9 Rozszeżenia ggplot2

W ostatnich latatach powstało bardzo wiele pakietów rozbudowujących możliwości ggplot2. Część z nich została już wspomniana wcześniej np. ggbeeswarm lub patchwork. Tutaj znajdą się inne, które również mogą okazać się przydatne. Większość dobrze udokumentowanych pakietów można znaleźć na stronie ggplot2 extensions - gallery.

4.9.1 Pakiet GGally

Pakiet GGally stanowi rozszeżenie ggplot2, zawiera kilka szablonóe i pozwala na stworzenie wykresów niedostępnych w wersji podstawowej np. macierz korelacji, wykres pokazujący sieć albo macierz wykresów dla ramki danych.

4.9.1.1 Macierz wykresów - ggpairs

Funkcja `ggpairs` pozwala na szybką analizę danych. Jej argumentem jest ramka danych i dla każdej pary zmiennych zostanie narysowany wykres pozkazujący zalezność pomiędzy nimi. Wykresy są inne w zależności od rodzaju zmiennych - liczbowe lub kategoryczne. Dla pary zmiennych liczbowych zostanie narysowany wykres rozrzutu i obliczony współczynnik korelacji. Dla pary mieszanej (liczbowo-kategoryczna) narysuje wykres pudełkowy i histogram, dla dwóch zmiennych kategorycznych wykresy słupkowe. Rodzaje rysowanych wykresów można zmieniać, można też do macierzy dodać własny wykres.

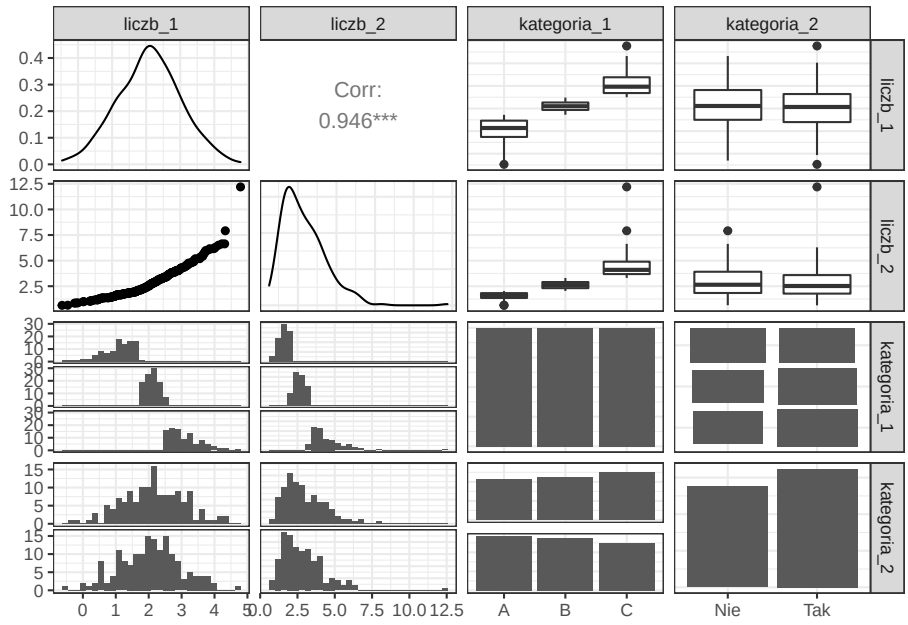
```
dane <- data.frame(liczb_1 = sort(rnorm(300, 2)), liczb_2 = sort(rlnorm(300, 1, 0.5)),
                   kategoria_1 = rep(c("A","B","C"), each=100),
                   kategoria_2 = sample(c("Tak","Nie"),300, replace=TRUE))

library(GGally)
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
ggpairs(dane)
```

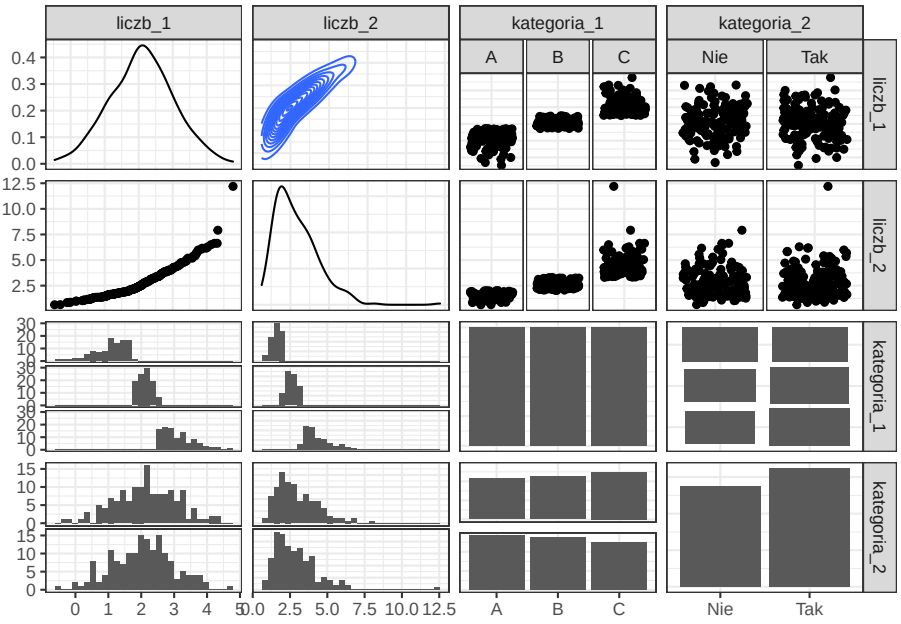
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



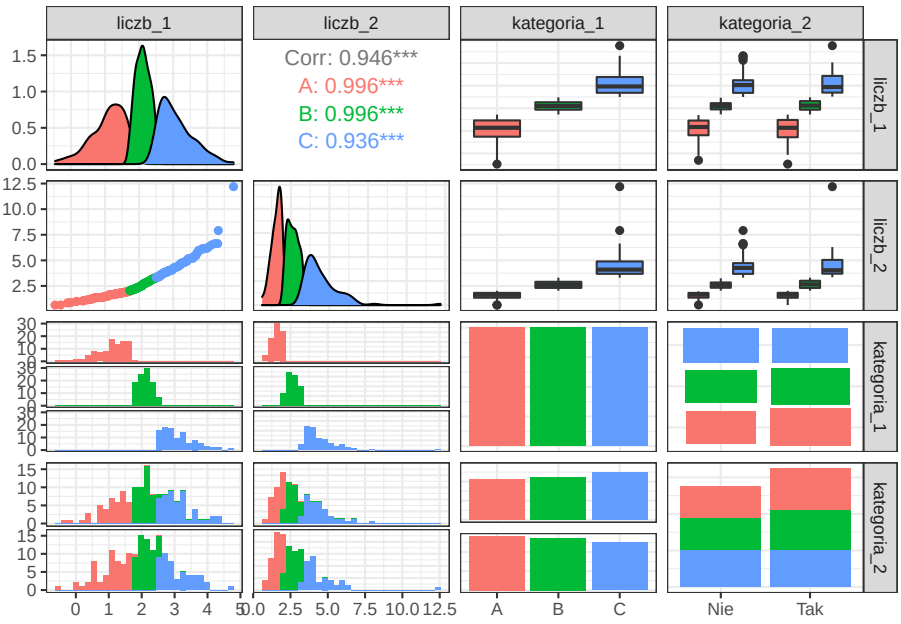

```
# zmiana rodzaju wykresu np. górny panel pokaże wykres gęstości zamiast korelacji
# i kropkowy zamiast boxplota
ggpairs(dane, upper=list(continuous="density", combo="dot"))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
# wykres pokolorowany według jednej z kategorii

ggpairs(dane, mapping = ggplot2::aes(color=kategoria_1))
```

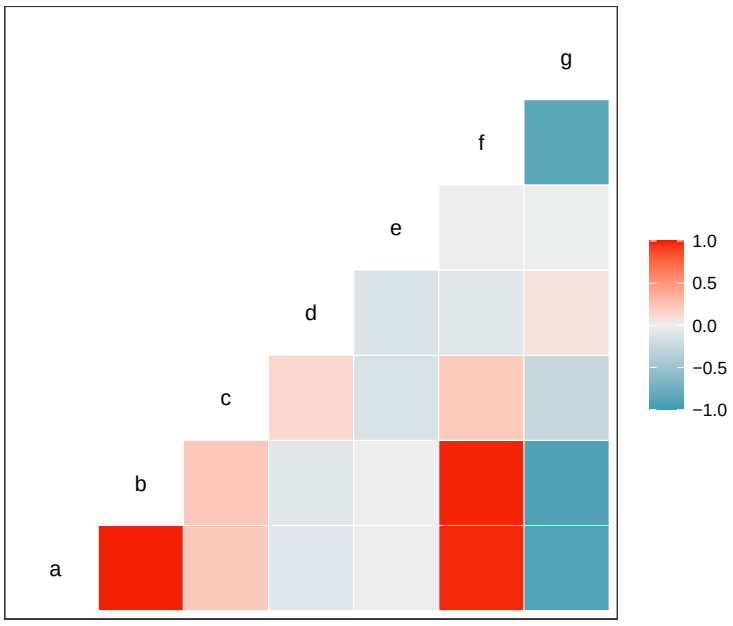
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



4.9.1.2 Macierz korelacji

Tworzenie macierzy korelacji jest opisane w dalszej części z wykorzystaniem pakietu corrplot, ale możliwe jest też użycie ggplot2.

```
dane <- data.frame(a=sort(rnorm(100)), b=sort(rnorm(100)), c=rnorm(100), d=rlnorm(100)
                   f=sort(runif(100)), g=sort(rexp(100), decreasing=TRUE))

ggcorr(dane)
```




```
# z wpisanymi wartościami korelacji

ggcorr(dane, label=TRUE, label_color="black", label_round=2)
```