# Разпределения

############################

# Преди да започнем с характеристиките на разпределенията трябва да знаем, че в R

# всяко разпределение има разновидност, която стартите с d-(име на разпределение),

# p-(име на разпределение), q-(име на разпределение) и r-(име на разпределение). Тези

# означения показват какво искаме да правим:

# d- - изчислява вероятността за сбъдване на събитието x

# p- - изчислява вероятността на предварително зададен квантил за разпределението

# q- - изчислява квантилът на предварително зададена вероятност за разпределението

# r- - генериране на случайни величини за разпределението

# p- е обратна функция на q-. Например qnorm(pnorm(0.9)) = 0.9, pnorm(qnorm(0.9)) = 0.9

############################

# 1. Биномно разпределение

# Биномното разпределение е дисктретно разпределение, което брои успехите в редица

# от n независими опити. Биномното разпределение приема параметри "n" - броя на опитите

# и "p" - вероятността за настъпване на успех.

# Вероятността за настъпване на събитие "x" е P(x) = C(n, x)\*p^x\*(1-p)^(n-x),

# за x = 0, ..., n

# В R, функциите за биномно разпределение са rbinom(), dbinom(), pbinom(), qbinom()

# n - броят на симулациите, size - големианта на редицата от независимите опити

# p - вероятността за настъпване на успех от един опит

# Частния случай, когато size = 1 поражда Бернулиево разпределение. Тоест,

# искаме да генерираме само един опит с вероятност за успех "p"

rbinom(n = 1, size = 1, p = 0.2)

rbinom(n = 10, size = 1, p = 0.2)

# Биномно разпределение

rbinom(n = 1, size = 6, p = 0.2)

rbinom(n = 10, size = 6, p = 0.2)

set.seed(4442)

rbd <- rbinom(n = 100, size = 6, p = 0.2)

# Средната стойност и медианата са равни на size\*p, къдетo "size" и "p" са броят на

# опитите в редицата от експерименти и вероятността за успех.

# Вариацията е равна на size\*p\*(1 - p)

summary(rbd)

var(rbd)

# 2. Геометрично разпределение

# Геометричното разпределение е дискретно разпределение, което показва броя на

# опитите до настъпването на успех.

# Вероятността е P(x) = p\*(1-p)^x, за x = 0, ..., n

set.seed(46871)

rgd <- rgeom(n = 100, prob = 0.2)

# Средната стойност са равни на 1/p, "p" е вероятността за успех.

# Вариацията е равна на (1-p)/(p^2)

summary(rgd)

var(rgd)

# 3. Отрицателно биномно разпределение

# Отрицателното биномно разпределение е дискретно разпределение, което измерва броя

# на успехите в редица от бернулиеви опити до настъпване на r на брой неуспеха. Разпределението

# има два параметъра r (броя на неуспехите) и p (вероятността за успех за всеки опит). При

# r = 1 имаме геометрично разпределение.

# Функцията на масата е P(x) = C(x+r-1, x)\*(1-p)^r\*p^x, за x = 0, 1, ...

# Очакването на отрицателното биномно разпределение е r\*(1-p)/p, а вариацията е r\*(1-p)/p^2

set.seed(4457)

N <- 1000

nbd <- rnbinom(n = N, size = 30, prob = 0.2)

# size - броят на успехите

hist(nbd)

summary(nbd)

var(nbd)

# 4. Поасоново разпределение

# Поасоновото разпределение е дискретно разпределение, което измерва вероятността

# "lambda" (n\*p) на брой независими събития да се случат в определен интервал от време.

# При Поасоновото разпределение "n" клони към безкрайност, a "p" клони към нула. Това

# е и връзката между биномно разпределение и поасоново

# Вероятността е P(x) = exp(-lambda)\*lambda^x/(x!)

# При поасоновото разпределение, средната стойност и вариацията са равни на lambda

set.seed(1477)

rpd <- rpois(n = 100, lambda = 3.4)

summary(rpd)

var(rpd)

# 5. Равномерно непрекъснато разпределение

# Равномерното непрекъснато разпределение много наподобява равнометрното дискретно.

# Разликата е в това, че вместо дискретни стойности, при непрекъснатото може да

# приема в целия интервал.

# Вероятността P(X < x) = (x - a) / (b - a)

# Равномерното непрекъснато разпределение приема параметри a и b, които са

# минимална и максимална стойности.

# Геенрирането на равномерно разпределение става с функцията runif(n, a, b)

runif(1, 0, 2)

runif(5, 0, 2)

set.seed(6674)

x <- runif(1000) # get the random numbers

hist(x, prob = T, breaks = 10)

# 6. Нормално разпределение

# Нормалното разпределение е непрекъснато разпеделение, което има два основни

# параметъра mu (средна стойност) и sigma (стандартно отклонение).

set.seed(96637)

rnd <- rnorm(n = 1000, mean = 3, sd = 4)

summary(rnd)

var(rnd)

# Както не един път сме споменавали, нормалното разпределение има формата на камбана

hist(rnd, prob = TRUE)

# Всяко едно нормално разпределение, с очакване "mu" и стандартно отклонение

# "sigma", може да се стандартизира в нормално разпределение с очакване 0 и стандартно

# отклонение 1. Стандартизацията се нарича Z-score, а формулата е Z = (x - mean(x))/sd(x)

rnd1 <- (rnd - mean(rnd)) / sd(rnd)

# R притежава функция "scale", която го прави автоматично. Функцията е притежава

# параметрите center (булева или числова променлива) и scale (булева или числова

# променлива)

rnd2 <- scale(rnd)

all(rnd1 == rnd2) # Проверяваме дали всички стойности на rnd1 и rnd2 са равни

# 7. Експоненциално

# Друго важно непрекъснато разпределение е екопоненциалното. Това разпределение

# e подходящо за употреба в случаи, когато имаме работа с промеливи, свързани с време.

# Експоненциалното разпределение има само един параметър lambda.

# Експоненциалното разпределение се свързва с Поасоновото разпределение и като с

# него се оценява времето между настъпванията на две събития. Това разпределение се

# разглежда и като непрекъснат аналог на геометричното разпределение.

# Средната стойност на експоненциалното разпределение е равна на 1/lambda, а

# вариацията = 1/lambda^2.

set.seed(7114)

lambda <- 4

red <- rexp(1000, rate = 1/lambda)

hist(red)

summary(red)

var(red)

# 8. t разпределение

# Друго важно непрекъснато разпределение е t разпределението. Това разпределение се

# използва за оценка на параметрите на популацията, когато размерът на извадката е

# малък или когато стандартното отклонение на популацията е неизвестно. T разпределението

# има един параметър nu = n - 1 (n - броят на наблюденията), който представлява степените

# на свобода.

# Средната стойност и медианата на t разпределението са 0, а вариацията nu/(nu-2)

set.seed(7114)

df <- 14

td <- rt(10^4, df = df)

hist(td)

summary(td)

var(td)

# 9. Chi квадрат разпределение

# Chi квадрат разпределението е непрекъснато разпределение, което представлява сума на k

# на брой независими стандартно нормално разпределени величини. Използваме го за тестване на

# хипотези свързани с дисперсията и при определяне на доверителните интервали. Разпределението

# намира приложение и при изследването на категорийните модели и по-точно до колко прогнозите

# на един модел съответстват на реалните стойности.

# Разпределенеито има един параметър (k), показващ степените на свобода.

# Очакването на Chi квадрат е k, а дисперсията - 2\*k

set.seed(11478)

N <- 1000

rch <- rchisq(n = N, df = 30)

hist(rch)

summary(rch)

var(rch)

# 10. Хипергеометрично разпределение

# Това е дискретно разпределение, което описва вероятността от k успеха в n на брой извадка,

# без замествания, взета от крайна популация с размер N и съдържаща K на брой успеха. Разпределението

# намира приложение при изследването дали една популация е overrepresented или underrepresented.

# Функцията на масата е P(x) = C(K, k)\*C((N - K), (n - k)) / C(N, n)

# Очакването е n\*K/N, а вариацията - n\*(K/N)\*((N-K)/N)\*((N-n)/(N-1))

# 11. Boostrap

# Booostrap е метод, който представлява създаване на извадка с големина, равна на

# големината на вектора X и всеки елемент от X може да участва пвоече от веднъж в новата

# извадка.

data(faithful)

names(faithful)

eruptions <- faithful[, "eruptions"]

sample(eruptions, 10, replace = TRUE)

par(mfrow = c(1, 2))

hist(eruptions, breaks = 25)

hist(sample(eruptions, length(eruptions), replace = TRUE), breaks = 25)

par(mfrow = c(1, 1))

# Едно от приложенията на bootstrap метода е при определянето на локацията на разпределение с тежки опашки

# и/или изразена асиметрия. Целта е да се включат в анализа и наблюдения, които се считат за "outlier"-и.

# Асиметрия

# За да видим дали имаме асиметрия в разпределението, ще изчислим статистиката skewness.

# В R има доста пакети, които предлагат тази опция ("DistributionUtils", "fBasics", "moments", "e1071"), но за целта

# ще използваме наша собствена

Skewness <- function(x) {

x\_centred <- x - mean(x)

n <- length(x\_centred)

y <- sqrt(n)\*sum(x\_centred^3) / (sum(x\_centred^2)^(3/2))

list(estim = y\*((1 - 1/n))^(3/2), se = sqrt((6\*n\*(n-1)) / (n-2) / (n+1) / (n+3)))

}

N <- 1000

set.seed(4741)

x1 <- -rexp(N, rate = 1/3); x1\_skewness <- Skewness(x1)

x3 <- rgamma(N, shape = 2, rate = 1/6); x3\_skewness <- Skewness(x3)

x2 <- rnorm(N, mean = 4, sd = 3); x2\_skewness <- Skewness(x2)

par(mfrow = c(1, 3))

hist(x1, xlab = "Values", main = paste("Skewness:", round(x1\_skewness$estim, 3)), col = "red")

hist(x2, xlab = "Values", main = paste("Skewness:", round(x2\_skewness$estim, 3)), col = "forestgreen")

hist(x3, xlab = "Values", main = paste("Skewness:", round(x3\_skewness$estim, 3)), col = "blue")

par(mfrow = c(1, 1))

# Лесно се забелязва, че при отрицателна стойност на skewness имаме асиметрия, при която

# лявата опашка е по-дълга. И обратното - при положителна стойност имаме по-дълга дясна опашка

# Ако стойността е близка до 0, тогава нямаме доказана асиметрия в разпределението.

# - Но какво означава, стойност близка до 0-та, при положение, че в статистиката разлика от 10

# може да бъде незначима, а разлика от 0.00001 да бъде?

# - Това е така и ето защо има и начин за определянето на значимостта ?. Най-лесно съотношението

# abs(estim/se) трябва да бъде по-малко от 2

x1\_skewness$estim / x1\_skewness$se

x2\_skewness$estim / x2\_skewness$se

x3\_skewness$estim / x3\_skewness$se

# Да се върнем към bootstrap алгоритъма за изчисляване на локацията на разпределението. Той се

# състои в следните стъпки:

# 1. Определяме броя (M) на извадките (извадките са с повторения и имат дължина, равна на вектора,

# от който сме взели извадката).

# 2. Изчисляваме средната стойност на всички M извадки и ги съхраняваме в нов вектор.

# 3. От централната гранична теорема (ЦГТ, CLT) следва, че този вектор е с нормално разпределение

# 4. Взимаме средната стойност на този вектор го приемаме за локация на разпределението, а с помощта

# на стандартното отклонение, можем да определоим доверителни интервали

N <- 400

set.seed(9504)

X <- rgamma(n = N, shape = 2, rate = 1/20)

hist(X)

summary(X)

mean(X, trim = 0.05)

M <- 500

Mat <- matrix(nrow = M, ncol = length(X))

# Създаваме матрица, която има M на брой редове и length(X) брой колони

# В тази матрица, по редове ще съхраняваме извадките

set.seed(10)

for(row\_index in 1:nrow(Mat)) {

Mat[row\_index, ] <- sample(x = X, size = length(X), replace = TRUE)

}

# С долния ред взичмаме всички средни стойности по редове

mu\_array <- apply(Mat, 1, FUN = mean)

par(mfrow = c(1, 2))

hist(mu\_array, main = "Histrogram of mu's distribution", xlab = "Values", col = "pink")

qqnorm(mu\_array); qqline(mu\_array)

par(mfrow = c(1, 1))

# Проверката на хипотези е важна част от анализа на данни. В сегашния случай ще проверим

# хипотезата, че разпределението е нормално. За целта ще използваме тест, който проверява хипотезата

shapiro.test(mu\_array)

# Стойността p-value > 0.05 => разпределението може да го приемем за нормално.

# Понеже не сме учили тестовете, към настоящия момент ще ви се наложи да ми се доверите.

# По-нататък в упражненията ще бъдат засегнати тестовете подробно.

summary(X)

mean(X, trim = 0.05)

(loc <- mean(mu\_array))

# Предимството на този тест е, че можем да определим някакви неблагоприятни стойности, които да

# използваме вместо реалното очакване.

# Така например, ако искам в 95% от случаите да съм познал средната си стойност, тогава взимам

# 5% квантил.

(a <- quantile(mu\_array, prob = 0.05))

hist(mu\_array, main = "Histrogram of mu's distribution", xlab = "Values", col = "orange")

abline(v = loc, lwd = 3, col = "blue")

abline(v = a, lwd = 3, lty = 3, col = "forestgreen")