Limit Theorems

2020

Convergence

Almost sure convergence (a.s) / Почти сигурна сходимост (п.с.)/

A sequence of random variables $X_1, X_2, \ldots$ converges to a random variable X **almost surely**, when the probability that the sequence X_n converges to X is equal to 1.

$$X_n \xrightarrow{a.s.} X$$

$$\mathbb{P}(X_n \longrightarrow X) = 1$$

$$\forall \varepsilon > 0,\ \mathbb{P}(\lim\ \mathrm{Sup}\,|X_n - X| > \varepsilon) \underset{n\to\infty}{\longrightarrow} 0$$

Convergence in probability / Сходимост по вероятност/

A sequence of random variables $X_1, X_2, \ldots$ converges to a random variable X **in probability**, when for all $\varepsilon > 0$ the probability $\mathbb{P}(|X_n - X| \geq \varepsilon)$ goes to 0 when $n \to \infty$.

$$X_n \xrightarrow{\mathbb{P}} X$$

$$\forall \varepsilon > 0,\ \mathbb{P}(|X_n - X| > \varepsilon) \underset{n\to\infty}{\longrightarrow} 0$$

Convergence in distribution / Сходимост по разпределение/

A sequence of random variables $X_1, X_2, \ldots$ converges to a random variable X **in distribution**, when in any point of continuity of the limiting cumulative distribution function F_X the cumulative distribution function of X_n converges to F_X when n goes to infinity.

$$X_n \xrightarrow{d} X$$
$$F_{X_n}(x) \underset{n\to\infty}{\longrightarrow} F_X(x),$$

for all x which are points of continuity of F_X.

L^p convergence /L^p сходимост/

We can define the distance between X_n and X as $\mathbb{E}(|X_n - X|^p)$. If this distance get smaller and smaller for $n \to \infty$ we call this convergence L^p convergence.

$$X_n \xrightarrow{L^p} X$$
$$\mathbb{E}(|X_n - X|^p) \underset{n\to\infty}{\longrightarrow} 0$$

L^p convergence for $p = 1$ is called **convergence in mean**.

Implications

$$X_n \xrightarrow{a.s} X \Rightarrow X_n \xrightarrow{\mathbb{P}} X \Rightarrow X_n \xrightarrow{d} X$$
$$X_n \xrightarrow{L^p} X \Rightarrow X_n \xrightarrow{\mathbb{P}} X$$

If the sequence is **monotone** the almost sure convergence $\xrightarrow{a.s}$ coincides with convergence in probability $\xrightarrow{\mathbb{P}}$.

.

If the limit is a **constant** convergence in probability $\xrightarrow{\mathbb{P}}$ coincides with convergence in distribution $\xrightarrow{d}$.

Law of large numbers (LLN) / Закона за големите числа (ЗГЧ)/

Strong law of large numbers /Силен закон за големите числа/ states that the average $\overline{X}_n$ of a large number of i.i.d. random variables $X_1, X_2, \ldots, X_n$ almost sure converges to the expected value $\mathbb{E}[X]$, when it exists.

$$\frac{X_1 + X_2 + \ldots + X_n}{n} \xrightarrow[n\to\infty]{a.s} \mathbb{E}[X], \mathbb{E}[X] < \infty$$

$$\overline{X}_n \xrightarrow[n\to\infty]{a.s} \mathbb{E}[X], \mathbb{E}[X] < \infty$$

Weak law of large numbers /Слаб закон за големите числа/ states that the average $\overline{X}_n$ of a large number of i.i.d. random variables $X_1, X_2, \ldots, X_n$ converges in probability to the expected value $\mathbb{E}[X]$, when it exists.

$$\overline{X}_n \xrightarrow[n\to\infty]{\mathbb{P}} \mathbb{E}[X], \mathbb{E}[X] < \infty$$

Particular case:

When we have an **indicator** variables $I_A \in Bernoulli(p)$ the average is equal to the relative frequency of the event A. As far $\mathbb{E}[I_A] = \mathbb{P}(A)$ then

$$\frac{I_{A_1} + I_{A_2} + \ldots + I_{A_n}}{n} \xrightarrow[n\to\infty]{\mathbb{P}} \mathbb{P}(A)$$
$$X \in DUnif(1,6)$$

Example - Dice

Let X be the number of points on the upper side of a 6 wall regular dice.

$X \in DUnif(1,6)$

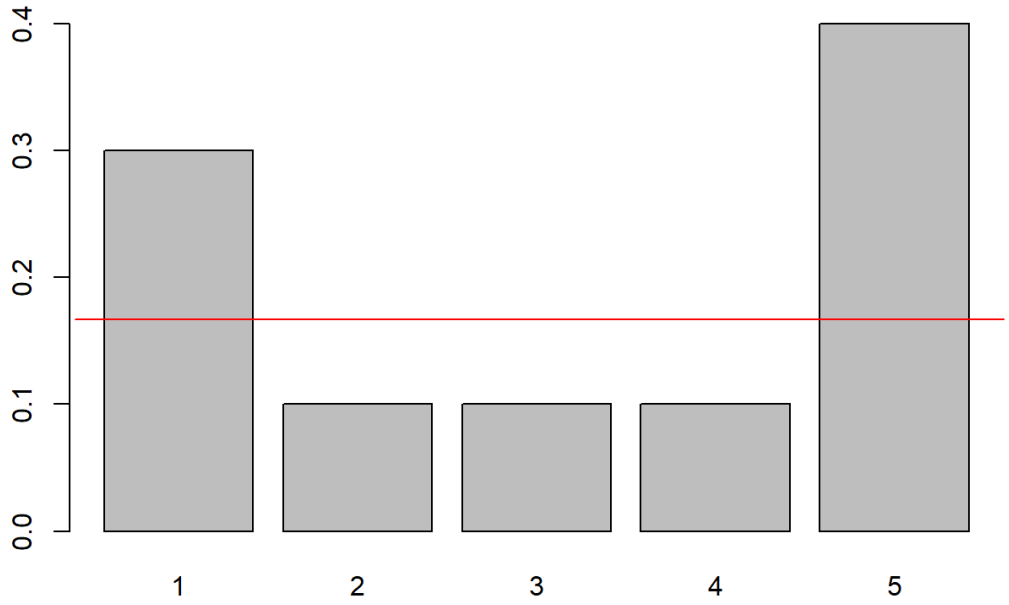
i	1	2	3	4	5	6
$\mathbb{P}(X = i)$	$\frac{1}{6}$	$\frac{1}{6}$	$\frac{1}{6}$	$\frac{1}{6}$	$\frac{1}{6}$	$\frac{1}{6}$

$$\mathbb{E}[X] = \sum_{i=1}^{n} i\mathbb{P}(X = i) = \frac{1}{6}\sum_{i=1}^{6} i = \frac{1 + 2 + \ldots + 6}{6} = \frac{6 \times 7}{2 \times 6} = 3.5$$

$$\overline{X}_n = \frac{X_1 + X_2 + \ldots + X_n}{n}$$

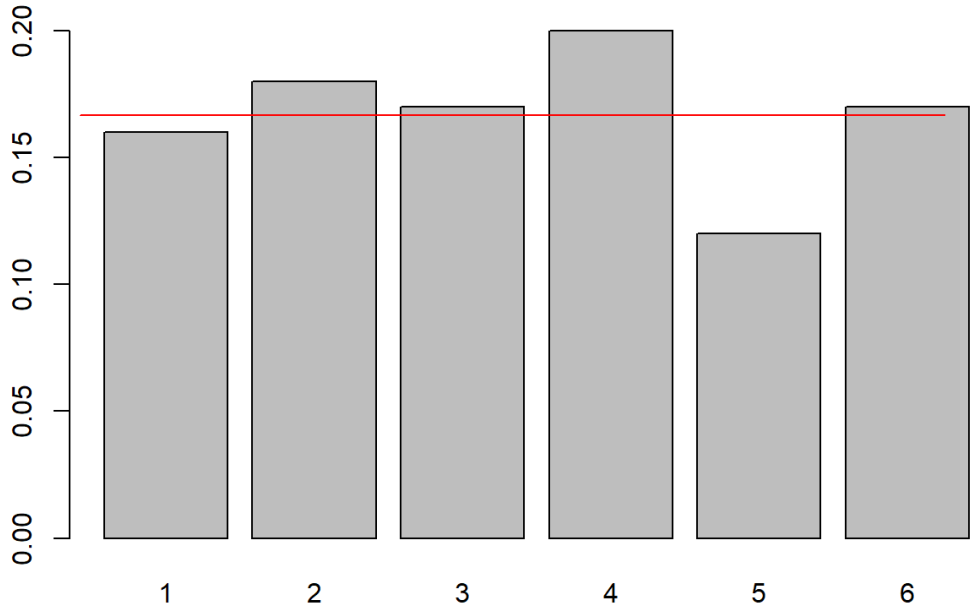
Simulate throwing a dice 10 times. Compute the average and estimate the probability mass function of X.

```
> x <- sample(1:6, 10, replace = TRUE)
> mean(x)
[1] 3.2
> table(x) / length(x)
x
  1   2   3   4   5
0.3 0.1 0.1 0.1 0.4
> barplot(table(x) / length(x))
> lines(c(0, 7), c(1/6, 1/6), col = "red")
```



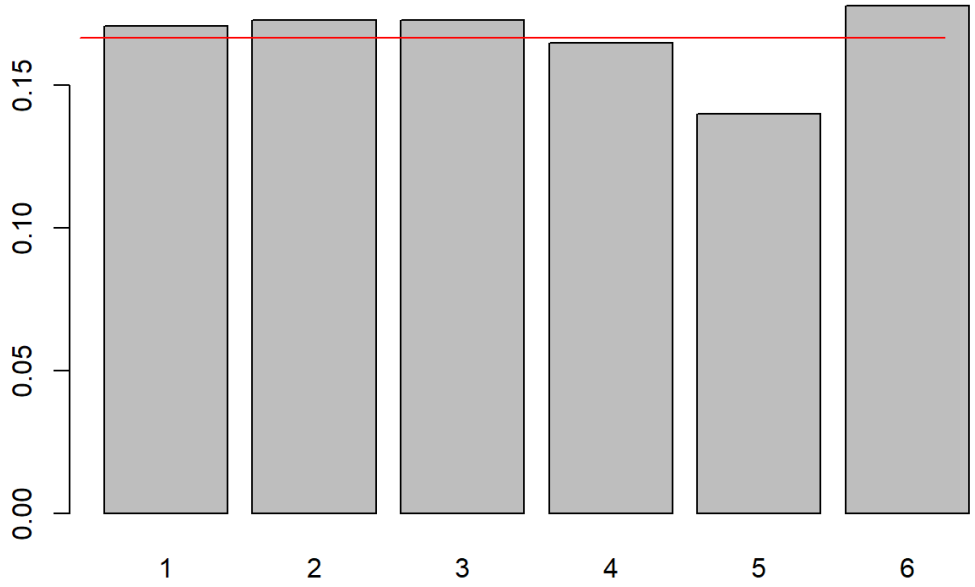
Simulate throwing a dice 100 times. Compute the average and estimate the probability mass function of X.

```
> x <- sample(1:6, 10^2, replace = TRUE)
> mean(x)
[1] 3.45
> table(x) / length(x)
x
   1    2    3    4    5    6
0.16 0.18 0.17 0.20 0.12 0.17
> barplot(table(x) / length(x))
> lines(c(0, 7), c(1/6, 1/6), col = "red")
```



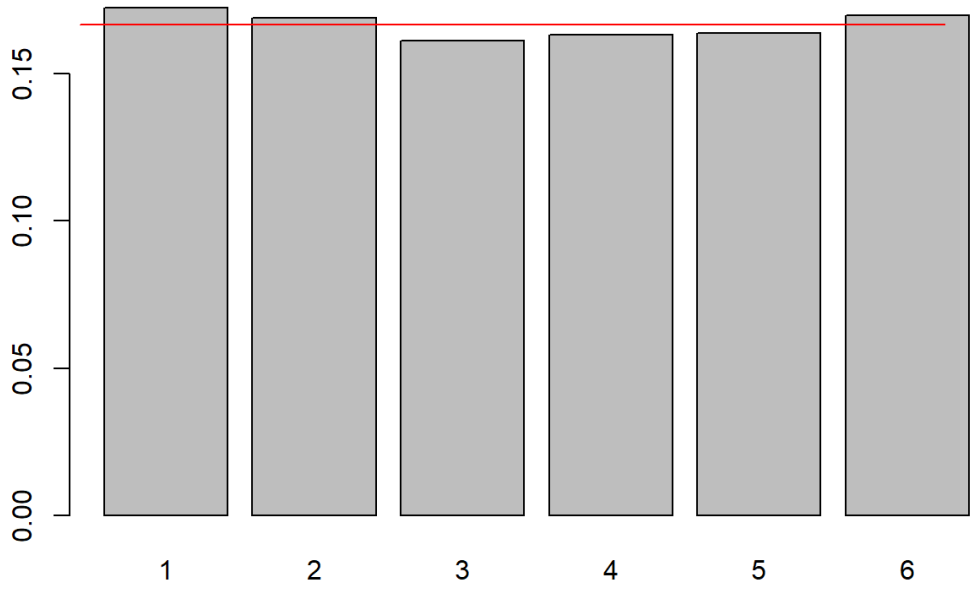
Simulate throwing a dice 1000 times. Compute the average and estimate the probability mass function of X.

```
> x <- sample(1:6, 10^3, replace = TRUE)
> mean(x)
[1] 3.464
> table(x) / length(x)
x
    1     2     3     4     5     6
0.171 0.173 0.173 0.165 0.140 0.178
> barplot(table(x) / length(x))
> lines(c(0, 7), c(1/6, 1/6), col = "red")
```



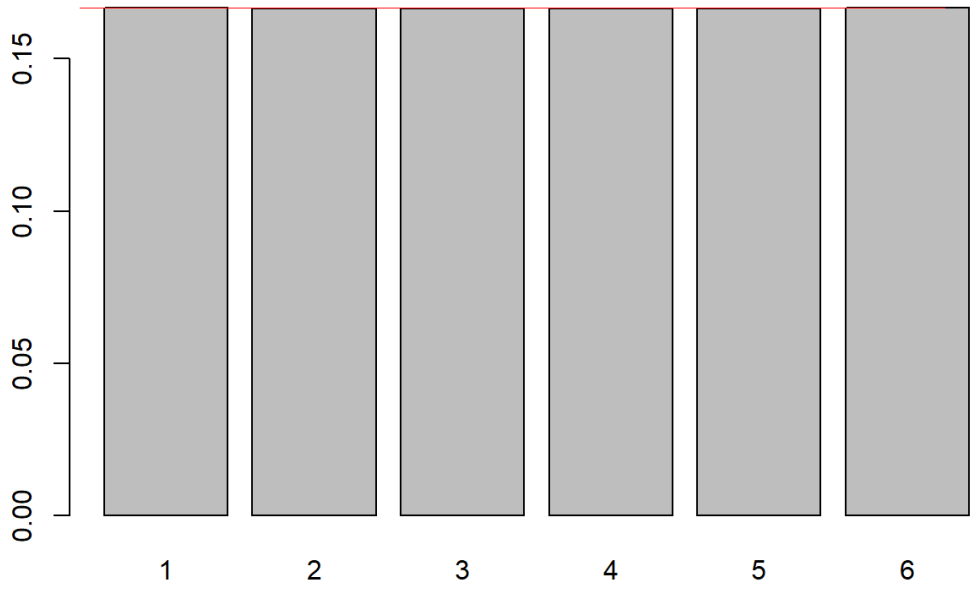
Simulate throwing a dice 10 000 times. Compute the average and estimate the probability mass function of X.

```
> x <- sample(1:6, 10^4, replace = TRUE)
> mean(x)
[1] 3.4866
> table(x) / length(x)
x
     1      2      3      4      5      6
0.1725 0.1690 0.1614 0.1634 0.1639 0.1698
> barplot(table(x) / length(x))
> lines(c(0, 7), c(1/6, 1/6), col = "red")
```



Simulate throwing a dice 10 000 000 times. Compute the average and estimate the probability mass function of X.

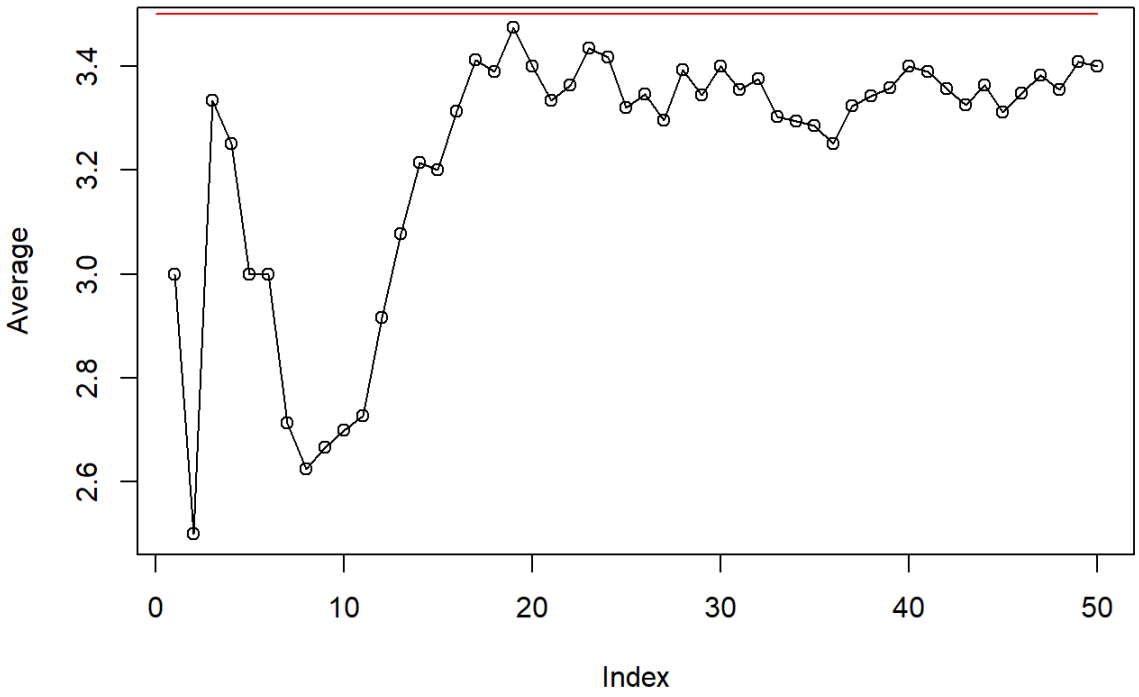
```
> x <- sample(1:6, 10^7, replace = TRUE)
> mean(x)
[1] 3.500075
> table(x) / length(x)
x
        1         2         3         4         5
6
0.1668045 0.1665506 0.1666003 0.1666385 0.1666219
0.1667842
> barplot(table(x) / length(x))
> lines(c(0, 7), c(1/6, 1/6), col = "red")
```



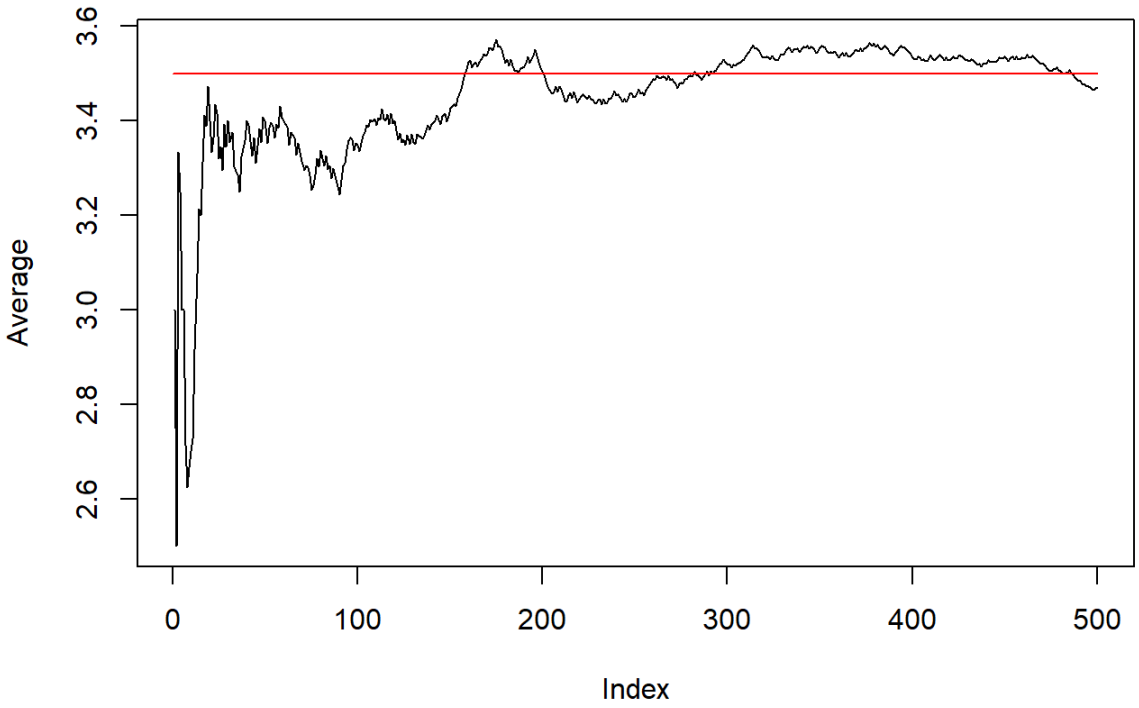
What do you observe? The average gets closer and closer to the theoretical mean of the observed random variable. The relative frequency approaches the theoretical probability of the observed event. These confirm the two forms of the law of large numbers.

Let's make a plot showing how is the average changing when we have more and more observations.

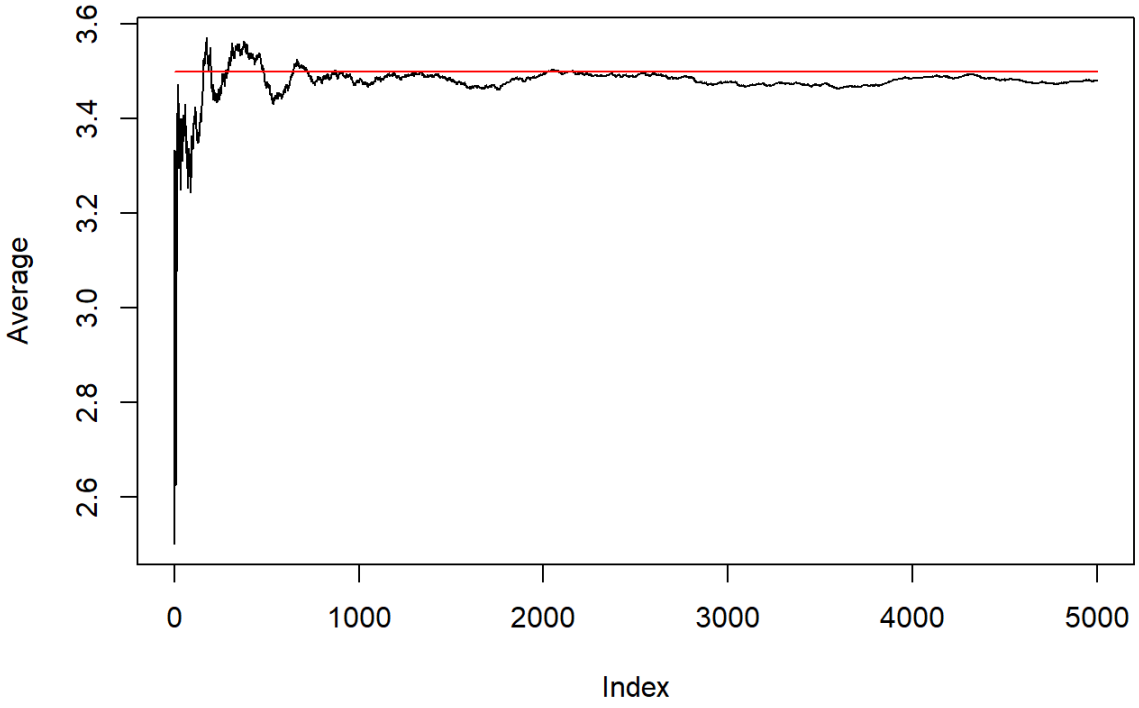
```
> n <- 5000
> x <- sample(1:6, n, replace = TRUE); x[1:10]
 [1] 3 2 5 3 2 3 1 2 3 3
> m <- cumsum(x) / 1:n; m[1:10]
 [1] 3.000000 2.500000 3.333333 3.250000 3.000000
3.000000 2.714286 2.625000
 [9] 2.666667 2.700000
> k <- 50
> plot(m[1:k], ylab = "Average", type = "o")
> lines(c(0, k), c(3.5, 3.5), col = "red")
```




```
> k <- 500
> plot(m[1:k], ylab = "Average", type = "l")
> lines(c(0, k), c(3.5, 3.5), col = "red")
```



```
> k <- 5000
> plot(m[1:k], ylab = "Average", type = "l")
> lines(c(0, k), c(3.5, 3.5), col = "red")
```

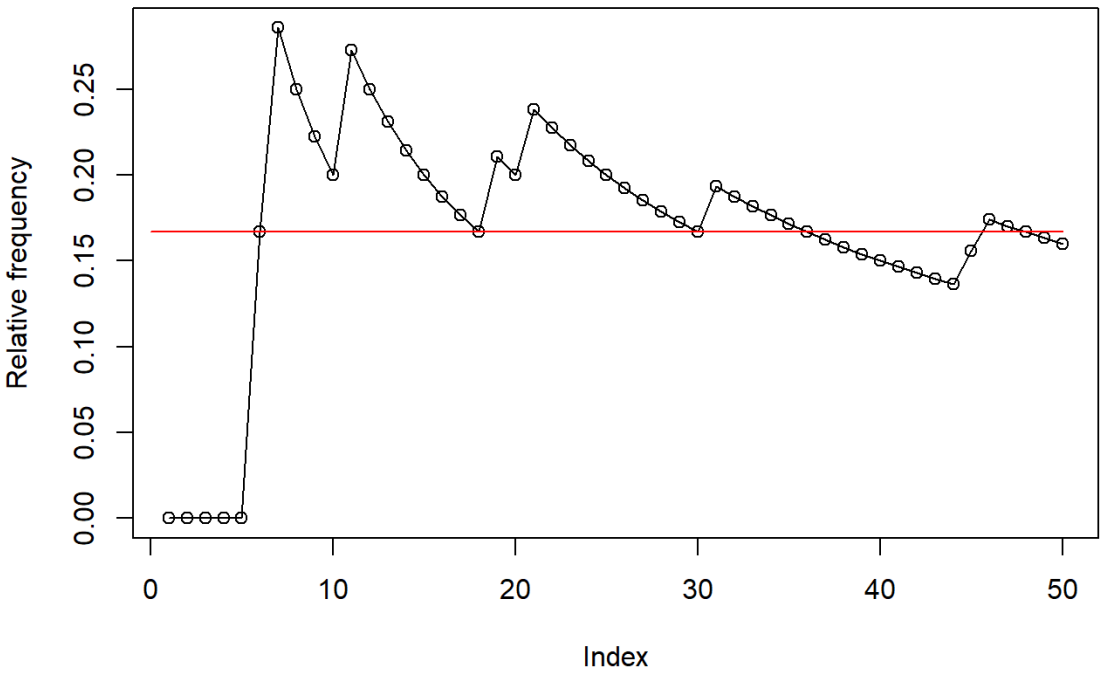


We again see that the average gets closer and closer to the theoretical mean of the observed random variable when the sample size increases.

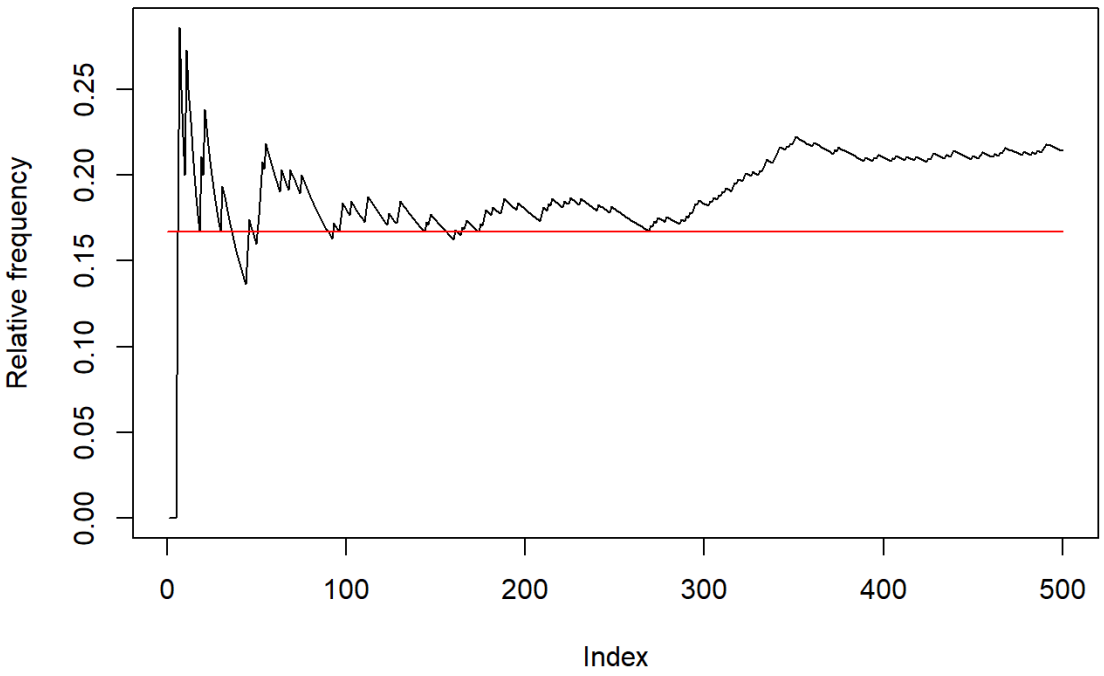
Let us now plot how the relative frequencies of the outcome “To have 6 points” changes when the sample size increases.

```
> n <- 5000
> x <- sample(1:6, n, replace = TRUE); x[1:20]
 [1] 3 3 5 5 5 6 6 2 1 3 6 4 5 2 1 5 1 1 6 2
> numSixes <- cumsum(x == 6); numSixes[1:20]
 [1] 0 0 0 0 0 1 2 2 2 2 3 3 3 3 3 3 3 3 4 4
> p <- numSixes / 1:n; p[1:20]
 [1] 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
0.1666667 0.2857143
 [8] 0.2500000 0.2222222 0.2000000 0.2727273 0.2500000
0.2307692 0.2142857
[15] 0.2000000 0.1875000 0.1764706 0.1666667 0.2105263
0.2000000
> k <- 50
> plot(p[1:k], ylab = "Relative frequency", type = "o")
```

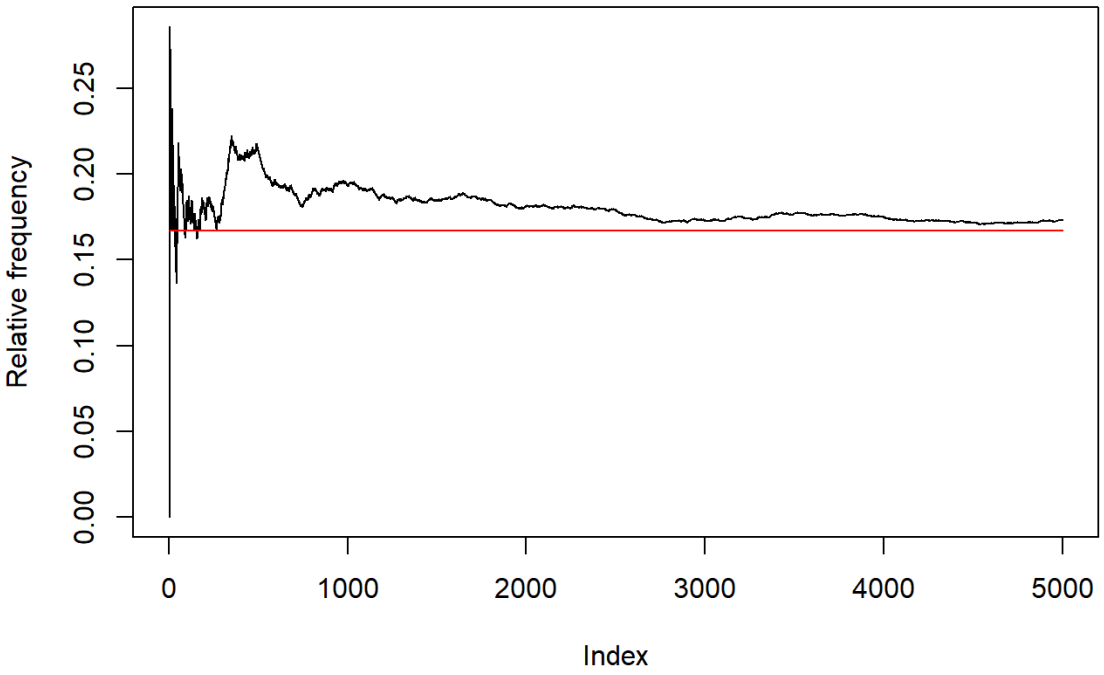
```
> lines(c(0, k), c(1/6, 1/6), col = "red")
```




```
> k <- 500
> plot(p[1:k], ylab = "Relative frequency", type = "l")
> lines(c(0, k), c(1/6, 1/6), col = "red")
```



```
> k <- 5000
> plot(p[1:k], ylab = "Relative frequency", type = "l")
> lines(c(0, k), c(1/6, 1/6), col = "red")
```



We again see that the relative frequency approaches the theoretical probability of the observed event when the sample size increases.

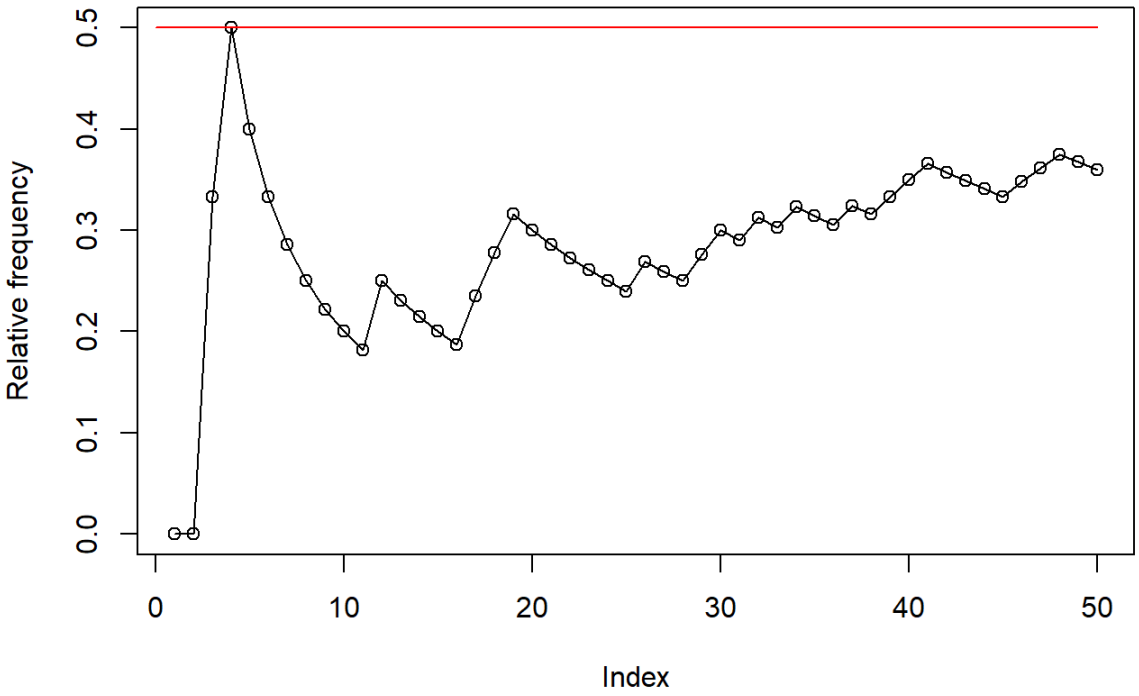
Example - Coin

Simulate 50000 flips of a coin. Let's call "Success" the event "To have a Head".

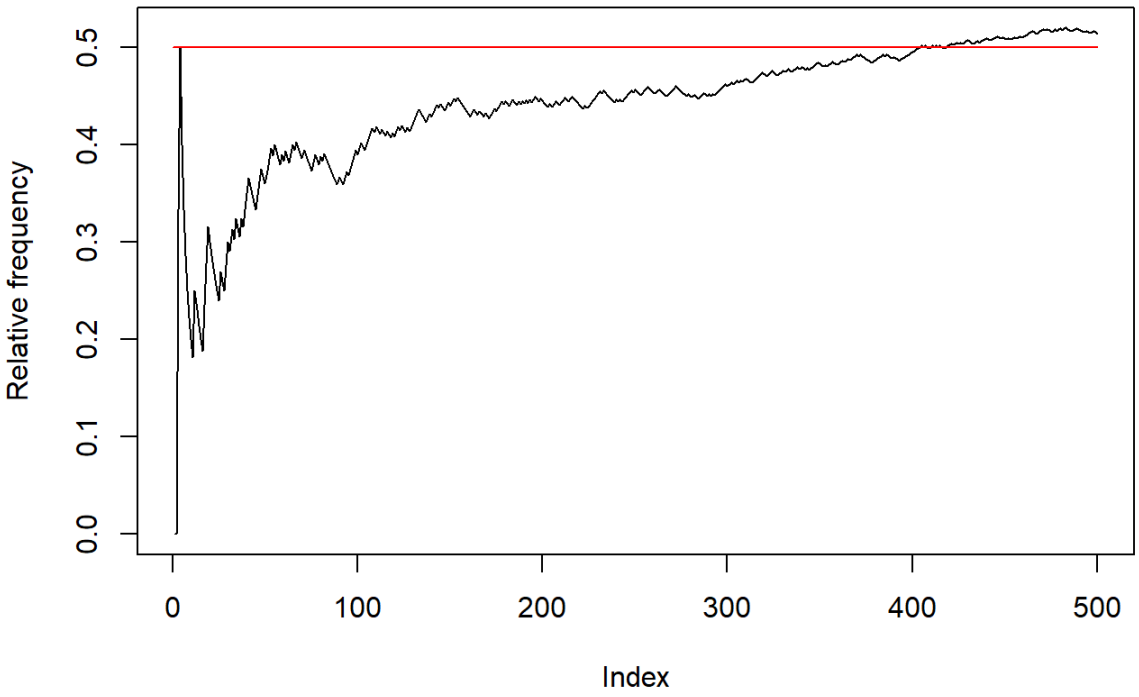
Compute the relative frequency of successes and show how it is converging to the probability for success.

```
> n <- 5000
> x <- sample(c("H","T"), n, replace = TRUE); x[1:15]
 [1] "T" "T" "H" "H" "T" "T" "T" "T" "T" "T" "T" "H" "T"
"T" "T"
> numHeads <- cumsum(x == "H"); numHeads[1:15]
 [1] 0 0 1 2 2 2 2 2 2 2 2 3 3 3 3
> p <- numHeads / (1:n); p[1:15]
 [1] 0.0000000 0.0000000 0.3333333 0.5000000 0.4000000
0.3333333 0.2857143
```

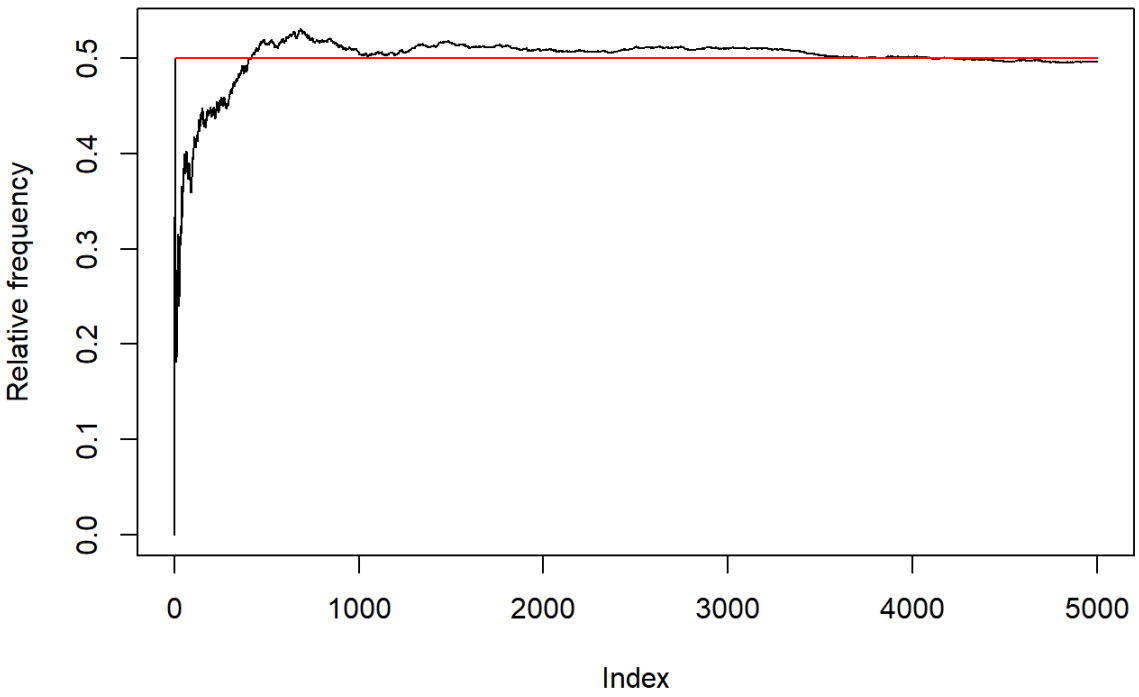
```
 [8] 0.2500000 0.2222222 0.2000000 0.1818182 0.2500000
0.2307692 0.2142857
[15] 0.2000000
> k <- 50
> plot(p[1:k], ylab = "Relative frequency", type = "o")
> lines(c(0, k), c(1/2, 1/2), col = "red")
```




```
> k <- 500
> plot(p[1:k], ylab = "Relative frequency", type = "l")
> lines(c(0, k), c(1/2, 1/2), col = "red")
```




```
> k <- 5000
> plot(p[1:k], ylab = "Relative frequency", type = "l")
> lines(c(0, k), c(1/2, 1/2), col = "red")
```



We observe that the relative frequency of successes approaches the theoretical probability for success when the sample size increases.

Central limit theorem (CLT) / Централна гранична теорема (ЦГТ)/

Central limit theorem for the sum /Централна гранична теорема за суми/

Let $X_1, X_2, \ldots, X_n$ be a sample of independent observations of a random variable X with mean μ and finite variance $\sigma^2 < \infty$. When the sample size increases the sum $S_n := X_1 + X_2 + \ldots + X_n$ centered and normalized converges in distribution to a standard normal random variable.

$$X_1 + X_2 + \ldots + X_n \overset{d}{\approx} N(n\mu, n\sigma^2)$$

$$\frac{S_n - n\mu}{\sigma\sqrt{n}} \xrightarrow{d} Z, Z \in N(0, 1)$$

Central limit theorem for the average /Централна гранична теорема за средното аритметично/

$$\frac{X_1 + X_2 + \ldots + X_n}{n} \overset{d}{\approx} N\left(\mu, \frac{\sigma^2}{n}\right)$$

$$\frac{\frac{S_n}{n} - \frac{n\mu}{n}}{\frac{\sigma\sqrt{n}}{n}} = \frac{\overline{X}_n - \mu}{\frac{\sigma}{\sqrt{n}}} \xrightarrow{d} Z, Z \in N(0, 1)$$

Special case:

Moivre-Laplace theorem

Probability mass function of the random number of “successes” observed in a series of n independent Bernulli trails, each with

probability of success p, approaches the probability density function of the normal distribution $N(np, npq)$ as n grows larger, where $p \in (0,1)$.

$$S_n = I_1 + I_2 + \ldots + I_n \overset{d}{\approx} N(np, npq)$$

$$\frac{S_n - np}{\sqrt{npq}} \xrightarrow{d} Z, Z \in N(0,1)$$

Then the relative frequency of successes is

$$\frac{S_n}{n} = \frac{I_1 + I_2 + \ldots + I_n}{n} \overset{d}{\approx} N\left(p, \frac{pq}{n}\right)$$

$$\frac{\frac{S_n}{n} - \frac{np}{n}}{\frac{\sqrt{npq}}{n}} = \frac{\frac{S_n}{n} - p}{\sqrt{\frac{pq}{n}}} \xrightarrow{d} Z, Z \in N(0,1)$$

Example - Bean machine (Galton Board)

Bean machine (Galton Board) is a device invented by **Sir Francis Galton** to demonstrate the central limit theorem (CLT).

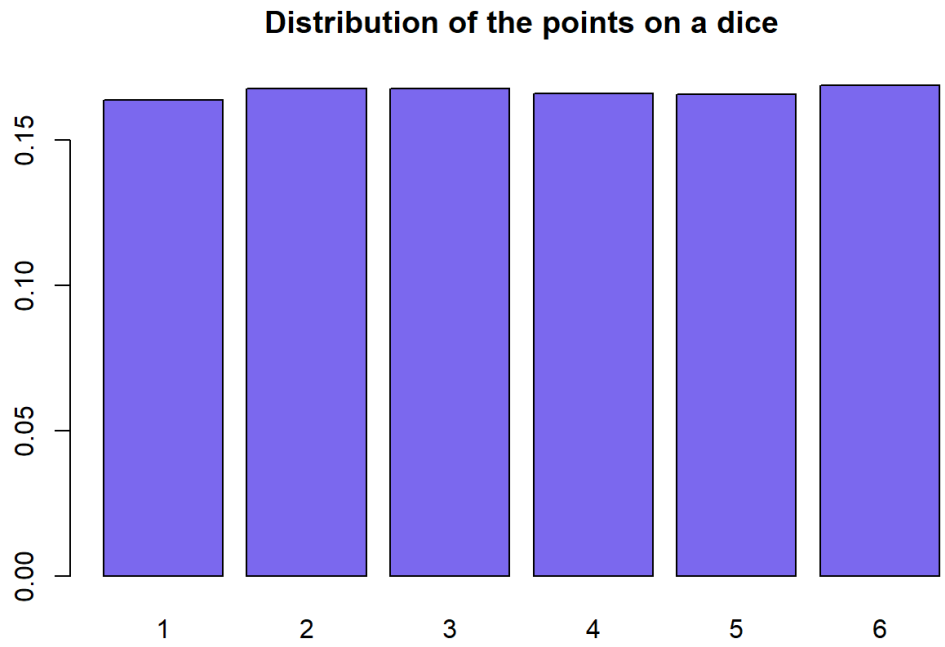
Galton bean machine

Example - Dice

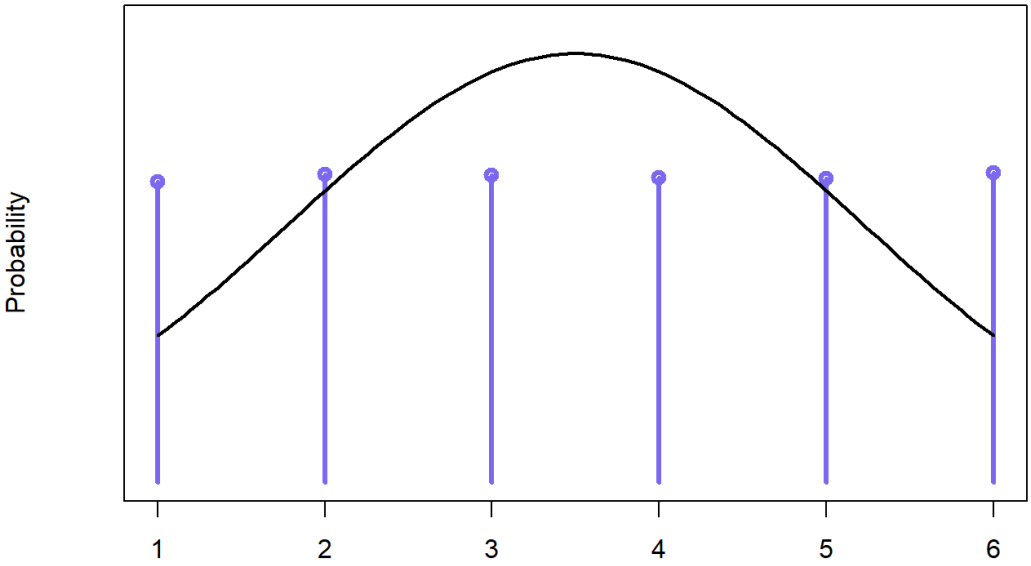
Let's compare the empirical probability mass functions, for the sum of the points on n fair 6-sided dice and to show their convergence to the normal distribution when n increases.

Simulate rolling a dice 100 000 times.

```
> x <- sample(1:6, 10^5, replace = TRUE)
> barplot(table(x) / length(x),
+         main = "Distribution of the points on a dice",
+         col = c("#7B68EE"))
```

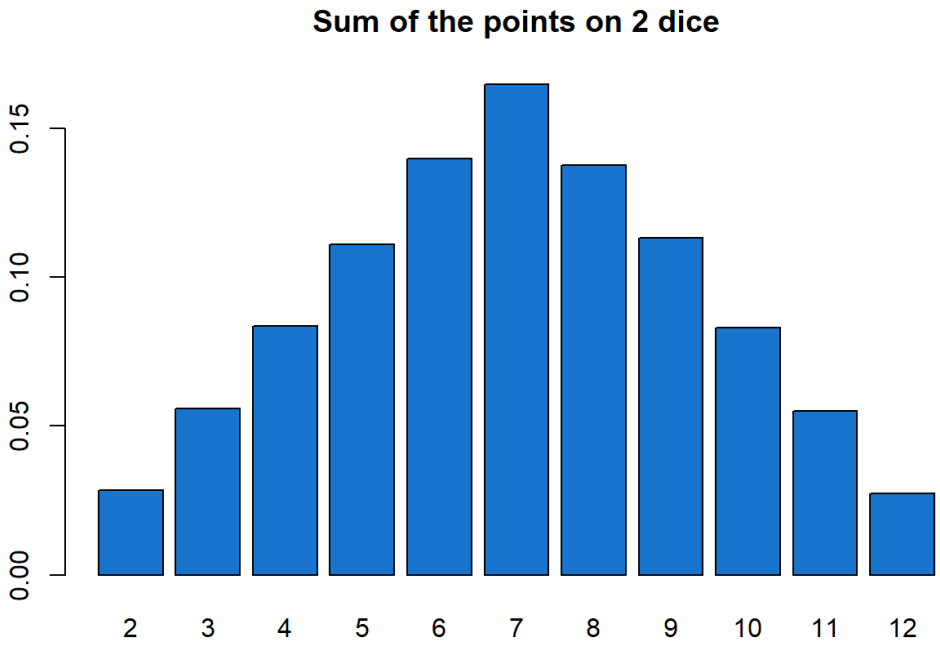



```
> plot(1:6, table(x) / length(x),
+     xlab = "", ylab = "Probability", ylim = c(0, 0.25),
+     type = "h", lwd = 3, col = "#7B68EE")
> points(1:6, table(x) / length(x),
+        type = "p", lwd = 3, col = "#7B68EE")
> m <- (1 + 6) / 2
> v <- ((6 - 1 + 1)^2 - 1) / 12
> curve(dnorm(x, m, sqrt(v)),
+       add = TRUE, col = "black", lwd = 2)
```

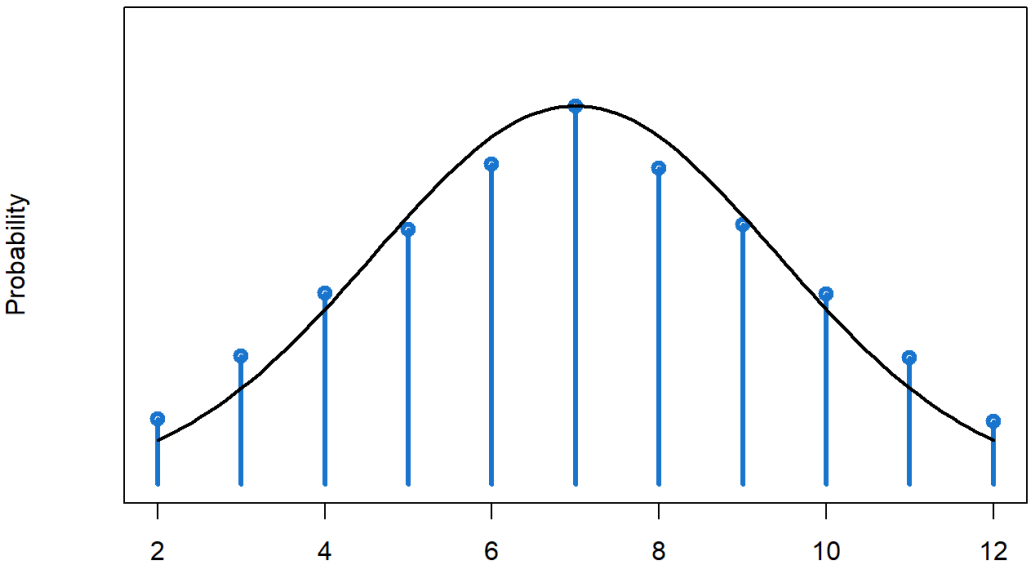


Simulate rolling two dice 100 000 times and calculate the sum of the points on them.

```
> d1 <- sample(1:6, 10^5, replace = TRUE)
> d2 <- sample(1:6, 10^5, replace = TRUE)
> x <- d1 + d2
> barplot(table(x) / length(x),
+         main = "Sum of the points on 2 dice",
+         col = c("#1874CD"))
```

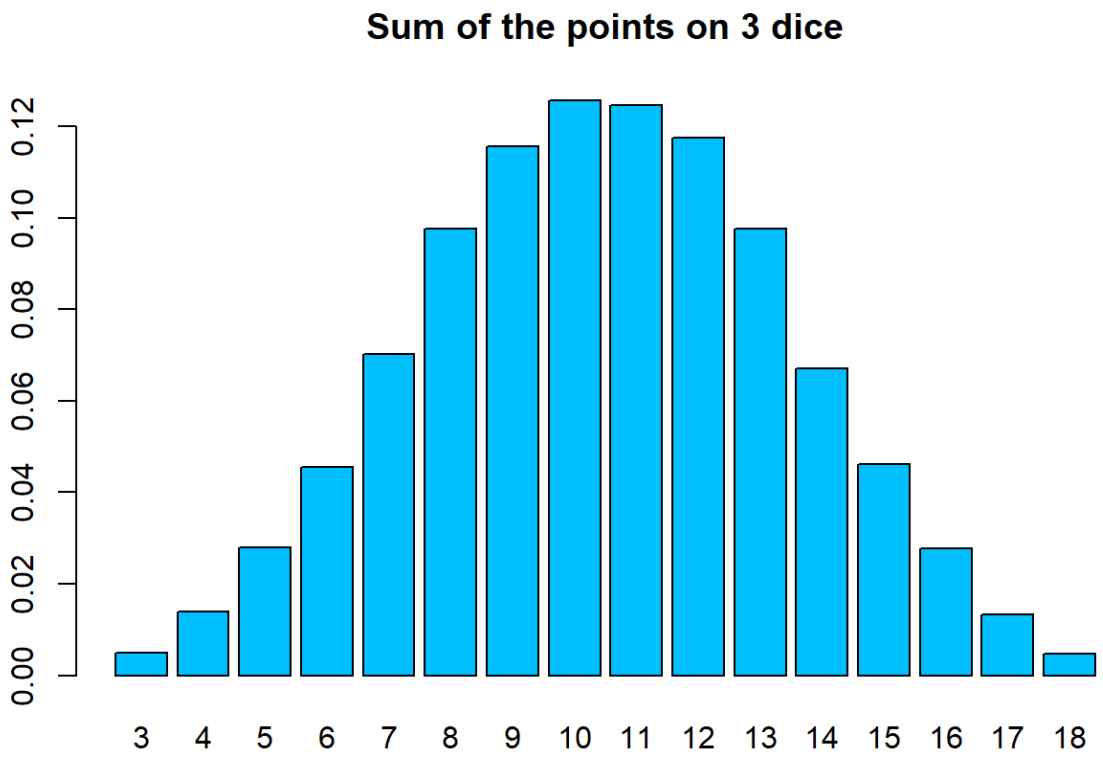



```
> plot(2:12, table(x) / length(x),
+      xlab = "", ylab = "Probability", ylim = c(0, 0.2),
+      type = "h", lwd = 3, col = "#1874CD")
> points(2:12, table(x) / length(x),
+        type = "p", lwd = 3, col = "#1874CD")
> curve(dnorm(x, 2 * m, sqrt(2 * v)),
+       add = TRUE, col = "black", lwd = 2)
```

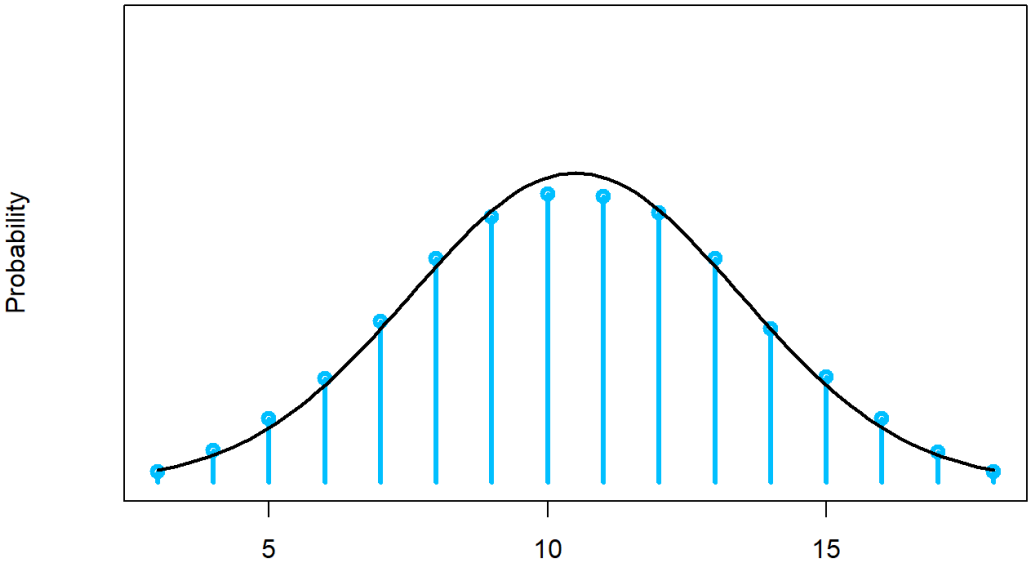


Simulate rolling 3 dice 100 000 times and calculate the sum of the points on them.

```
> di <- replicate(3, sample(1:6, 10^5, replace = TRUE));
head(x)
[1] 11  5  4  6  7  4
> x <- rowSums(di)
> barplot(table(x) / length(x),
+         main = "Sum of the points on 3 dice",
+         col = c("#00BFFF"))
```

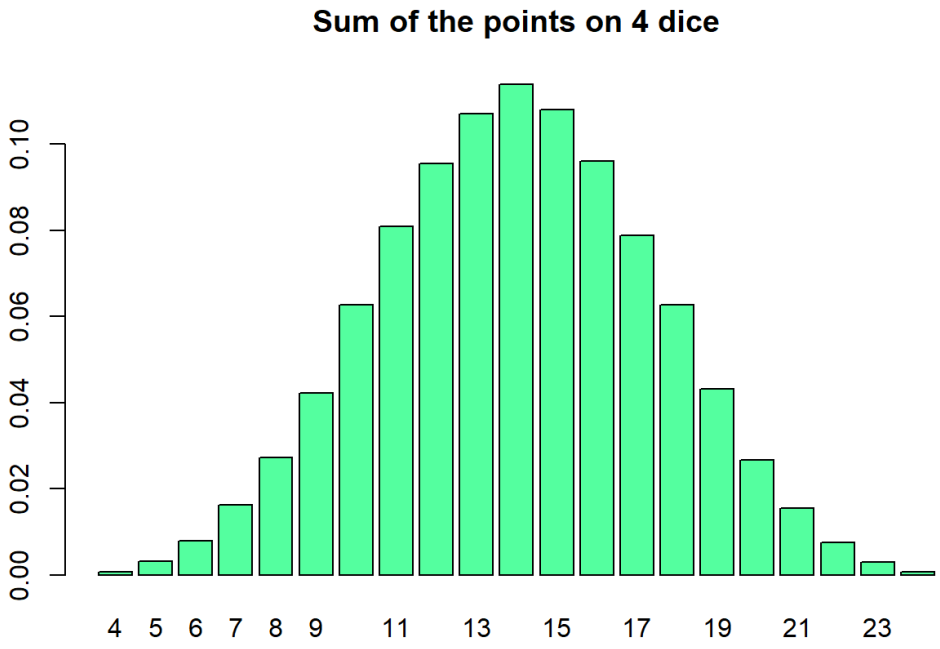



```
> plot(3:18, table(x) / length(x),
+     xlab = "", ylab = "Probability", ylim = c(0, 0.2),
+     type = "h", lwd = 3, col = "#00BFFF")
> points(3:18, table(x) / length(x),
+        type = "p", lwd = 3, col = "#00BFFF")
> curve(dnorm(x, 3 * m, sqrt(3 * v)),
+       add = TRUE, col = "black", lwd = 2)
```

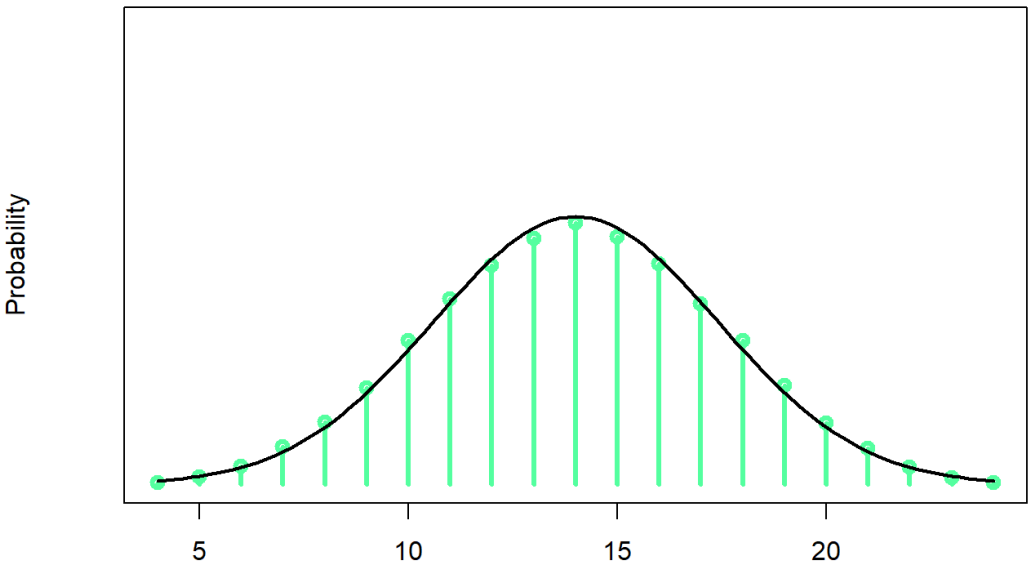


Simulate rolling 4 dice 100 000 times and calculate the sum of the points on them.

```
> di <- replicate(4, sample(1:6, 10^5, replace = TRUE));
head(x)
[1] 18  5  7  4 15 15
> x <- rowSums(di)
> barplot(table(x) / length(x),
+         main = "Sum of the points on 4 dice",
+         col = c("#54FF9F"))
```

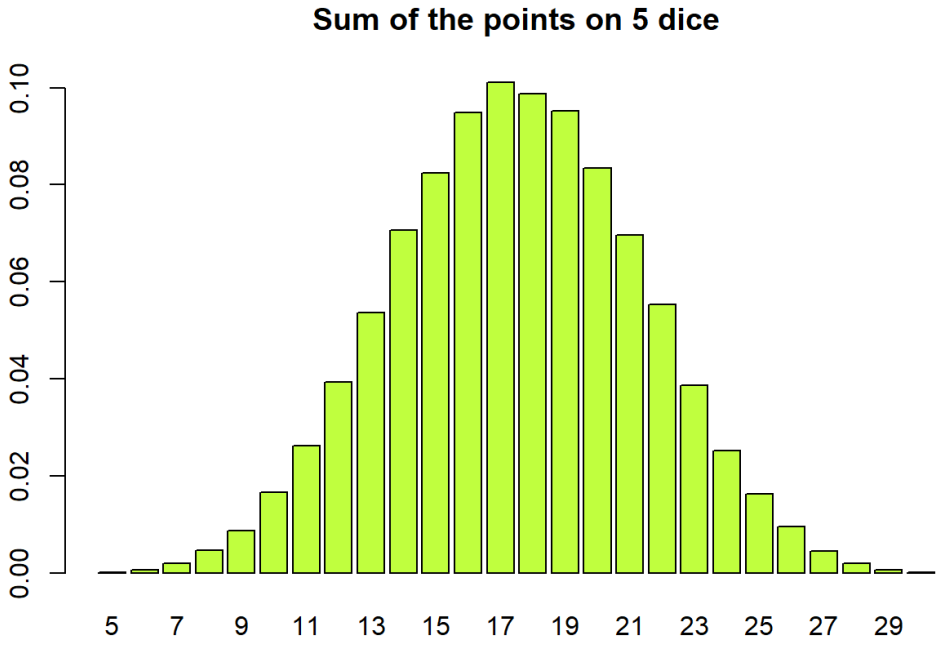



```
> plot(4:24, table(x) / length(x),
+      xlab = "", ylab = "Probability", ylim = c(0, 0.2),
+      type = "h", lwd = 3, col = "#54FF9F")
> points(4:24, table(x) / length(x),
+        type = "p", lwd = 3, col = "#54FF9F")
> curve(dnorm(x, 4 * m, sqrt(4 * v)),
+       add = TRUE, col = "black", lwd = 2)
```

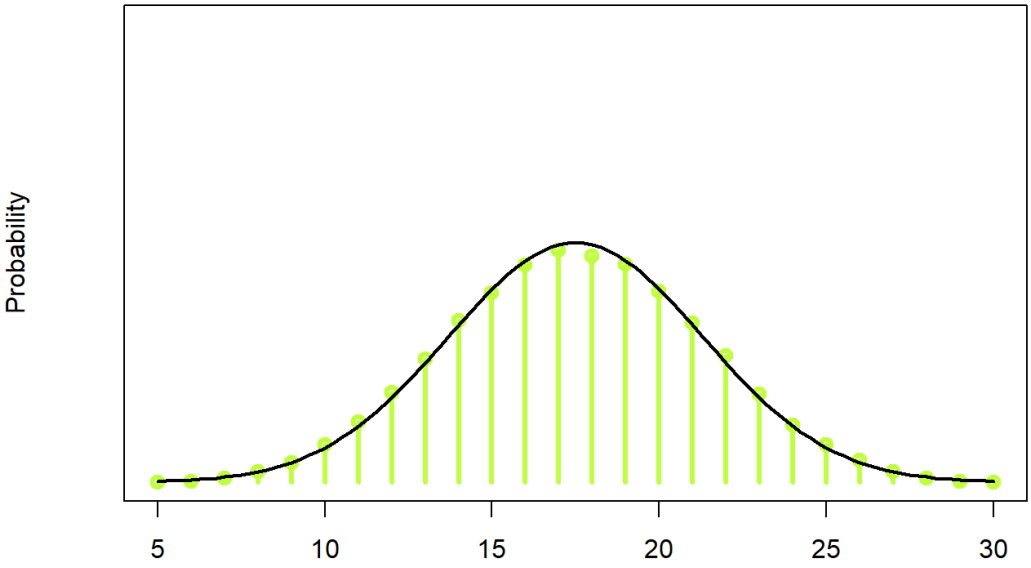


Simulate rolling 5 dice 100 000 times and calculate the sum of the points on them.

```
> di <- replicate(5, sample(1:6, 10^5, replace = TRUE));
head(x)
[1] 21 16 15 12 17 14
> x <- rowSums(di)
> barplot(table(x) / length(x),
+         main = "Sum of the points on 5 dice",
+         col = c("#C0FF3E"))
```




```
> plot(5:30, table(x) / length(x),
+     xlab = "", ylab = "Probability", ylim = c(0, 0.2),
+     type = "h", lwd = 3, col = "#C0FF3E")
> points(5:30, table(x) / length(x),
+        type = "p", lwd = 3, col = "#C0FF3E")
> curve(dnorm(x, 5 * m, sqrt(5 * v)),
+       add = TRUE, col = "black", lwd = 2)
```



Similarly there are functions `colSums` and `colMeans`.

Example - Dice

Let's compare the empirical probability mass functions, for the number of 6's on n fair 6-sided dice and to show their convergence to the normal distribution when n increases.

$$I_A \in Bernoulli(p)$$

$$\mu = \mathbb{E}[I_A] = p = \frac{1}{6}$$

$$\sigma^2 = D[I_A] = pq = \frac{1}{6} \times \frac{5}{6} = \frac{5}{36}$$

$$S_n = I_{A_1} + I_{A_2} + \ldots + I_{A_n} \in Bi(n, p)$$

$$\mathbb{E}[S_n] = np = \frac{n}{6}$$

$$D[S_n] = npq = \frac{1}{6} \times \frac{5}{6} \times n = \frac{5n}{36}$$

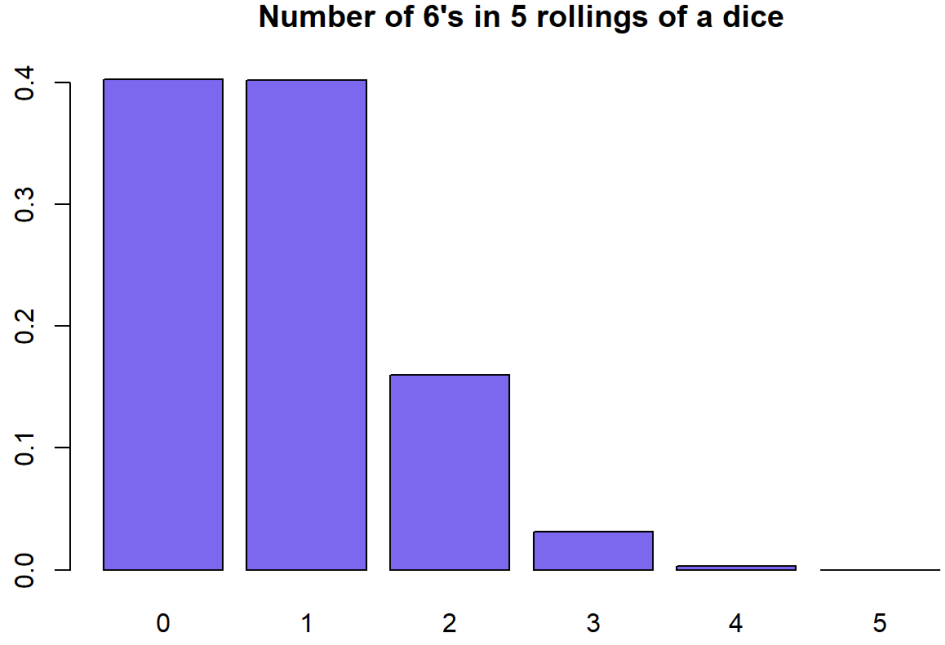
$$S_n = I_{A_1} + I_{A_2} + \ldots + I_{A_n} \overset{d}{\approx} N(np, npq)$$

$$S_n \overset{d}{\approx} N\left(\frac{n}{6}, \frac{5n}{35}\right)$$

$$S_n \overset{d}{\approx} N\left(\frac{10}{6}, \frac{5 \times 10}{36}\right)$$

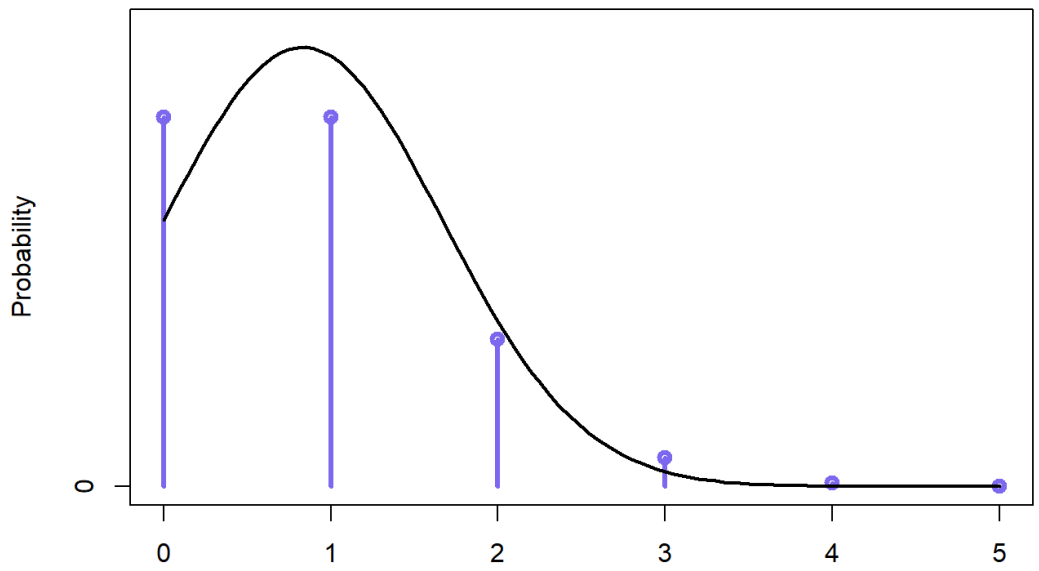
Simulate the number of 6's on 5 rollings of a dice.

```
> n <- 5; p <- 1/6
> x <- rbinom(10^5, n, p)
> barplot(table(x) / length(x),
+         main = "Number of 6's in 5 rollings of a dice",
+         col = c("#7B68EE"))
```



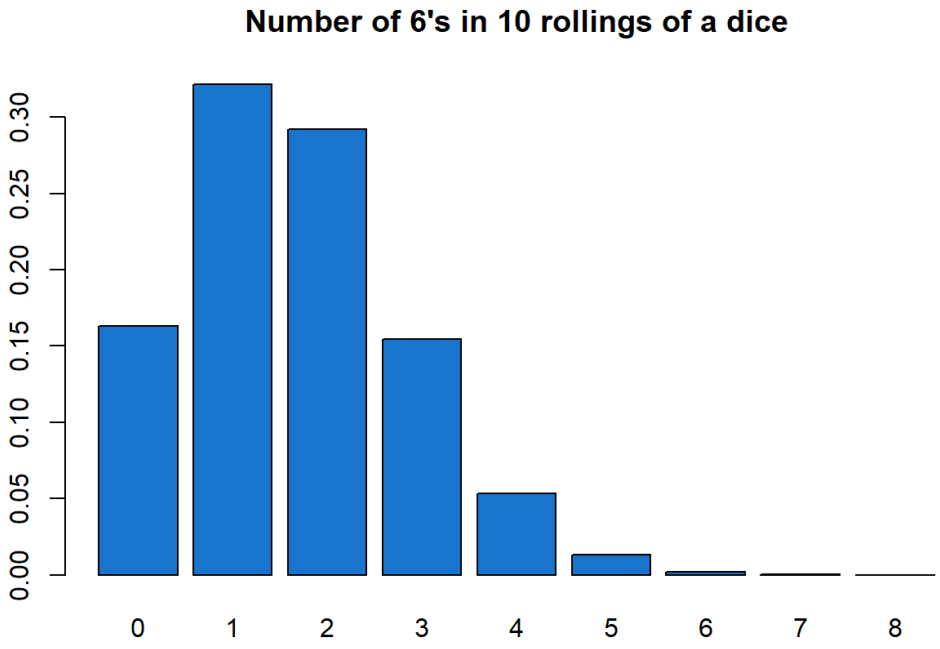

```
> plot(0:(length(table(x)) - 1), table(x) / length(x),
+     xlab = "", ylab = "Probability", ylim = c(0, 0.5),
+     type = "h", lwd = 3, col = c("#7B68EE"))
> points(0:(length(table(x)) - 1), table(x) / length(x),
```

```
+         type = "p", lwd = 3, col = c("#7B68EE"))
> curve(dnorm(x, n * 1 / 6, sqrt(n * 5 / 36)),
+       add = TRUE, col = "black", lwd = 2)
```

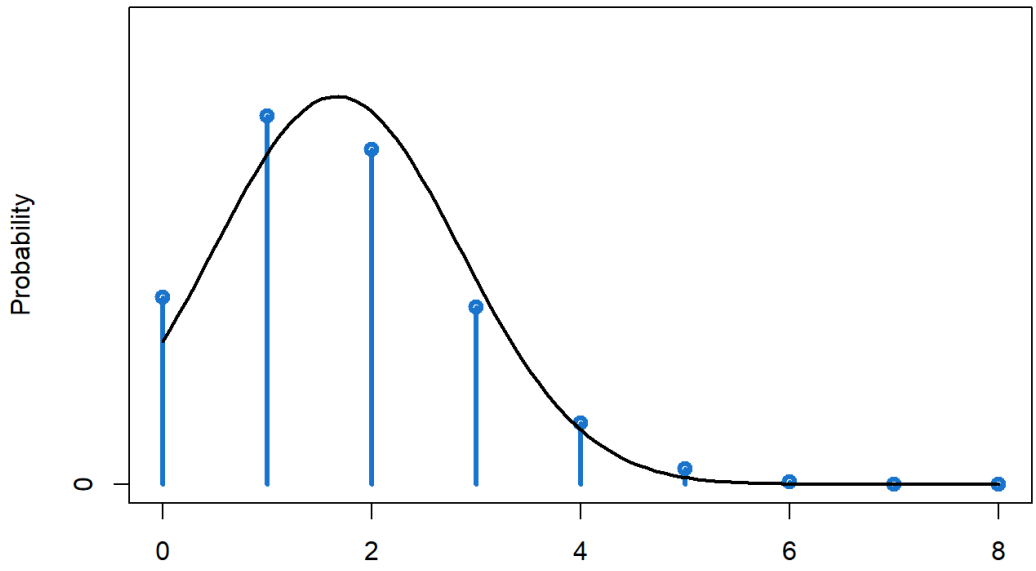


Simulate the number of 6's on 10 rollings of a dice.

```
> n <- 10; p <- 1/6
> x <- rbinom(10^5, n, p)
> barplot(table(x) / length(x),
+         main = "Number of 6's in 10 rollings of a
dice",
+         col = c("#1874CD"))
```

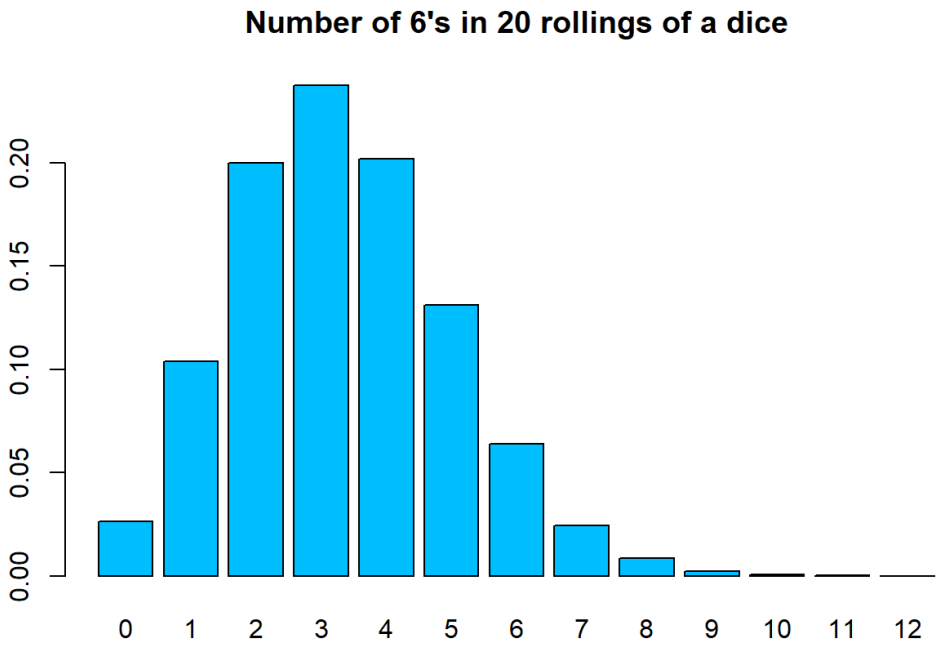



```
> plot(0:(length(table(x)) - 1), table(x) / length(x),
+      xlab = "", ylab = "Probability", ylim = c(0, 0.4),
+      type = "h", lwd = 3, col = c("#1874CD"))
> points(0:(length(table(x)) - 1), table(x) / length(x),
+        type = "p", lwd = 3, col = c("#1874CD"))
> curve(dnorm(x, n * 1 / 6, sqrt(n * 5 / 36)),
+       add = TRUE, col = "black", lwd = 2)
```

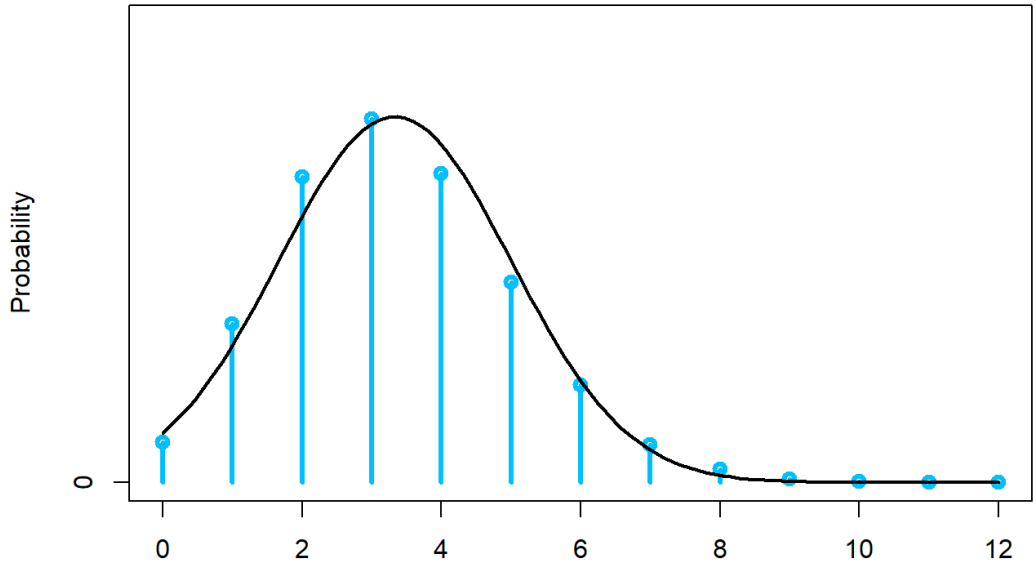


Simulate the number of 6's on 20 rollings of a dice.

```
> n <- 20; p <- 1/6
> x <- rbinom(10^5, n, p)
> barplot(table(x) / length(x),
+         main = "Number of 6's in 20 rollings of a
dice",
+         col = c("#00BFFF"))
```

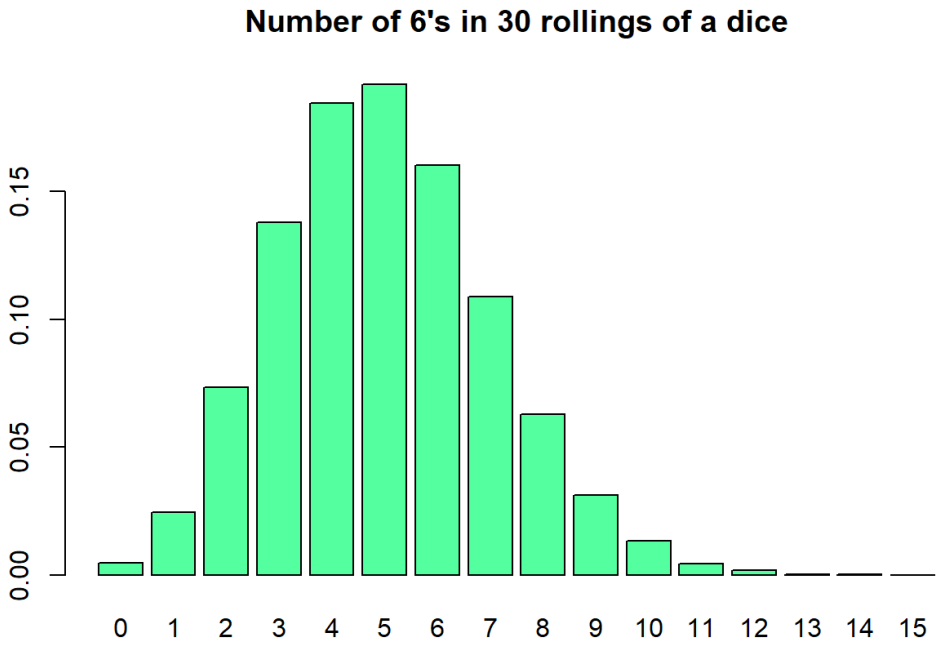



```
> plot(0:(length(table(x)) - 1), table(x) / length(x),
+     xlab = "", ylab = "Probability", ylim = c(0, 0.3),
+     type = "h", lwd = 3, col = c("#00BFFF"))
> points(0:(length(table(x)) - 1), table(x) / length(x),
+        type = "p", lwd = 3, col = c("#00BFFF"))
> curve(dnorm(x, n * 1 / 6, sqrt(n * 5 / 36)),
+       add = TRUE, col = "black", lwd = 2)
```

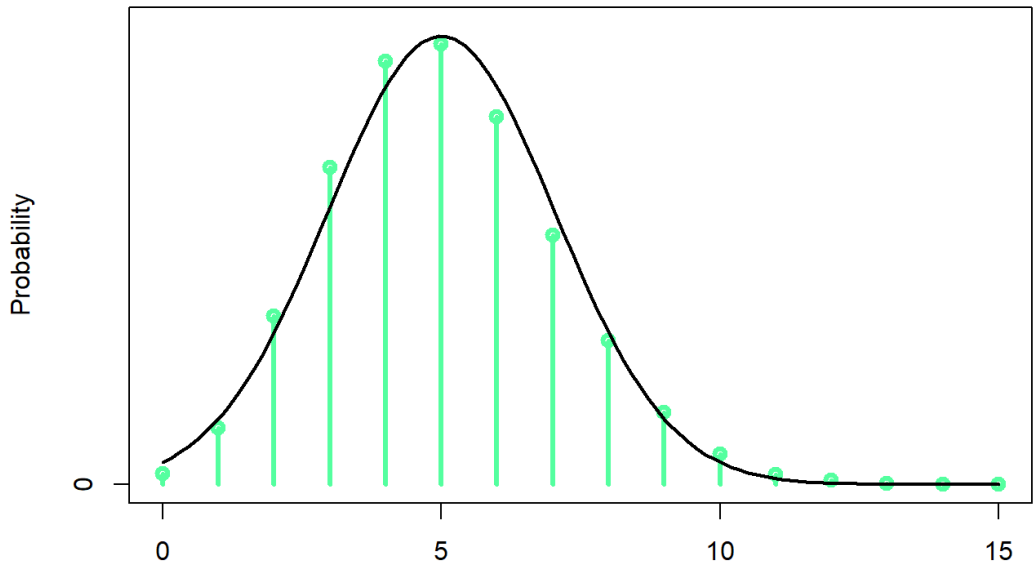


Simulate the number of 6's on 30 rollings of a dice.

```
> n <- 30; p <- 1/6
> x <- rbinom(10^5, n, p)
> barplot(table(x) / length(x),
+         main = "Number of 6's in 30 rollings of a
dice",
+         col = c("#54FF9F"))
```

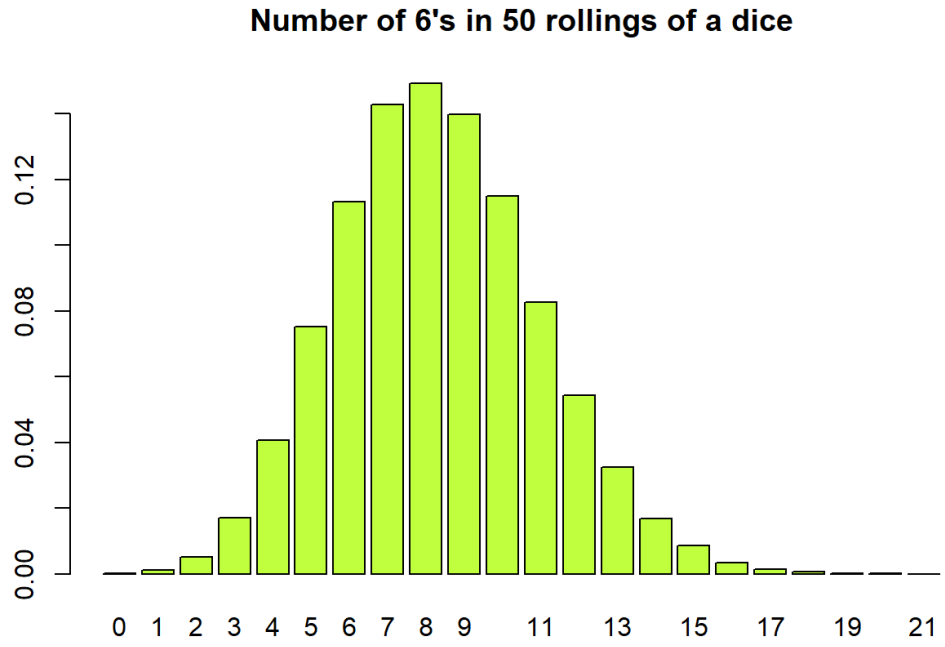



```
> plot(0:(length(table(x)) - 1), table(x) / length(x),
+      xlab = "", ylab = "Probability", ylim = c(0, 0.2),
+      type = "h", lwd = 3, col = c("#54FF9F"))
> points(0:(length(table(x)) - 1), table(x) / length(x),
+        type = "p", lwd = 3, col = c("#54FF9F"))
> curve(dnorm(x, n * 1 / 6, sqrt(n * 5 / 36)),
+       add = TRUE, col = "black", lwd = 2)
```

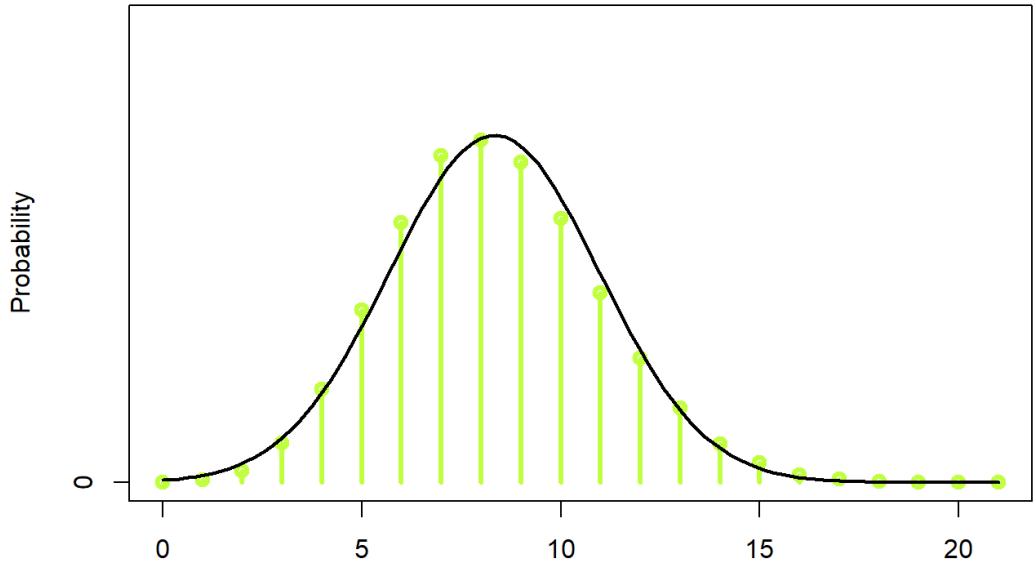


Simulate the number of 6's on 50 rollings of a dice.

```
> n <- 50; p <- 1/6
> x <- rbinom(10^5, n, p)
> barplot(table(x) / length(x),
+         main = "Number of 6's in 50 rollings of a
dice",
+         col = c("#C0FF3E"))
```




```
> plot(0:(length(table(x)) - 1), table(x) / length(x),
+     xlab = "", ylab = "Probability", ylim = c(0, 0.2),
+     type = "h", lwd = 3, col = c("#C0FF3E"))
> points(0:(length(table(x)) - 1), table(x) / length(x),
+        type = "p", lwd = 3, col = c("#C0FF3E"))
> curve(dnorm(x, n * 1 / 6, sqrt(n * 5 / 36)),
+       add = TRUE, col = "black", lwd = 2)
```



Example - Normal

When $X_i \in N(\mu_i, \sigma^2)$, $i = 1, \ldots, n$ and $X_1, \ldots, X_n$ are mutually independent then

$$S_n = X_1 + \ldots + X_n \in N(\mu_1 + \ldots + \mu_n, \sigma_1^2 + \ldots + \sigma_n^2)$$

When additionally X_i, $i = 1, \ldots, n$ are identically distributed

$$\mu := \mu_1 = \mu_2 = \ldots = \mu_n$$
$$\sigma := \sigma_1 = \sigma_2 = \ldots = \sigma_n$$

then

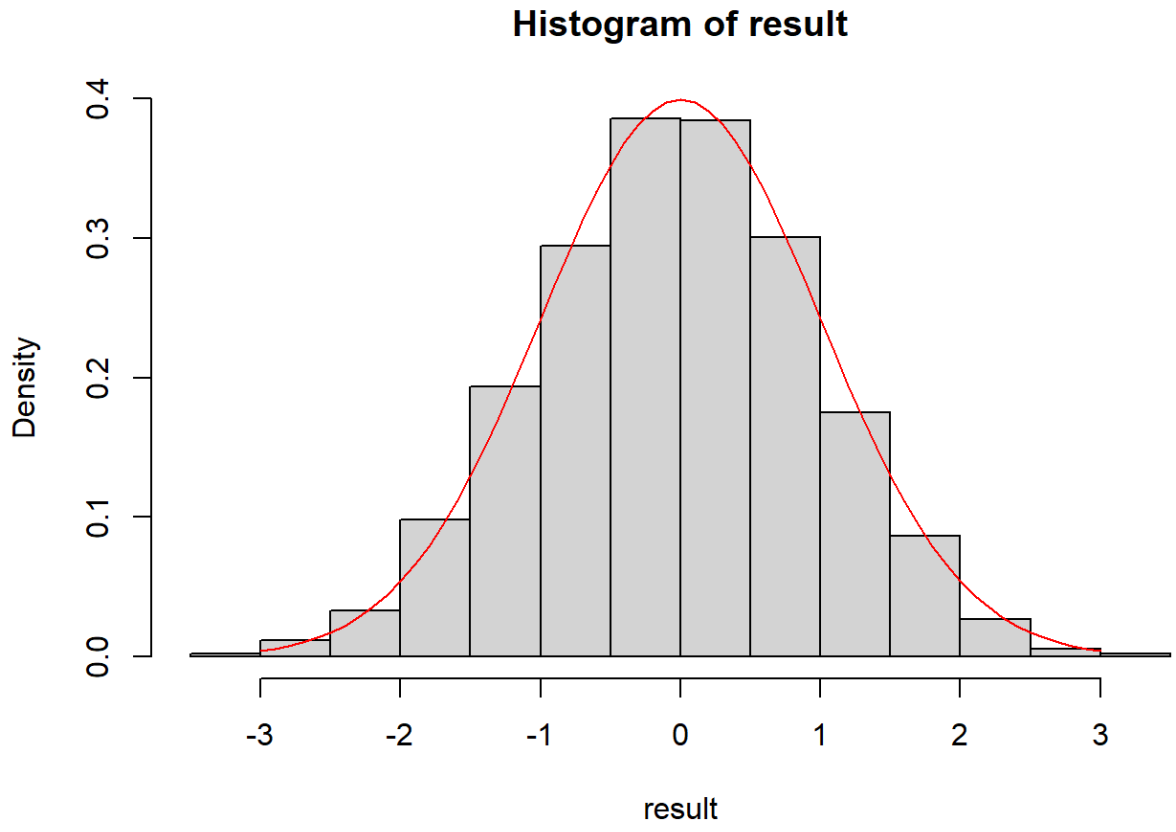
$$S_n = X_1 + \ldots + X_n \in N(n\mu, n\sigma^2)$$
$$\frac{S_n}{n} \in N\left(\mu, \frac{\sigma^2}{n}\right)$$

$$\frac{\frac{S_n}{n} - \mu}{\frac{\sigma}{\sqrt{n}}} \in N(0,1)$$

Simulate 5000 times sum of $n = 3$ normal variables $N(2, 5^2)$. Plot the histogram of the normalized average and compare it with the standard normal $N(0,1)$ probability density function.

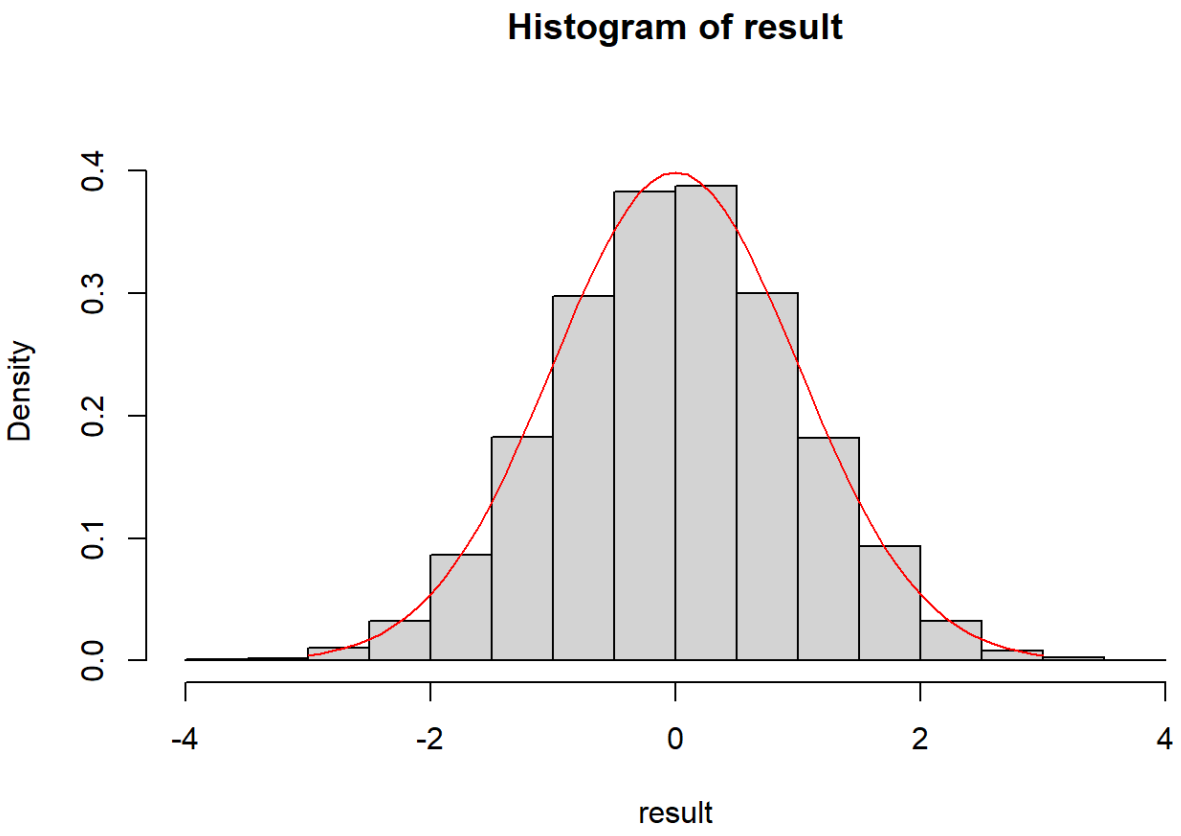
A solution can be to use a `for` loop.

```
> n <- 3; mu <- 2; sigma <- 5
> result <- 0
> for(i in 1:5000){
+   X <- rnorm(n, mu, sigma)
+   result[i] = (mean(X) - mu) / (sigma / sqrt(n))
+ }
> hist(result, probability = TRUE)
> x <- seq(from = -3, to = 3, by = 0.1)
> y <- dnorm(x, mean = 0, sd = 1)
> lines(x, y, col = "red", type = "l")
```



Or we can wright a function which generate the results.

```
> f <- function (nSim, n, mu = 0, sigma = 1){
+     result <- 0
+     for(i in 1:nSim){
+         x <- rnorm(n, mu, sigma)
+         result[i] <- (mean(x) - mu) / (sigma / sqrt(n))
+     }
+     return(result)
+ }
> result <- f(5000, n, mu, sigma)
> hist(result, probability = TRUE, ylim = c(0, 0.44))
> lines(x, y, col = "red", type = "l")
```



Or we can use `simple.sim` from the `UsingR` package to generate observations form a function.

```
> library("UsingR")
Warning: package 'UsingR' was built under R version 4.0.3
Loading required package: MASS
Loading required package: HistData
Loading required package: Hmisc
```

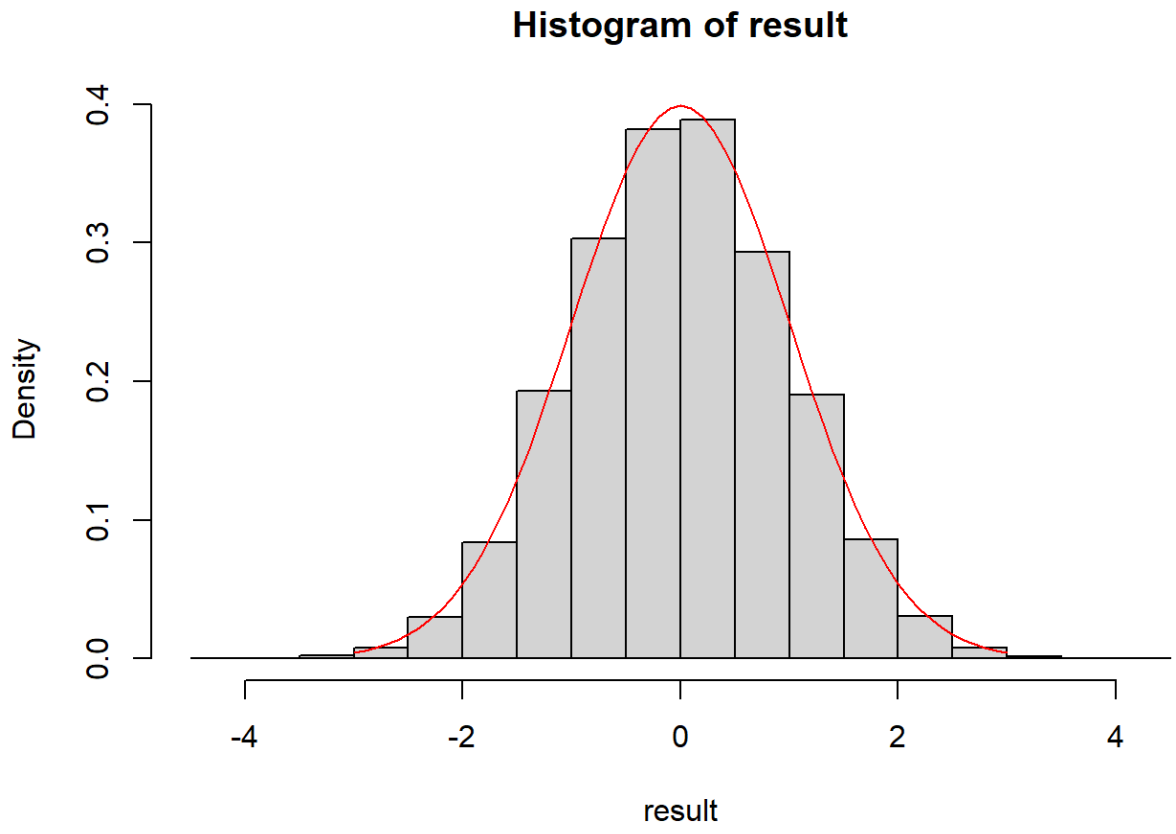
```
Loading required package: lattice
Loading required package: survival
Loading required package: Formula
Loading required package: ggplot2

Attaching package: 'Hmisc'
The following objects are masked from 'package:base':

    format.pval, units

Attaching package: 'UsingR'
The following object is masked from 'package:survival':

    cancer
> f <- function(n, mu, sigma){
+     x <- rnorm(n, mu, sigma)
+     (mean(x) - mu) / (sigma / sqrt(n))
+ }
> result <- simple.sim(5000, f, n, mu, sigma)
> hist(result, probability = TRUE)
> lines(x, y, col = "red", type = "l")
```

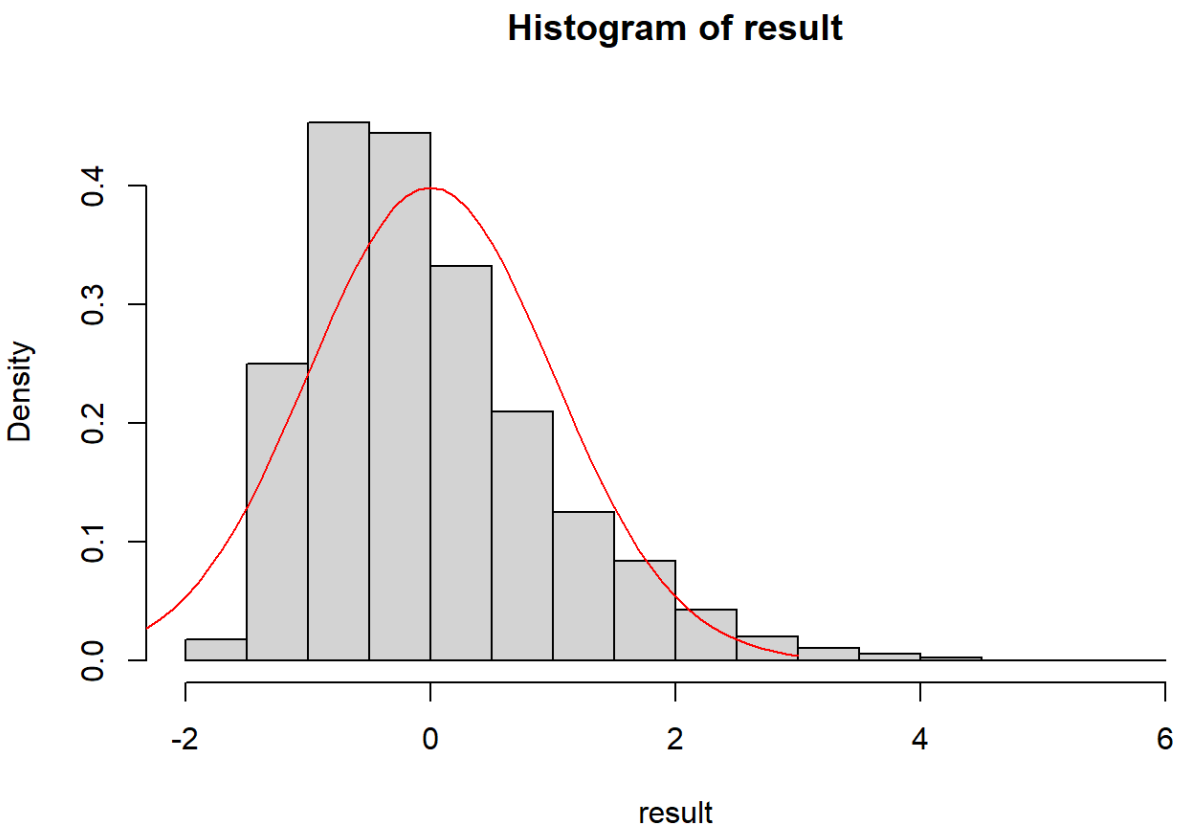


Example - Exponential

Simulate $5\ 000$ times sum of $n = 10$ exponential variables $Exp(0.5)$. Plot the histogram of the normalized average and compare it with the standard normal $N(0,1)$ probability density function.

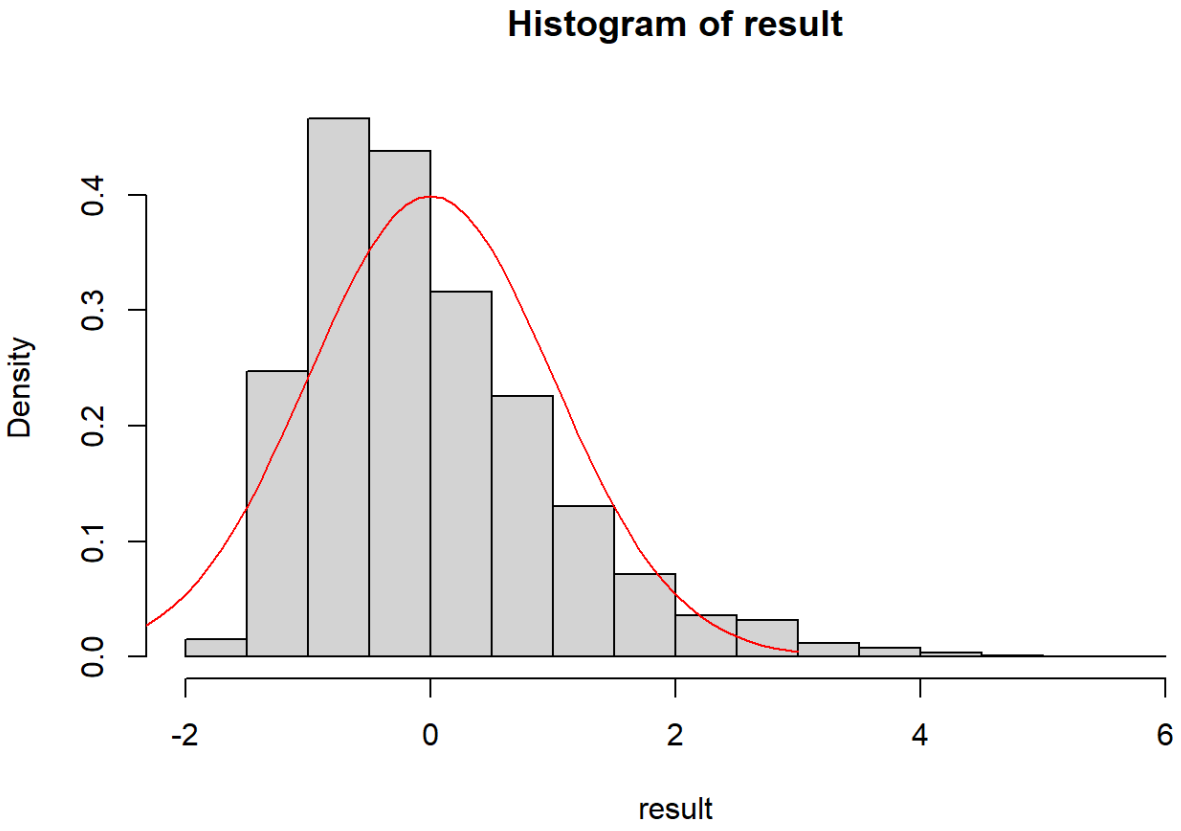
A solution can be to use a for loop.

```
> lambda <- 0.5
> nSum <- 3; mu <- 1/lambda; sigma <- 1/lambda
> result <- 0
> for(i in 1:5000){
+     x <- rexp(nSum, lambda)
+     result[i] <- (mean(x) - mu) / (sigma / sqrt(nSum))
+ }
> hist(result, probability = TRUE)
> x <- seq(from = -3, to = 3, by = 0.1)
> y <- dnorm(x, mean = 0, sd = 1)
> lines(x, y, col = "red", type = "l")
```



Or we can use `simple.sim` function.

```
> f <- function(nSum, lambda){
+     x <- rexp(nSum, lambda)
+     mu <- 1/lambda; sigma <- 1/lambda
+     (mean(x) - mu) / (sigma / sqrt(nSum))
+ }
> result <- simple.sim(5000, f, nSum, lambda)
> hist(result, probability = TRUE)
> lines(x, y, col = "red", type = "l")
```

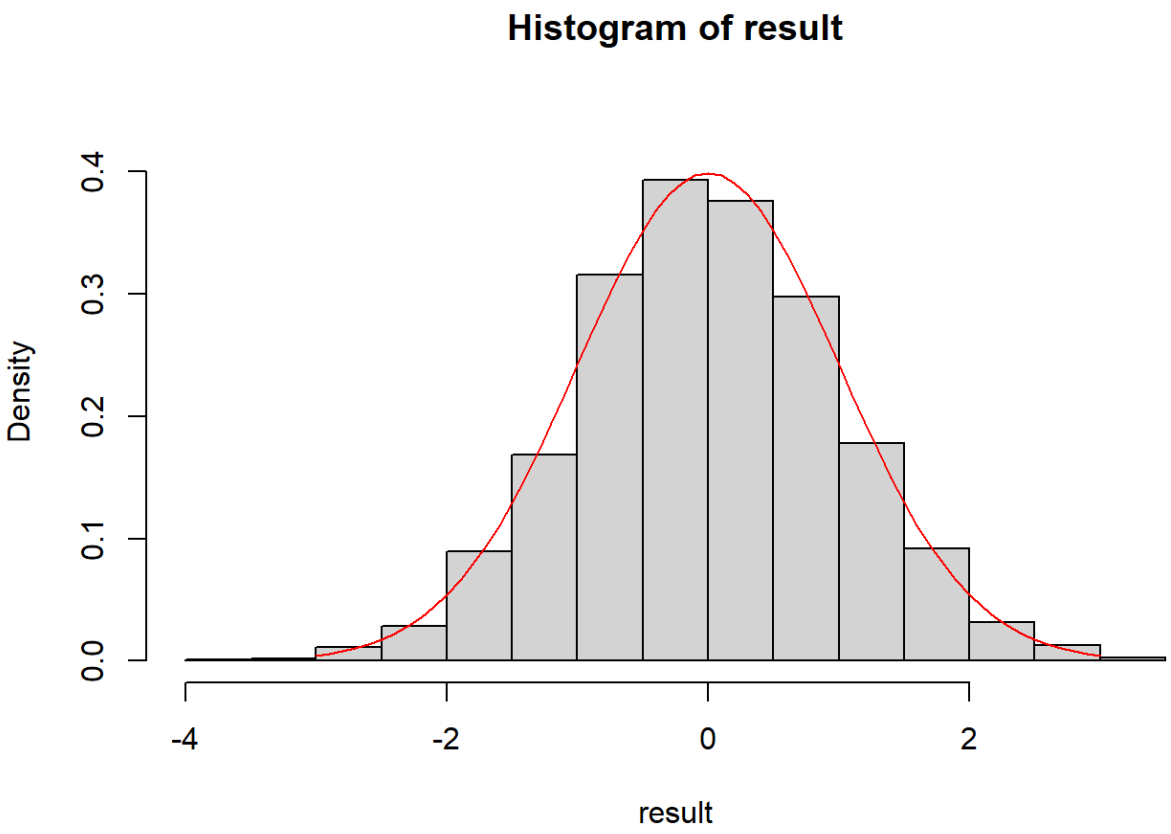


Make analogous simulations and plot for $n = 100\ 000$.

A solution can be to use a for loop.

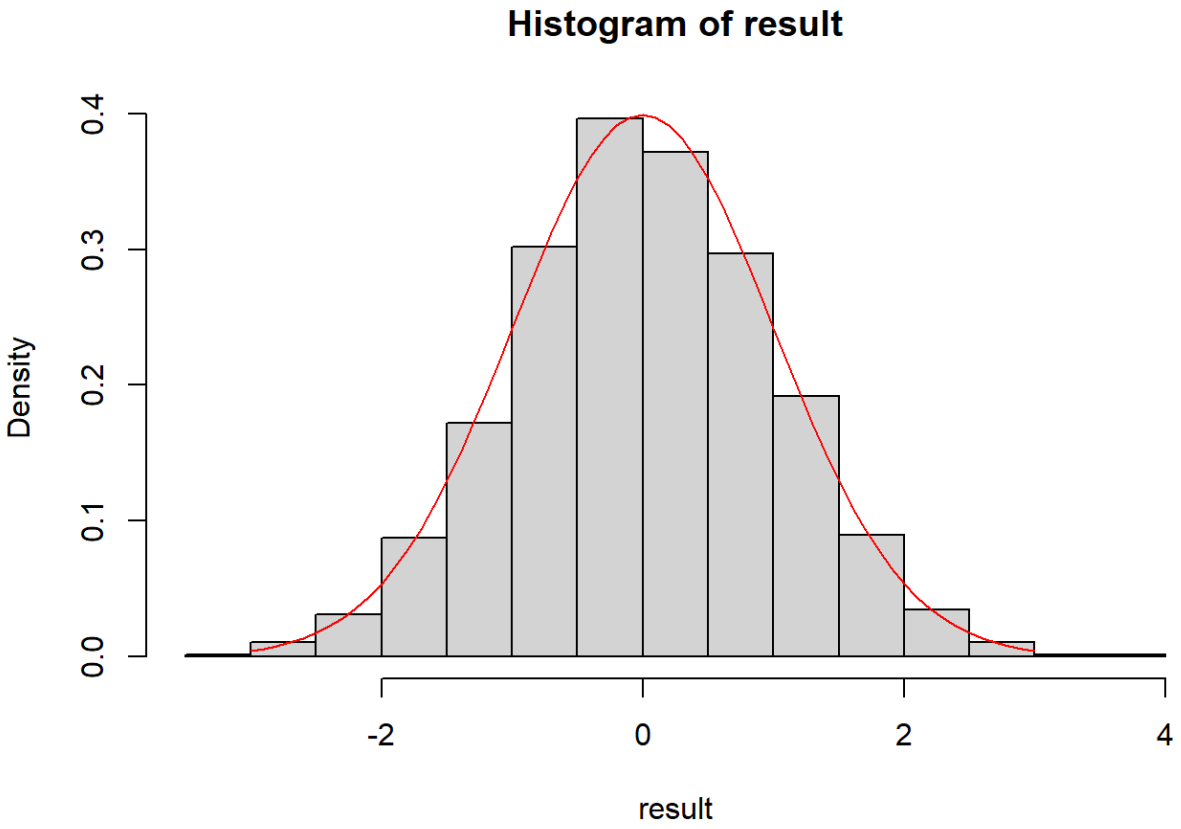
```
> lambda <- 0.5
> nSum <- 10^5; mu <- 1/lambda; sigma <- 1/lambda
> result <- 0
> for(i in 1:5000){
+     x <- rexp(nSum, lambda)
+     result[i] <- (mean(x) - mu) / (sigma / sqrt(nSum))
+ }
```

```
> hist(result, probability = TRUE, ylim = c(0, 0.44))
> x <- seq(from = -3, to = 3, by = 0.1)
> y <- dnorm(x, mean = 0, sd = 1)
> lines(x, y, col = "red", type = "l")
```



Or we can use `simple.sim` function.

```
> f <- function(nSum, lambda){
+     x <- rexp(nSum, lambda)
+     mu <- 1/lambda; sigma <- 1/lambda
+     (mean(x) - mu) / (sigma / sqrt(nSum))
+ }
> result <- simple.sim(5000, f, nSum, lambda)
> hist(result, probability = TRUE)
> lines(x, y, col = "red", type = "l")
```



Is the observed random variable normally distributed?

Let's denote by $F_n(x)$ the empirical cumulative distribution function of the sample and by $F_n(x)$ the cumulative distribution function of the observed random variable.

Glivenko-Cantelli theorem

Glivenko-Cantelli theorem states that when the sample size increases the empirical cumulative distribution function tends uniformly to the theoretical distribution of the observed random variable.

$$\sup_{x \in \mathbb{R}} | F_n(x) - F_X(x) | \underset{n \to \infty}{\longrightarrow} 0$$

pp-plot (probability-probability plot)

pp-plot contains the dots with coordinates

$$\left(F_n(x_{(i)}),\, F_T(x_{(i)})\right),\, i = 1,\, \ldots,\, n$$

where F_T is the tested cumulative distribution function.

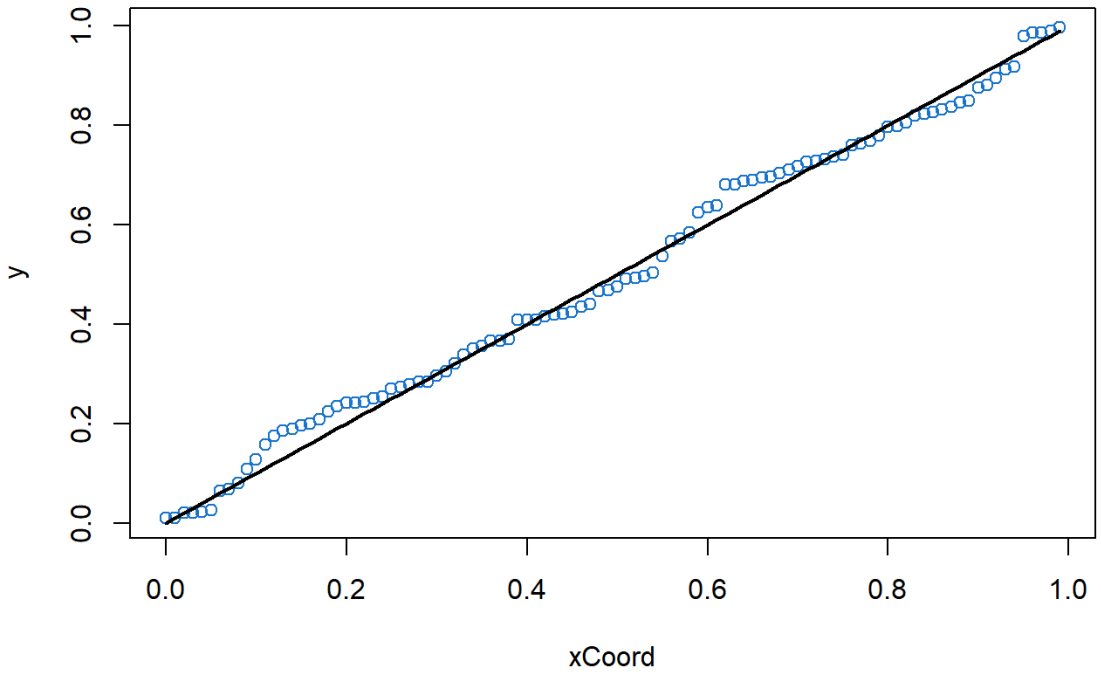
If the distribution is continuous $F_n\left(x_{(i)}\right) = \frac{i}{n}$ and pp-plot can be described as the set of points with coordinates

$$\left(\frac{i}{n},\, F_T(x_{(i)})\right),\, i = 1,\, \ldots,\, n$$

If the tested distribution coincides with the real distribution of the observed random variable X then these points will lie over the bisectrix of the first quadrant.

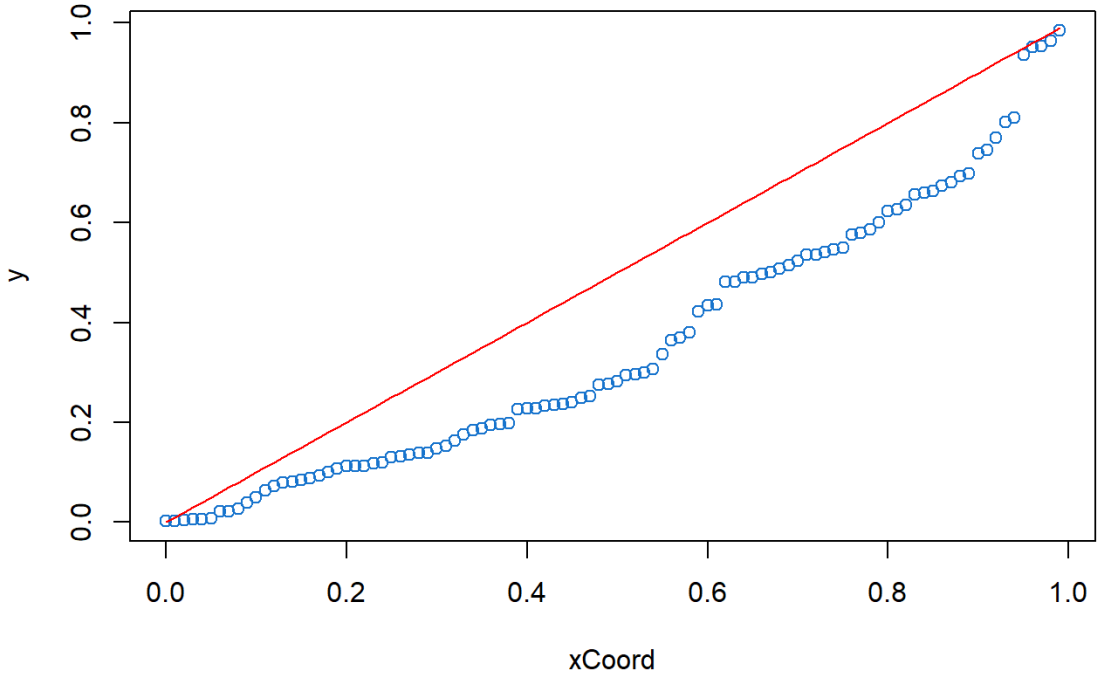
It can be plotted the following way.

```
> x <- rnorm(100, 3, 4)
> xCoord <- 0:99 / 100
> y <- pnorm(sort(x), mean(x), sd(x))
> plot(xCoord, y, col = "#1874CD")
> lines(xCoord, xCoord, lwd = 2)
```



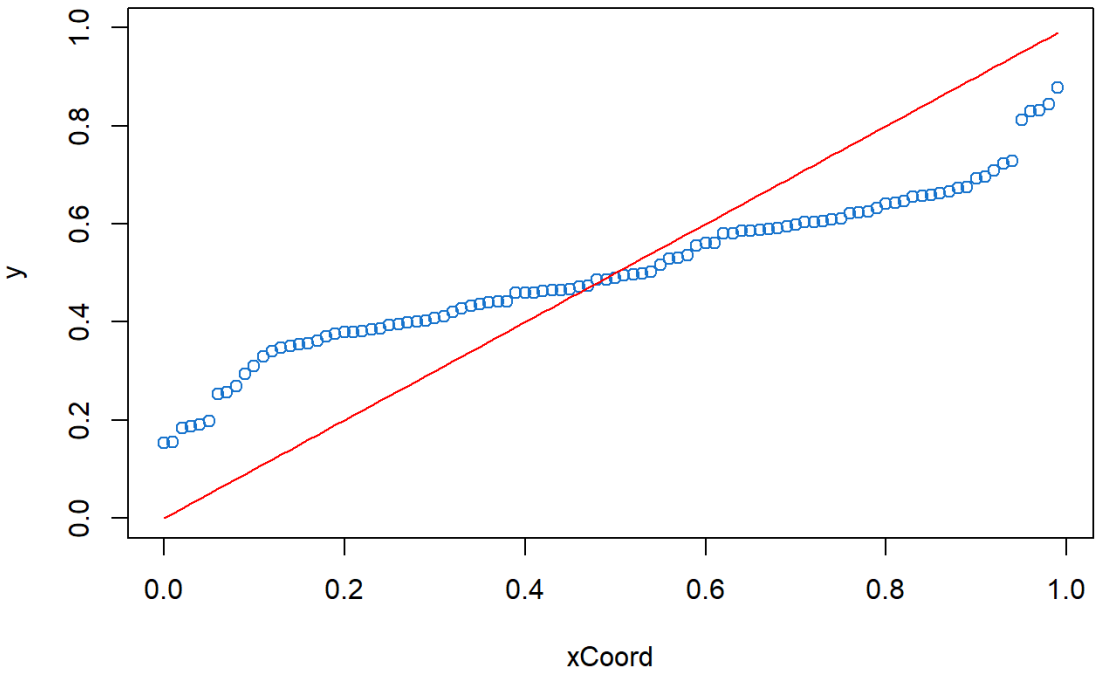
Let's observe what will happen if we write wrong mean.

```
> y <- pnorm(sort(x), mean(x) + 2, sd(x))
> plot(xCoord, y, col = "#1874CD")
> lines(xCoord, xCoord, col = "red")
```



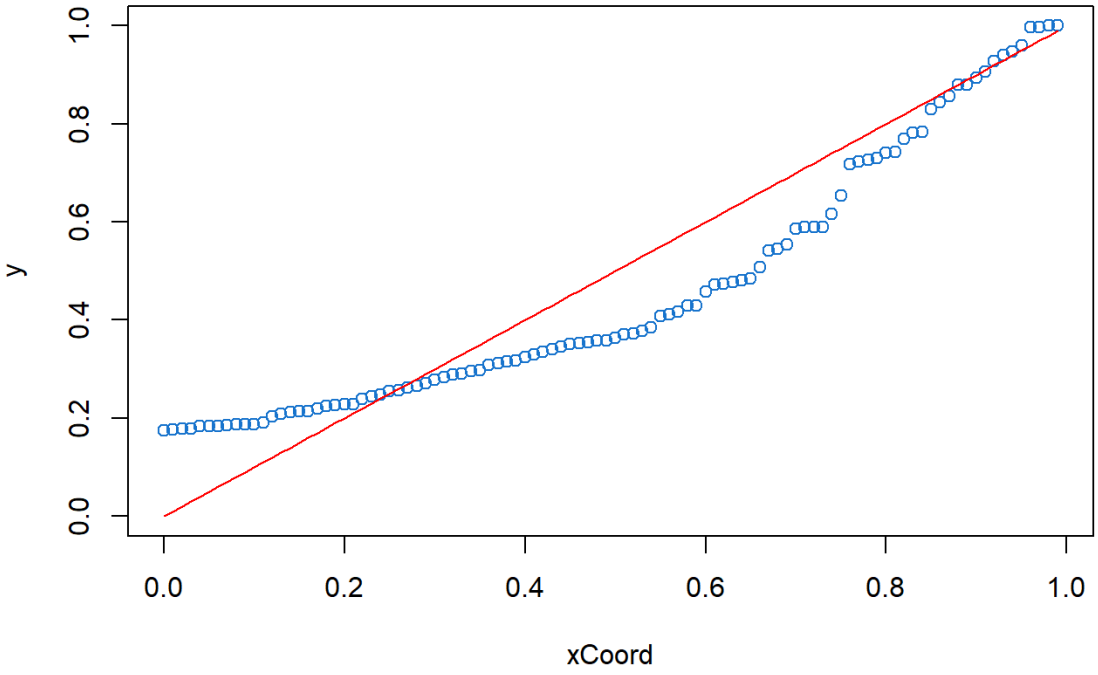
Let's observe what will happen if we write wrong standard deviation.

```
> y <- pnorm(sort(x), mean(x), sd(x) + 5)
> plot(xCoord, y, ylim = c(0,1), col = "#1874CD")
> lines(xCoord, xCoord, col = "red")
```



Let's observe what will happen if we use wrong probability distribution.

```
> x <- rexp(100, 1/3)
> xCoord <- 0:99 / 100
> y <- pnorm(sort(x), mean(x), sd(x))
> plot(xCoord, y, ylim = c(0, 1), col = "#1874CD")
> lines(xCoord, xCoord, col = "red")
```



These movements can be explored easily on the **qq-plot**.

qq-plot (quantile-quantile plot)

qq-plot contains the dots with coordinates

$$\left(F_{\overleftarrow{n}}, \left(\frac{i}{n} \right), F_{\overleftarrow{T}} \left(\frac{i}{n} \right) \right), i = 1, \ldots, n$$

where F_T is the tested cumulative distribution function.

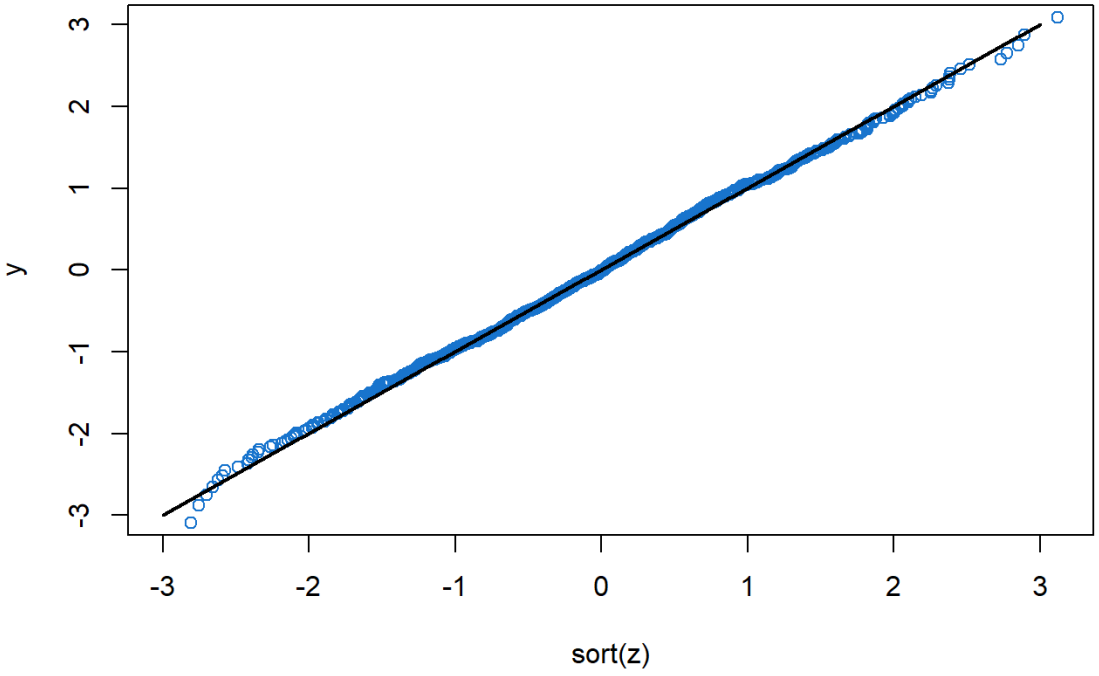
If the distribution is continuous $F_{\overleftarrow{T}} \left(\frac{i}{n} \right) = x(i)$ the qq-plot contains the set of points with coordinates

$$\left(x_{(i)}, F_{\overleftarrow{T}} \left(\frac{i}{n} \right) \right), i = 1, \ldots, n$$

If the tested distribution coincides with the real distribution of the observed random variable X then these points will lie over the bisectrix of the first and the third quadrant.

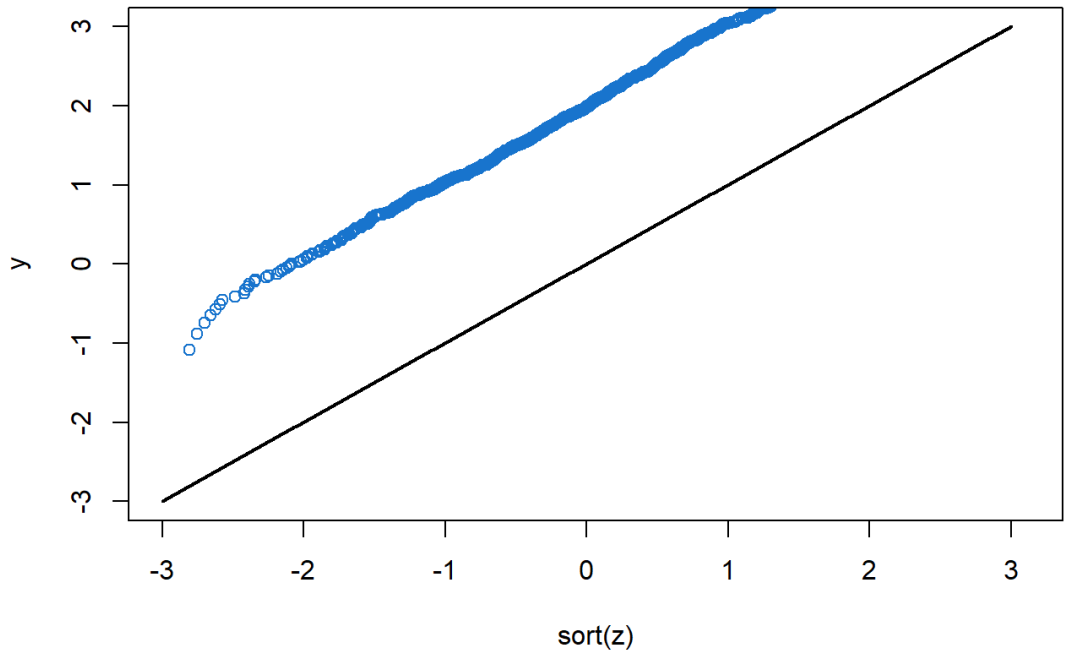
It can be plotted the following way.

```
> x <- rnorm(1000, 3, 4)
> z <- (x - 3) / 4
> y <- qnorm(p = 0:999 / 1000, mean = 0, sd = 1)
> plot(sort(z), y, ylim = c(-3, 3), col = "#1874CD")
> xCoord <- seq(from = -3, to = 3, length.out = 1000)
> lines(xCoord, xCoord, lwd = 2)
```



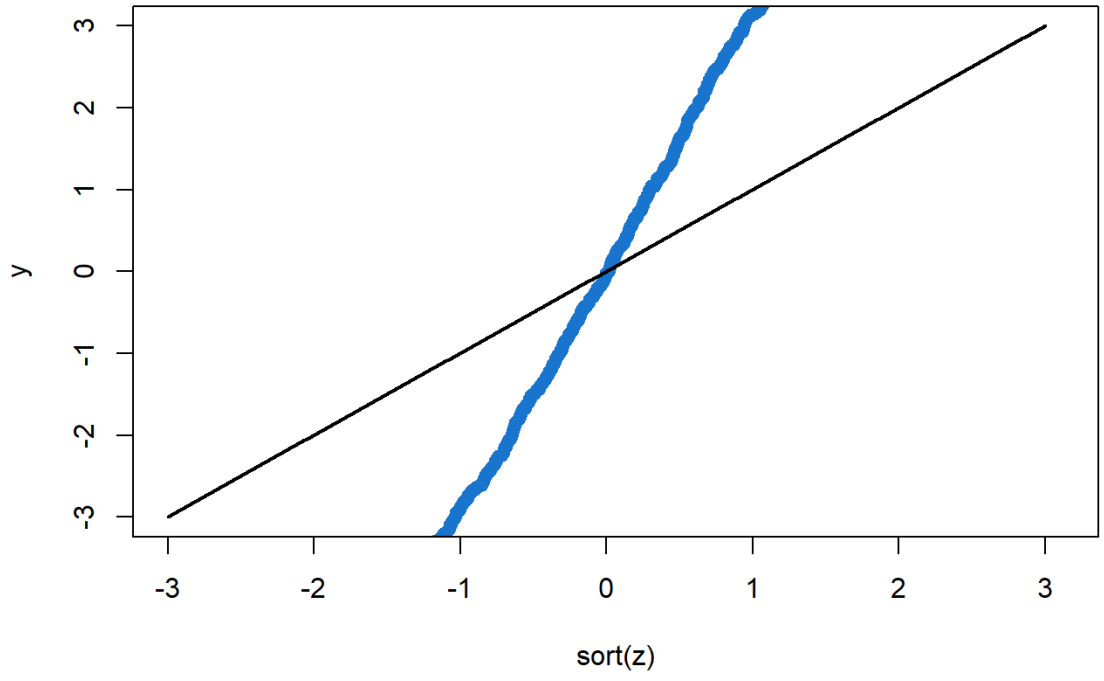
Let's observe what will happen if we write wrong mean.

```
> y <- qnorm(p = 0:999 / 1000, mean = 0 + 2, sd = 1)
> plot(sort(z), y, ylim = c(-3, 3), col = "#1874CD")
> lines(xCoord, xCoord, lwd = 2)
```



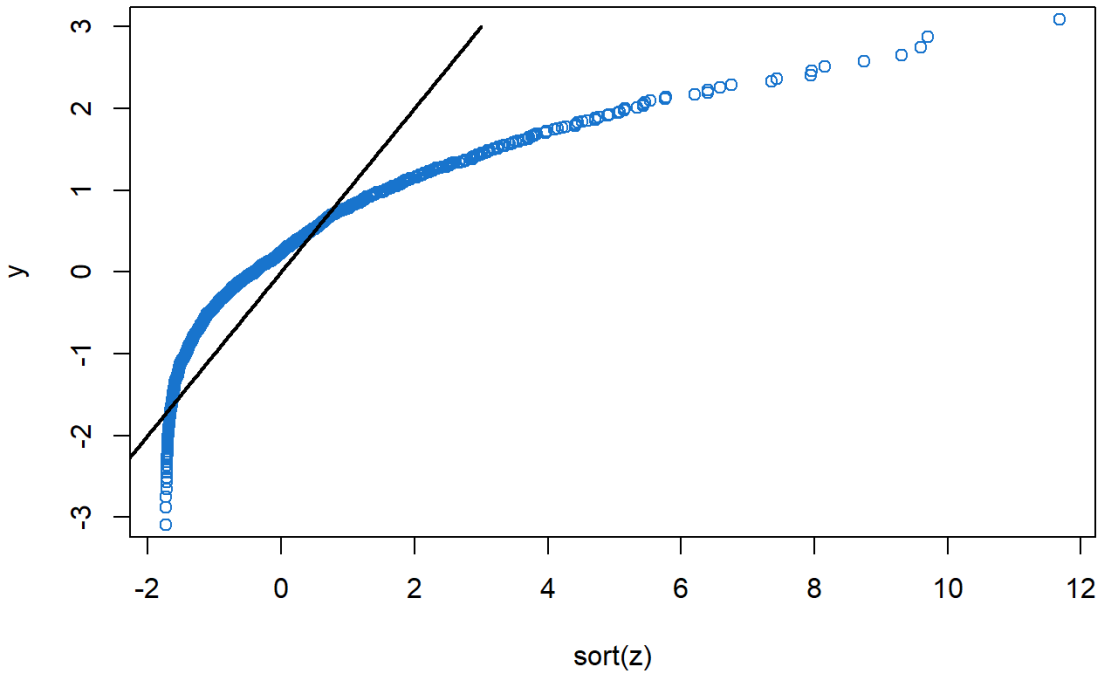
Let's observe what will happen if we write wrong standard deviation.

```
> y <- qnorm(p = 0:999 / 1000, mean = 0, sd = 1 + 2)
> plot(sort(z), y, ylim = c(-3, 3), col = "#1874CD")
> lines(xCoord, xCoord, lwd = 2)
```



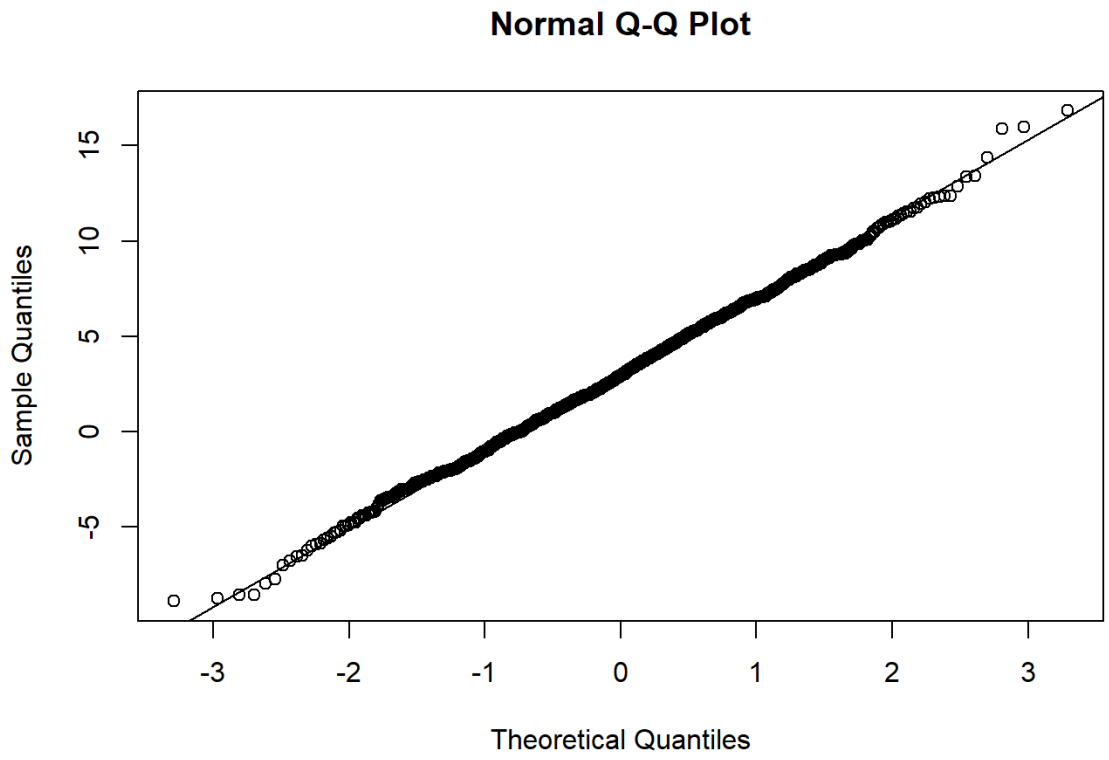
Let's observe what will happen if we use wrong probability distribution.

```
> x <- rexp(1000, 1/3)
> z <- (x - 3) / sqrt(3)
> y <- qnorm(p = 0:999 / 1000, mean = 0, sd = 1)
> plot(sort(z), y, ylim = c(-3, 3), col = "#1874CD")
> xCoord <- seq(from = -3, to = 3, length.out = 1000)
> lines(xCoord, xCoord, lwd = 2)
```

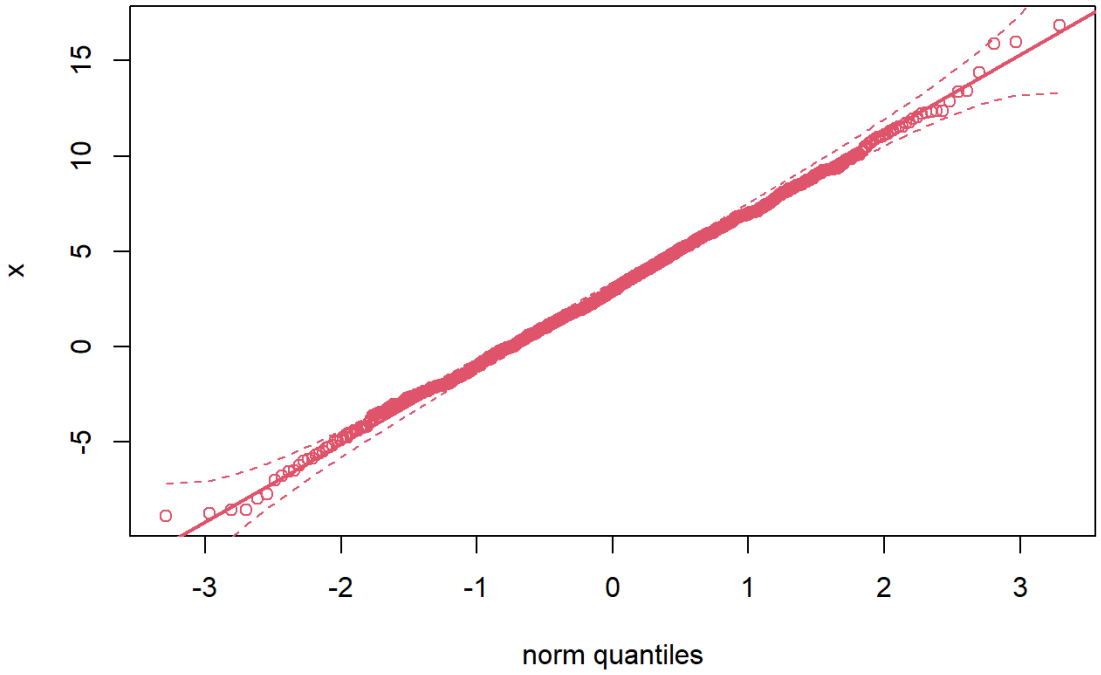


You can also use the `qqnorm` and `qqline` functions to test if the data is normally distributed or `qqplot.das` from `StatDA` package to test with other distributions.

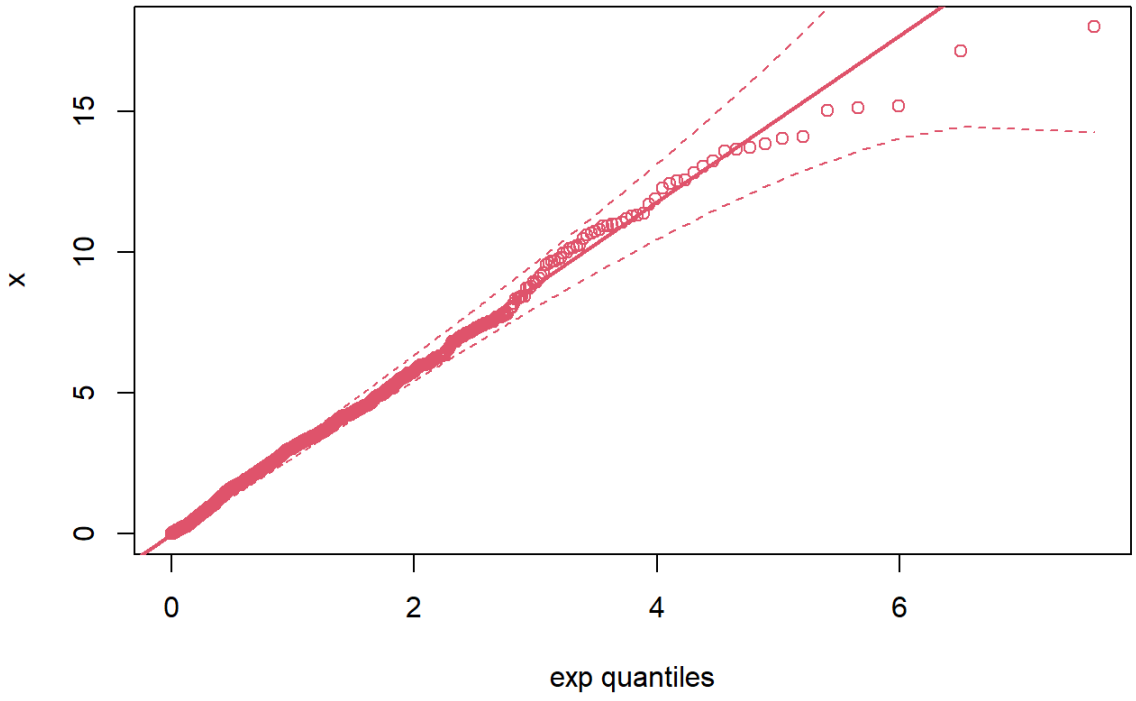
```
> x <- rnorm(1000, 3, 4)
> qqnorm(x)
> qqline(x)
```



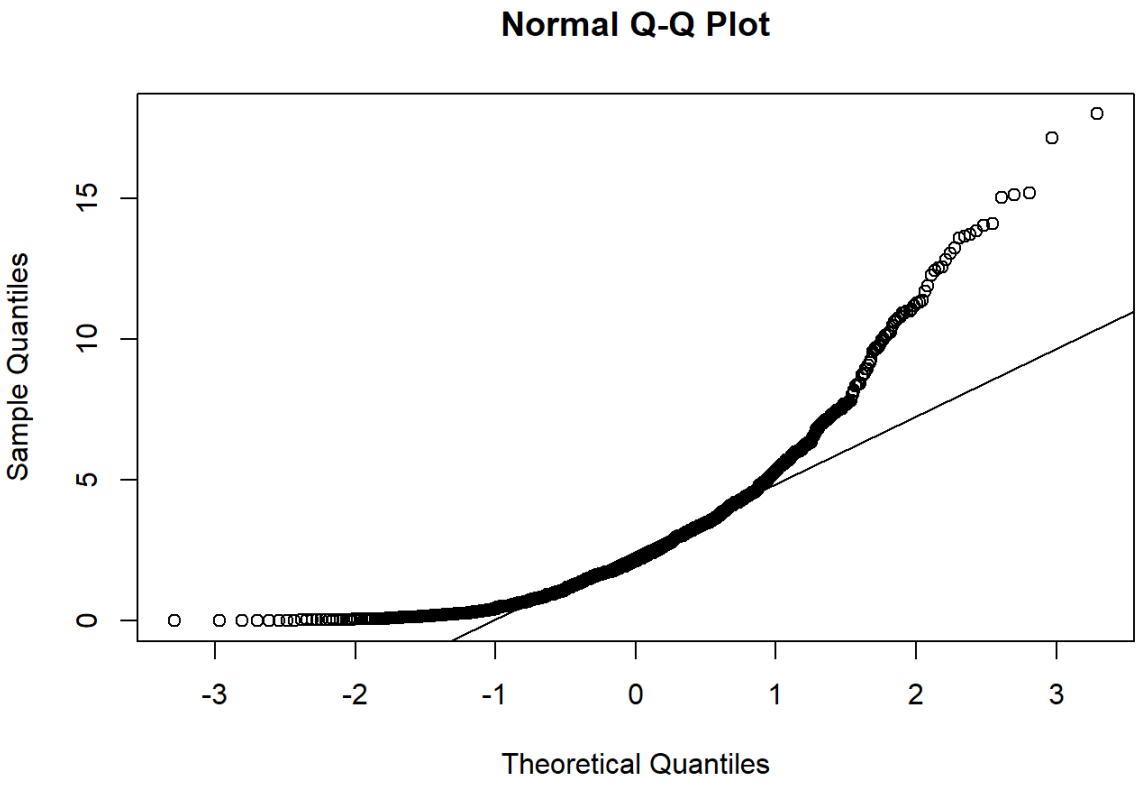

```
> library(StatDA)
Warning: package 'StatDA' was built under R version 4.0.3
Loading required package: sgeostat
Warning: package 'sgeostat' was built under R version
4.0.3
Registered S3 method overwritten by 'geoR':
  method          from
  plot.variogram sgeostat
> qqplot.das(x, "norm")
```



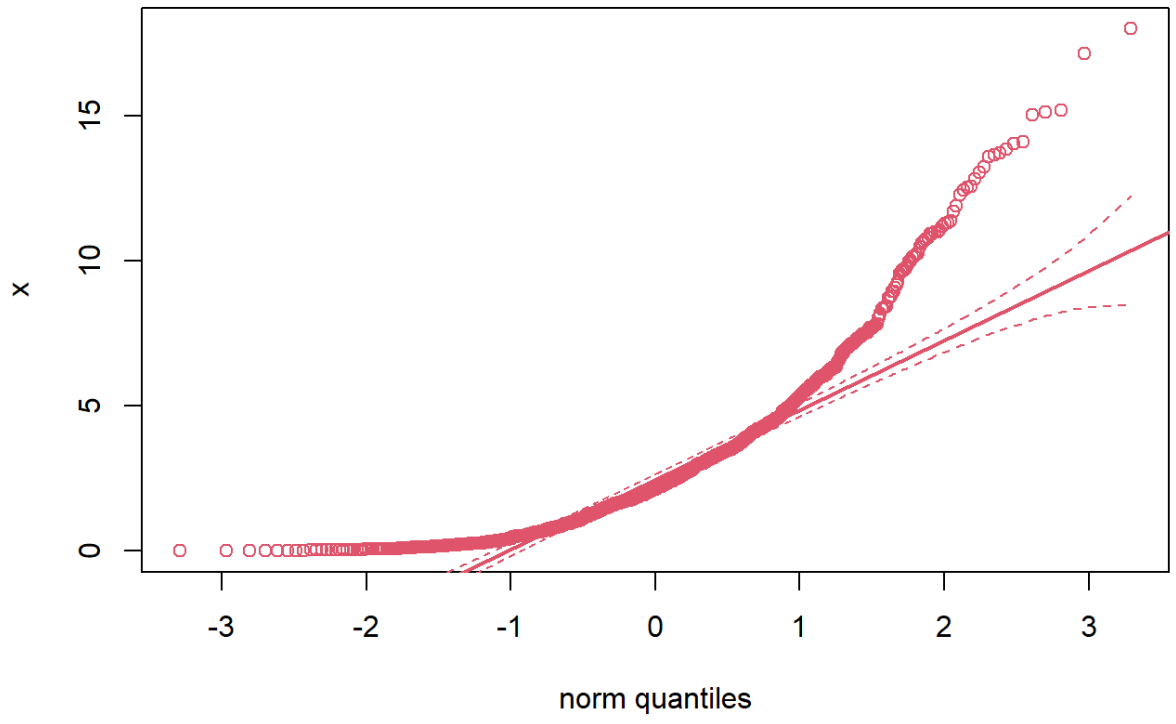

```
> x <- rexp(1000, 1/3)
> qqplot.das(x, "exp")
```



```
> qqnorm(x)
> qqline(x)
```

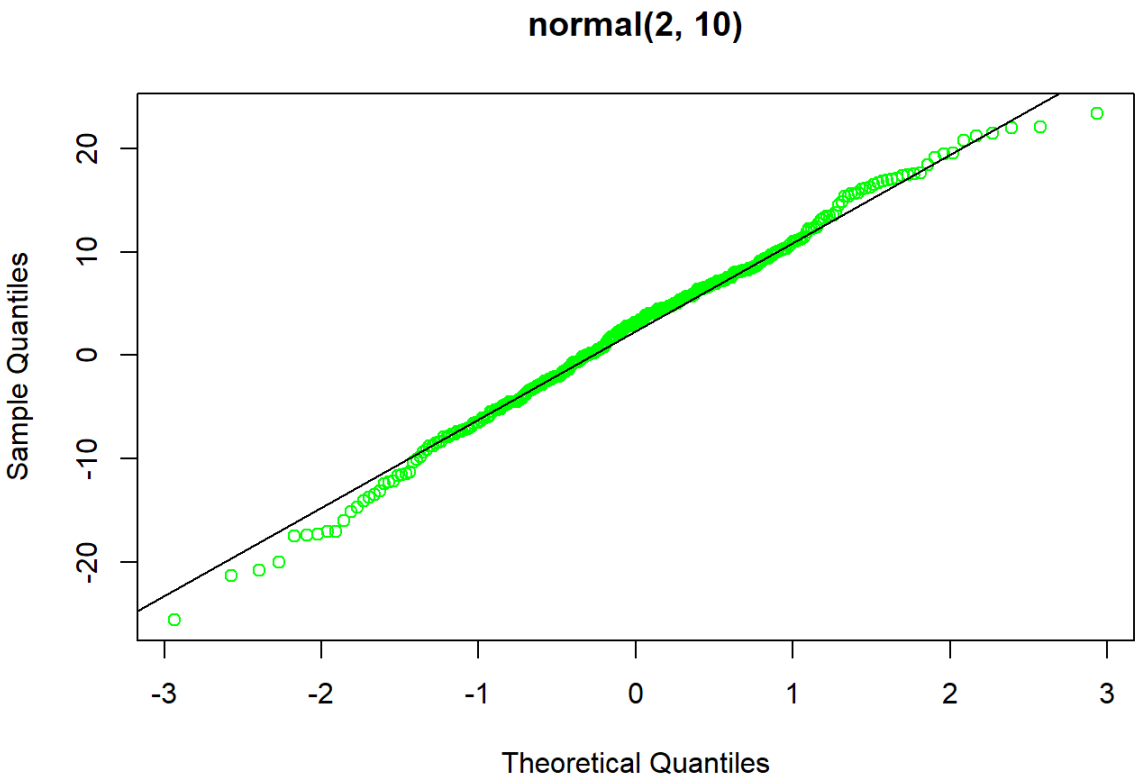



```
> qqplot.das(x, "norm")
```



Example X~N(2, 10^2)

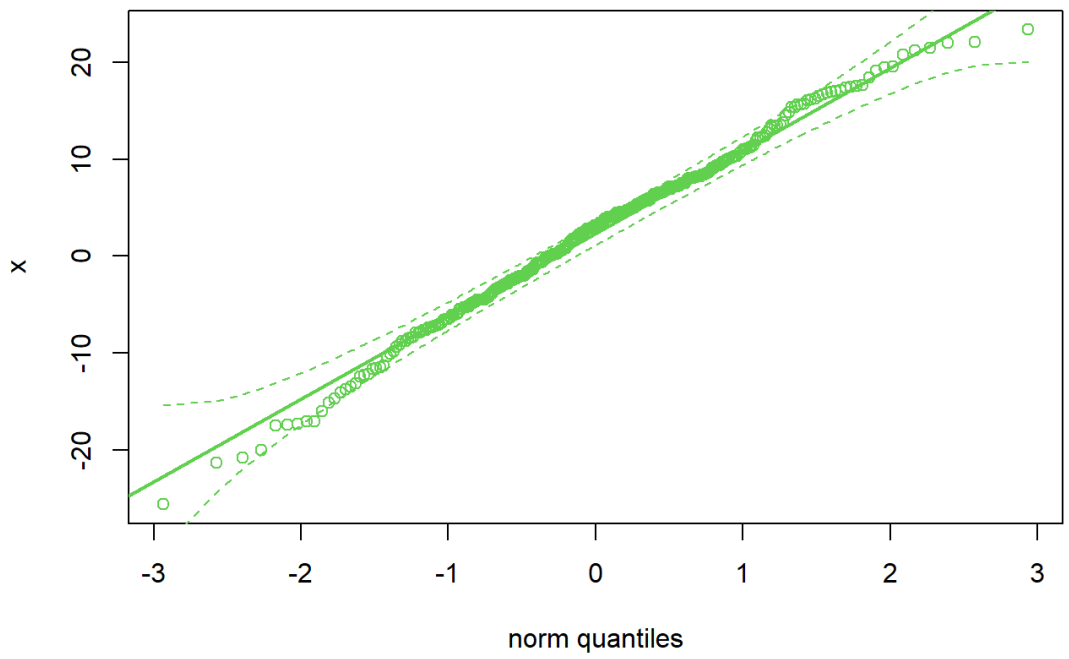
```
> x = rnorm(300, 2, 10)
> qqnorm(x, main = 'normal(2, 10)', col = "green")
> qqline(x)
```



The same example

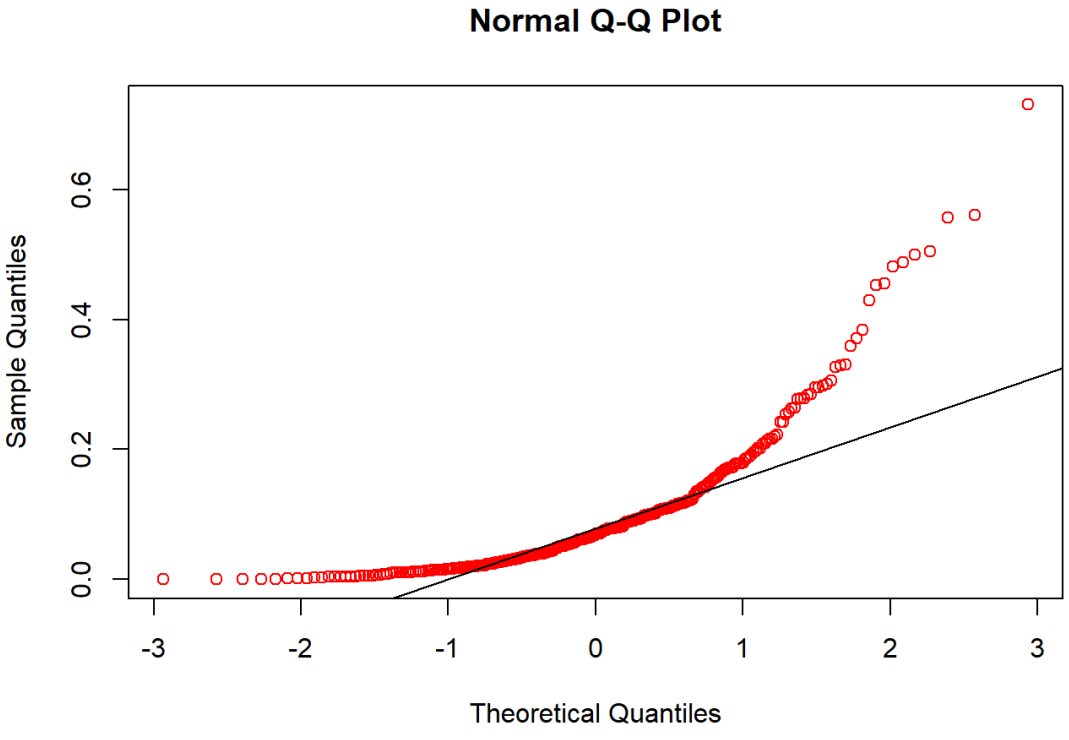
```
> library(StatDA)
```

```
> qqplot.das(x, "norm", col = palette()[3])
```



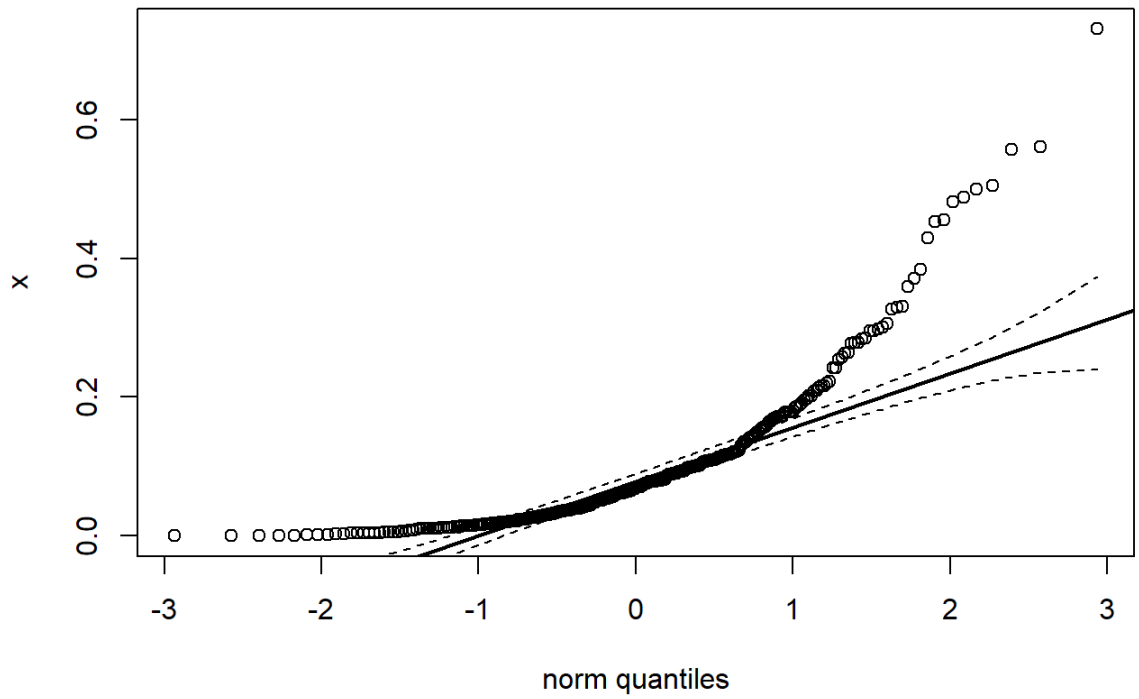
Example X~Exp(10)

```
> x = rexp(300, 10)
> qqnorm(x, col = "red")
> qqline(x)
```

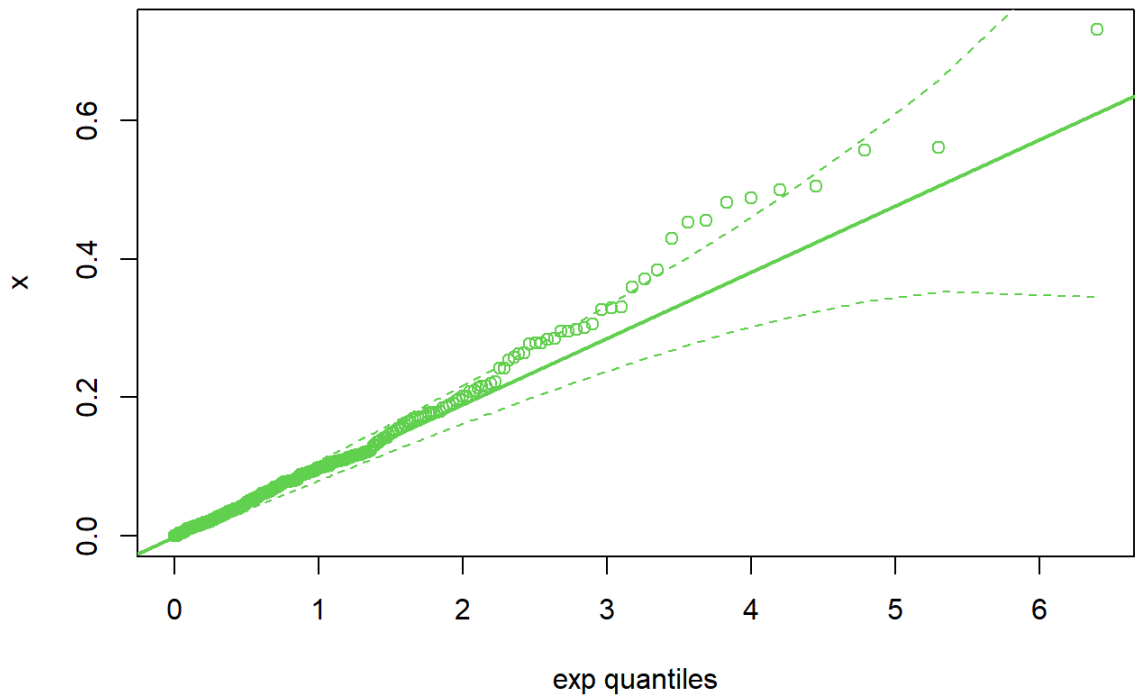


The same example

```
> qqplot.das(x, "norm", col = palette()[1])
```

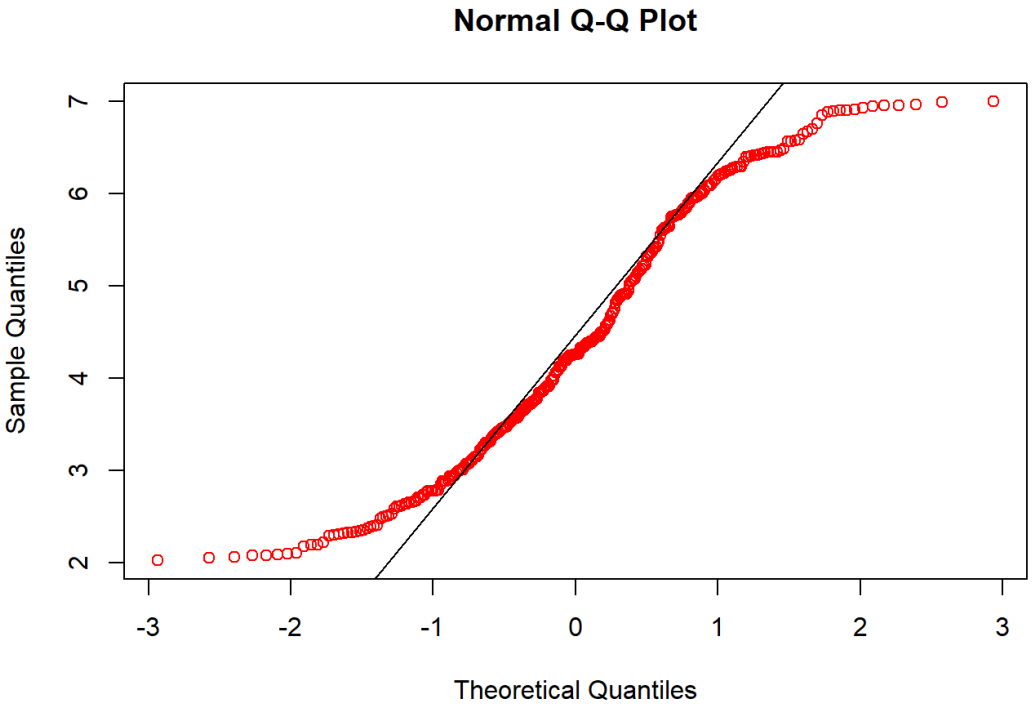



```
> qqplot.das(x, "exp", col = palette()[3])
```



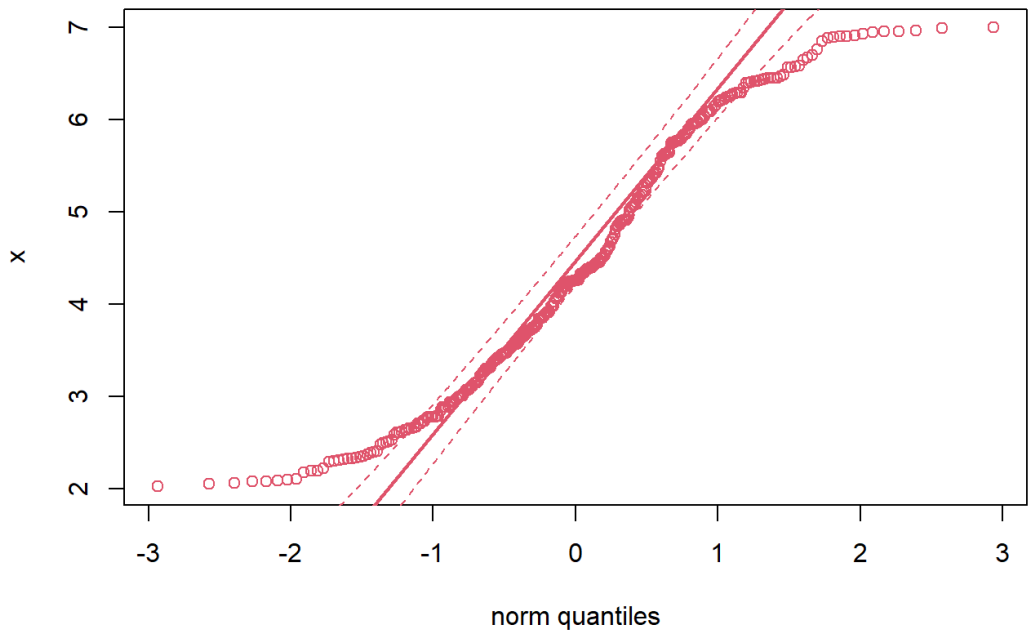
Example X~U(2,7)

```
> x = runif(300, min = 2, max = 7)
> qqnorm(x, col = "red")
> qqline(x)
```

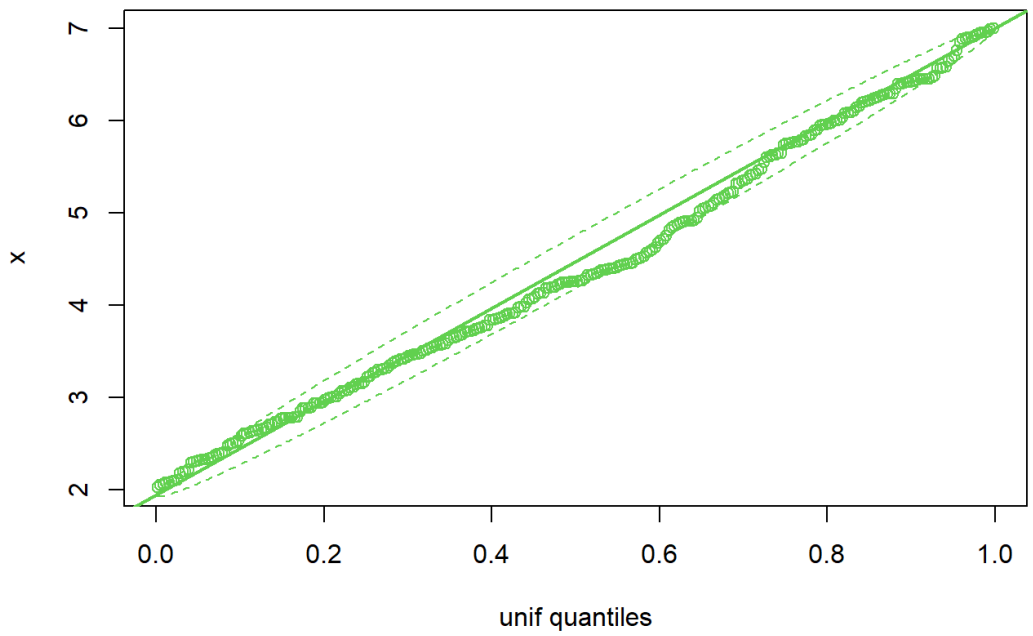


The same example

```
> qqplot.das(x, "norm", col = palette()[2])
```

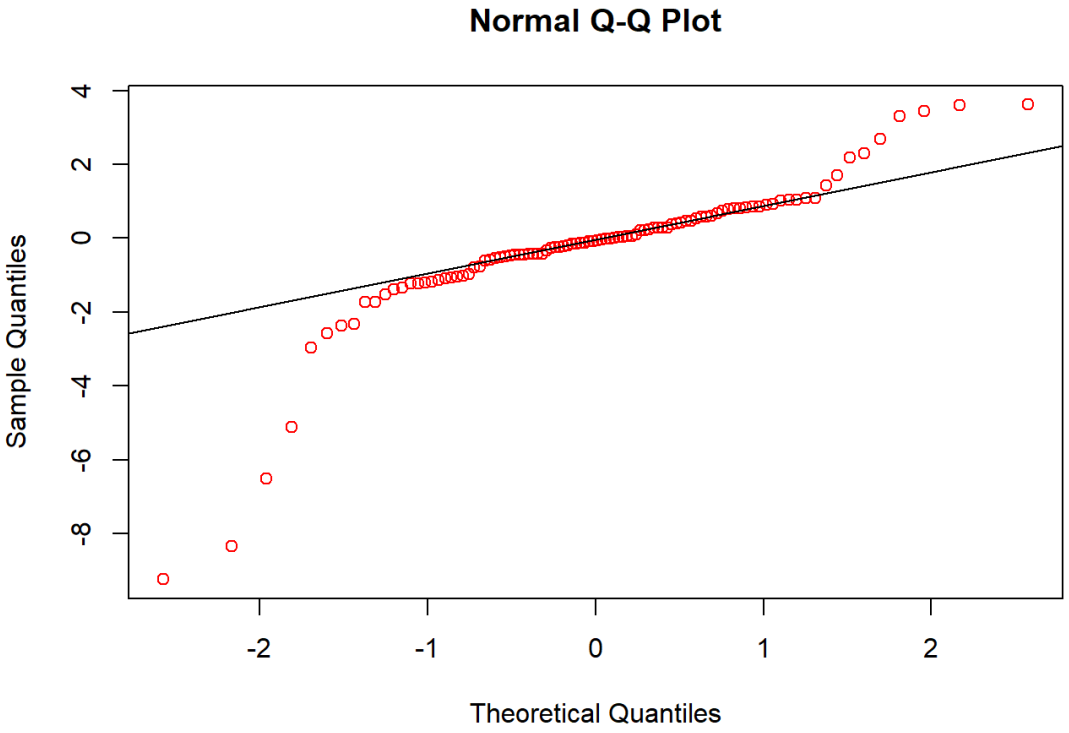


```
> qqplot.das(x, "unif", col = palette()[3])
```



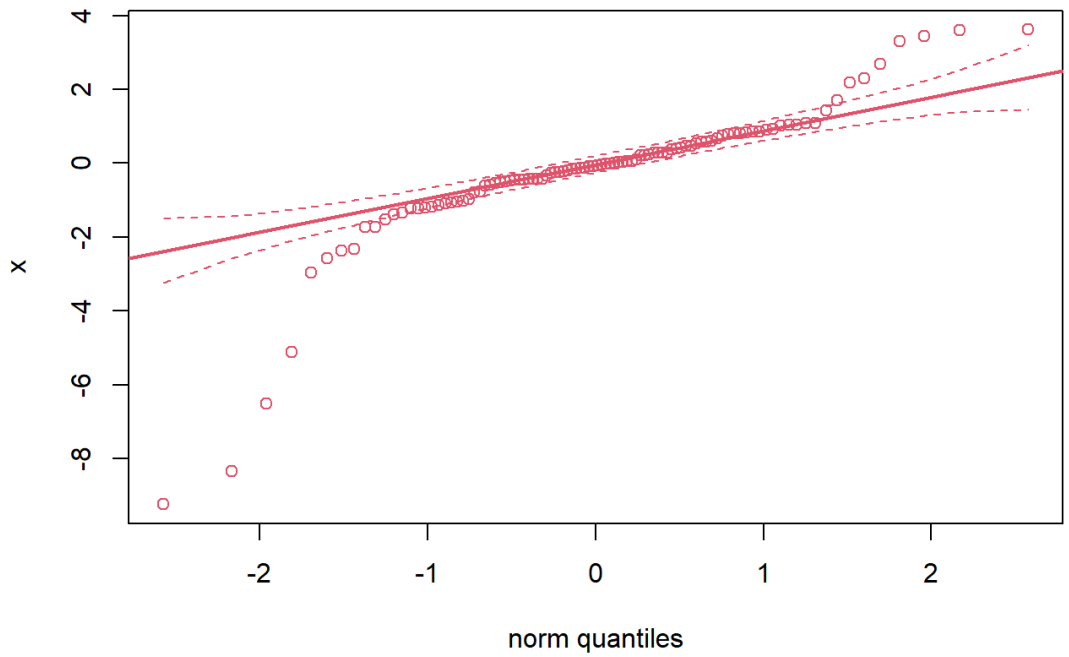
Example X~t(3)

```
> x = rt(100, df=3)
> qqnorm(x, col = "red")
> qqline(x)
```



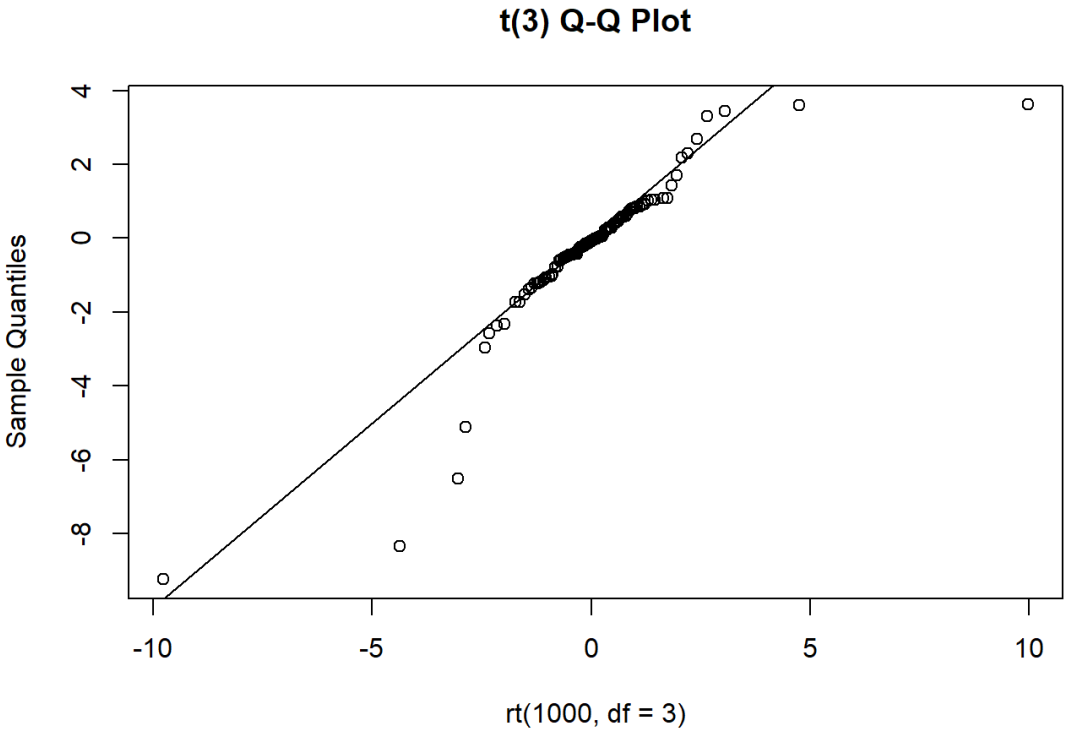
The same example

```
> qqplot.das(x, "norm", col = palette()[2])
```



The same example with qqplot

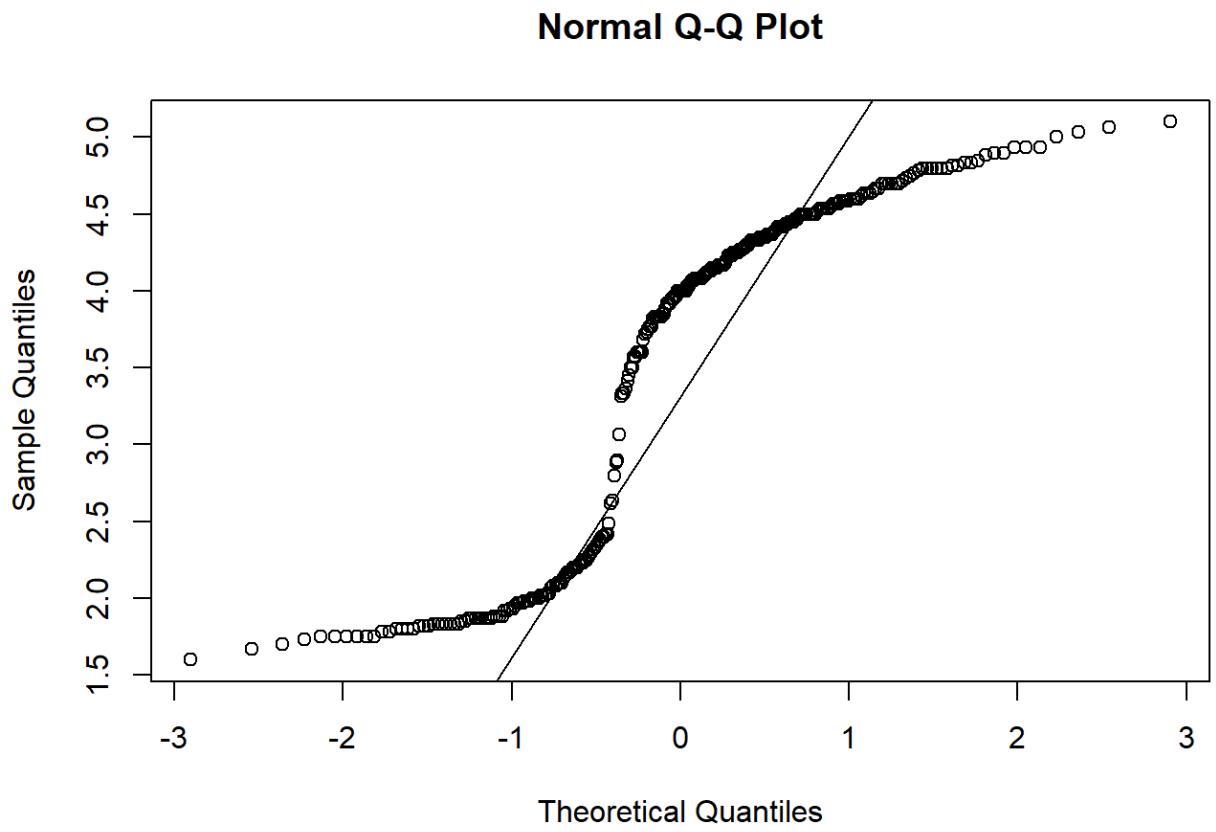
```
> qqplot(rt(1000,df=3), x,
+        main="t(3) Q-Q Plot",
+        ylab="Sample Quantiles")
> abline(0,1)
```



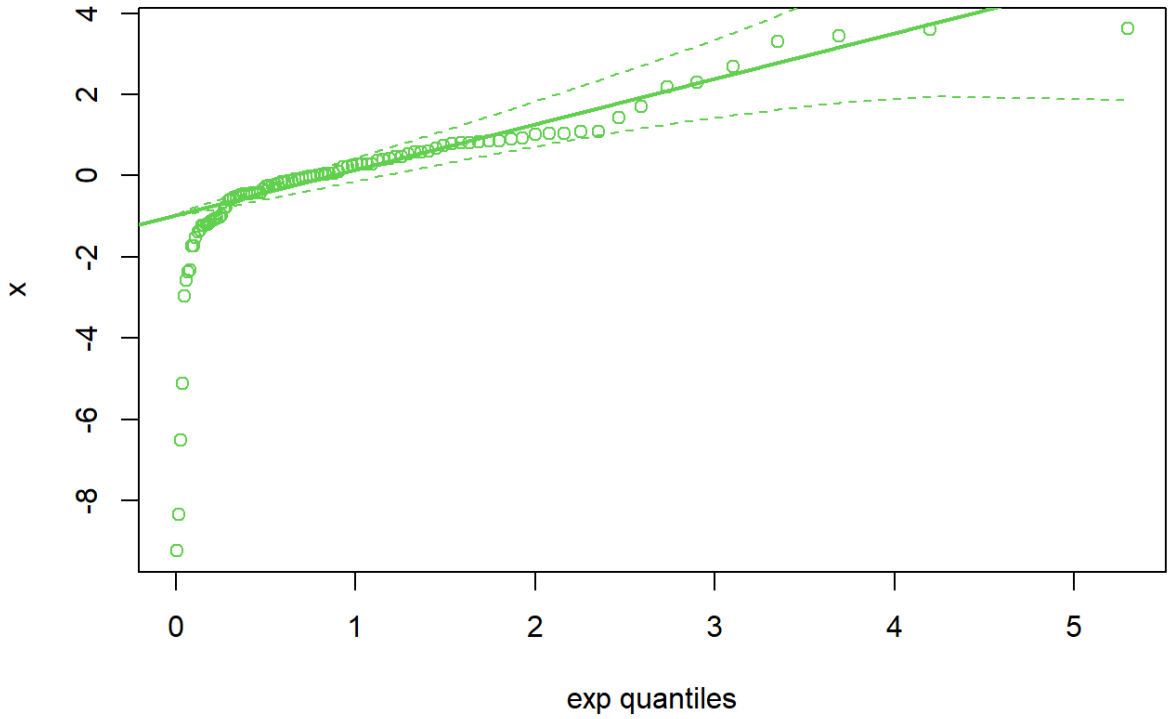
Example - faithful

```
> head(faithful)
  eruptions waiting
1     3.600      79
2     1.800      54
3     3.333      74
4     2.283      62
5     4.533      85
6     2.883      55
> faithful$eruptions
  [1] 3.600 1.800 3.333 2.283 4.533 2.883 4.700 3.600
1.950 4.350 1.833 3.917
 [13] 4.200 1.750 4.700 2.167 1.750 4.800 1.600 4.250
1.800 1.750 3.450 3.067
 [25] 4.533 3.600 1.967 4.083 3.850 4.433 4.300 4.467
3.367 4.033 3.833 2.017
 [37] 1.867 4.833 1.833 4.783 4.350 1.883 4.567 1.750
4.533 3.317 3.833 2.100
 [49] 4.633 2.000 4.800 4.716 1.833 4.833 1.733 4.883
3.717 1.667 4.567 4.317
 [61] 2.233 4.500 1.750 4.800 1.817 4.400 4.167 4.700
2.067 4.700 4.033 1.967
 [73] 4.500 4.000 1.983 5.067 2.017 4.567 3.883 3.600
4.133 4.333 4.100 2.633
 [85] 4.067 4.933 3.950 4.517 2.167 4.000 2.200 4.333
1.867 4.817 1.833 4.300
 [97] 4.667 3.750 1.867 4.900 2.483 4.367 2.100 4.500
4.050 1.867 4.700 1.783
[109] 4.850 3.683 4.733 2.300 4.900 4.417 1.700 4.633
2.317 4.600 1.817 4.417
[121] 2.617 4.067 4.250 1.967 4.600 3.767 1.917 4.500
2.267 4.650 1.867 4.167
[133] 2.800 4.333 1.833 4.383 1.883 4.933 2.033 3.733
4.233 2.233 4.533 4.817
[145] 4.333 1.983 4.633 2.017 5.100 1.800 5.033 4.000
2.400 4.600 3.567 4.000
[157] 4.500 4.083 1.800 3.967 2.200 4.150 2.000 3.833
3.500 4.583 2.367 5.000
[169] 1.933 4.617 1.917 2.083 4.583 3.333 4.167 4.333
4.500 2.417 4.000 4.167
```

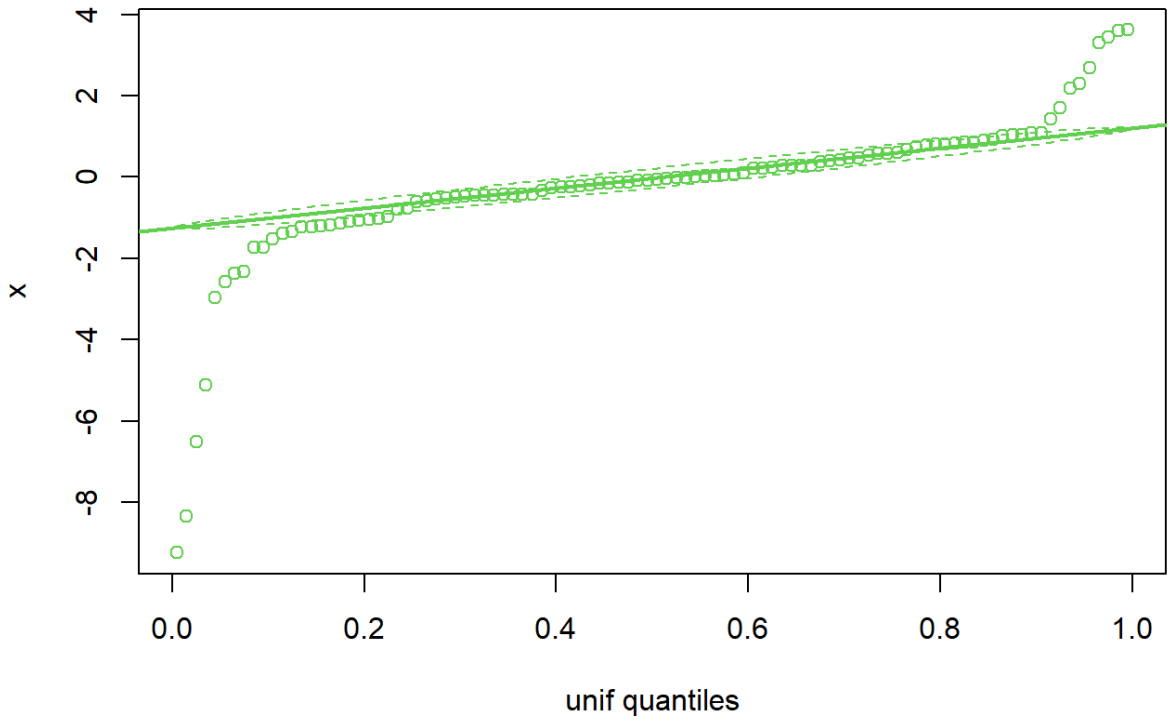
```
[181] 1.883 4.583 4.250 3.767 2.033 4.433 4.083 1.833
4.417 2.183 4.800 1.833
[193] 4.800 4.100 3.966 4.233 3.500 4.366 2.250 4.667
2.100 4.350 4.133 1.867
[205] 4.600 1.783 4.367 3.850 1.933 4.500 2.383 4.700
1.867 3.833 3.417 4.233
[217] 2.400 4.800 2.000 4.150 1.867 4.267 1.750 4.483
4.000 4.117 4.083 4.267
[229] 3.917 4.550 4.083 2.417 4.183 2.217 4.450 1.883
1.850 4.283 3.950 2.333
[241] 4.150 2.350 4.933 2.900 4.583 3.833 2.083 4.367
2.133 4.350 2.200 4.450
[253] 3.567 4.500 4.150 3.817 3.917 4.450 2.000 4.283
4.767 4.533 1.850 4.250
[265] 1.983 2.250 4.750 4.117 2.150 4.417 1.817 4.467
> qqnorm(faithful$eruptions)
> qqline(faithful$eruptions)
```



```
> qqplot.das(x, "exp", col = palette()[3])
```

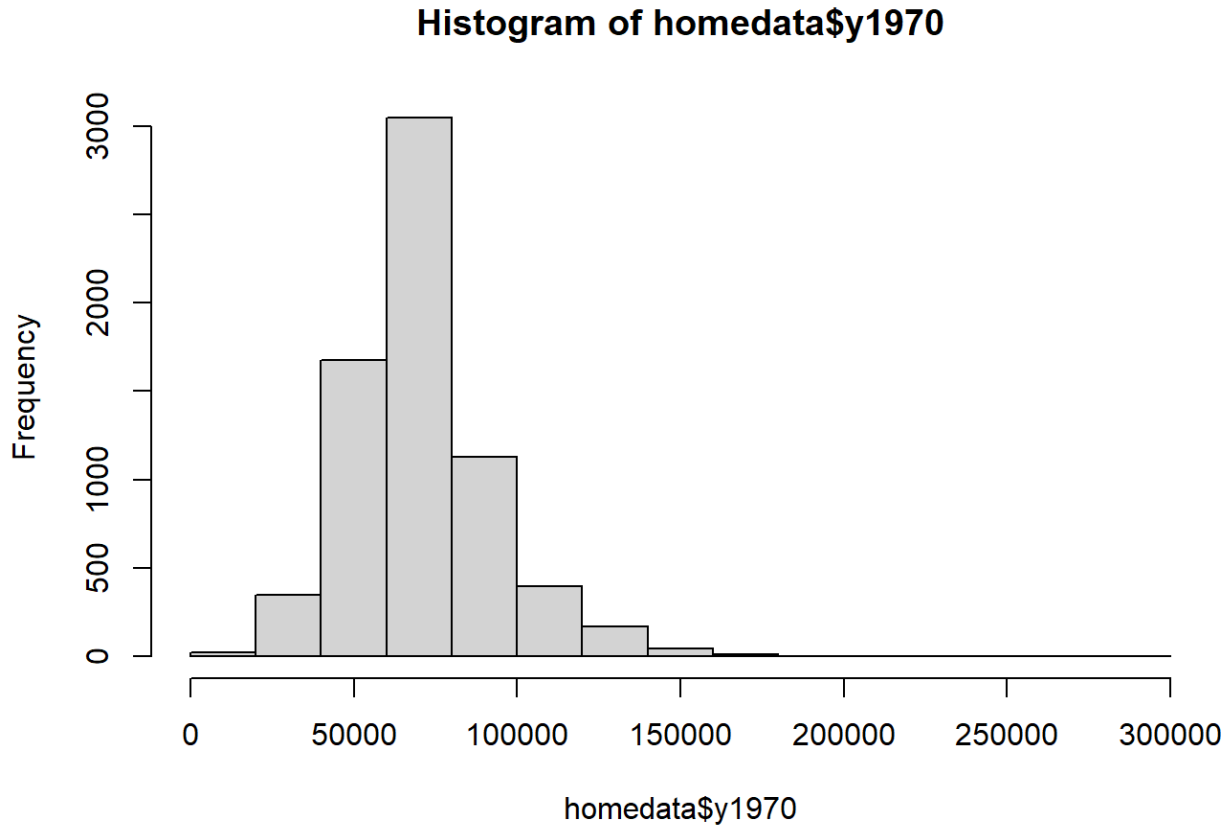



```
> qqplot.das(x, "unif", col = palette()[3])
```

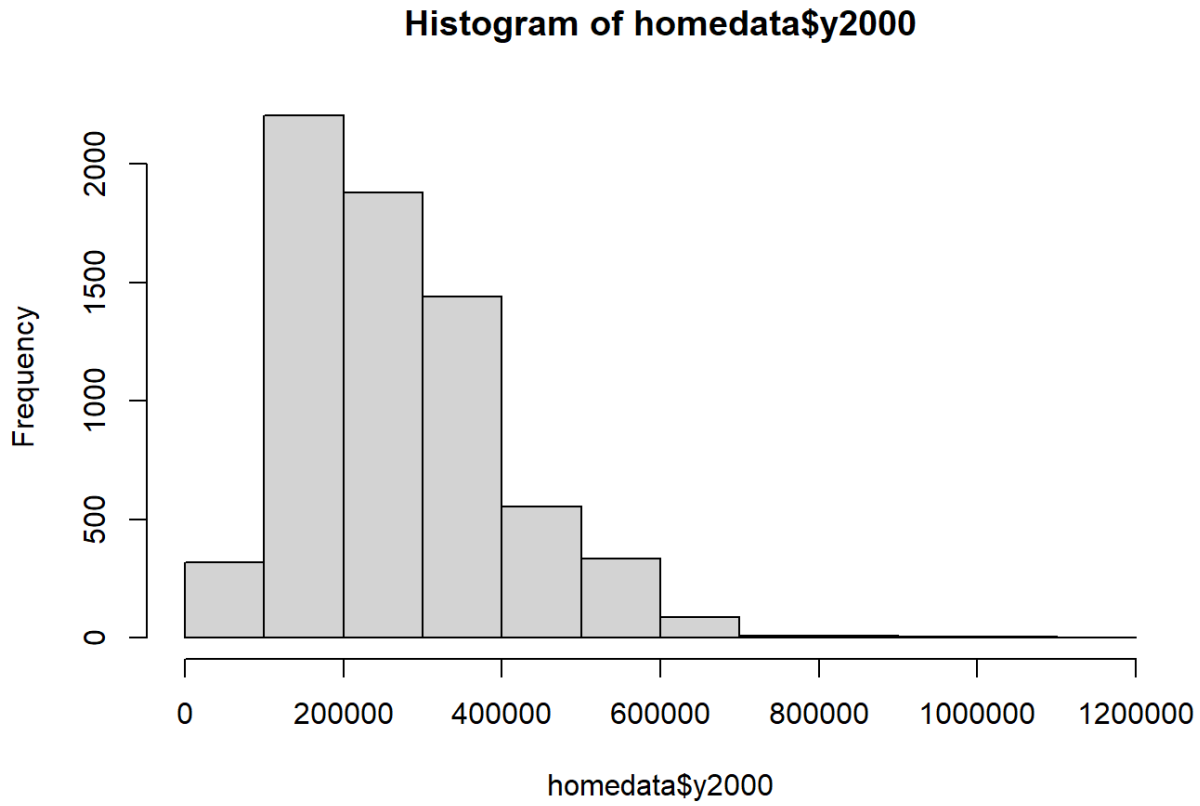


Example - homedata

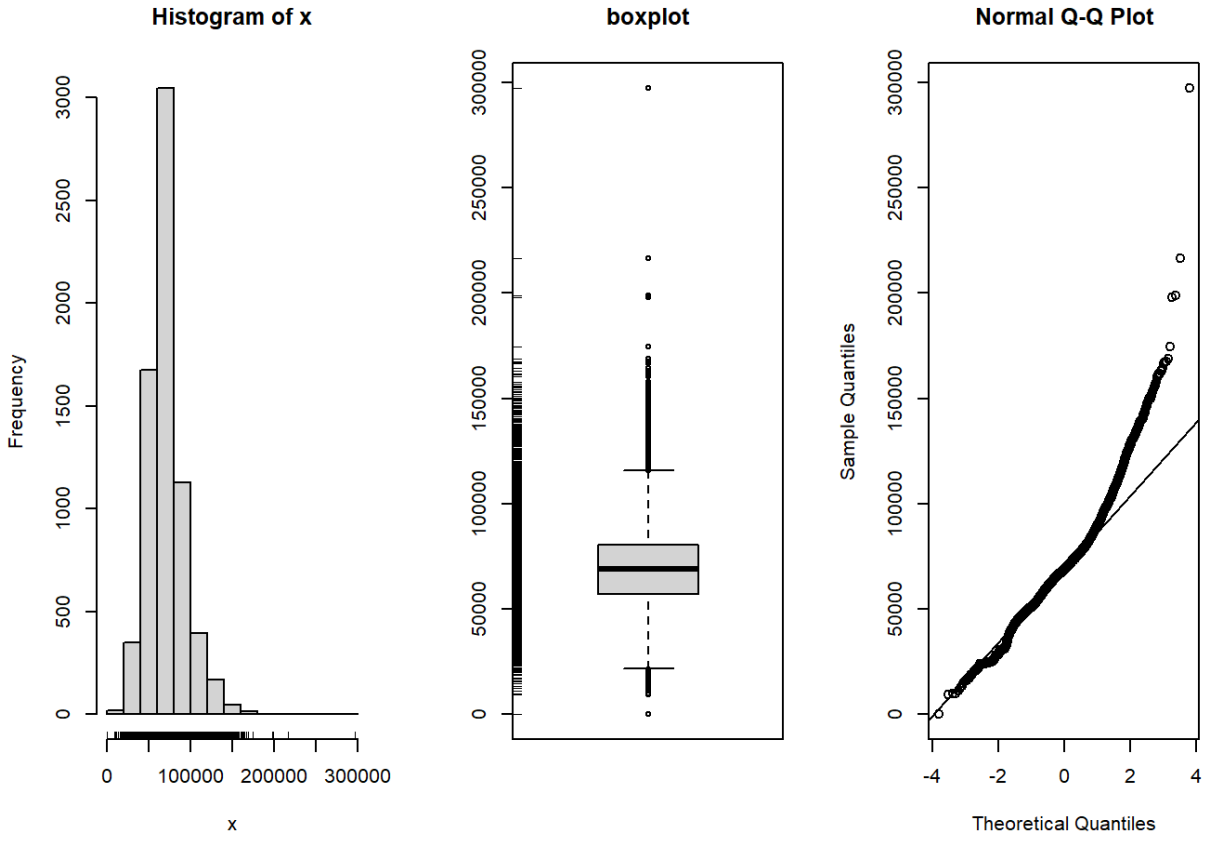
```
> head(homedata)
   y1970  y2000
1  89700 359100
2 118400 504500
3 116400 477300
4 122000 500400
5  91500 433900
6 102800 464800
> hist(homedata$y1970)
```



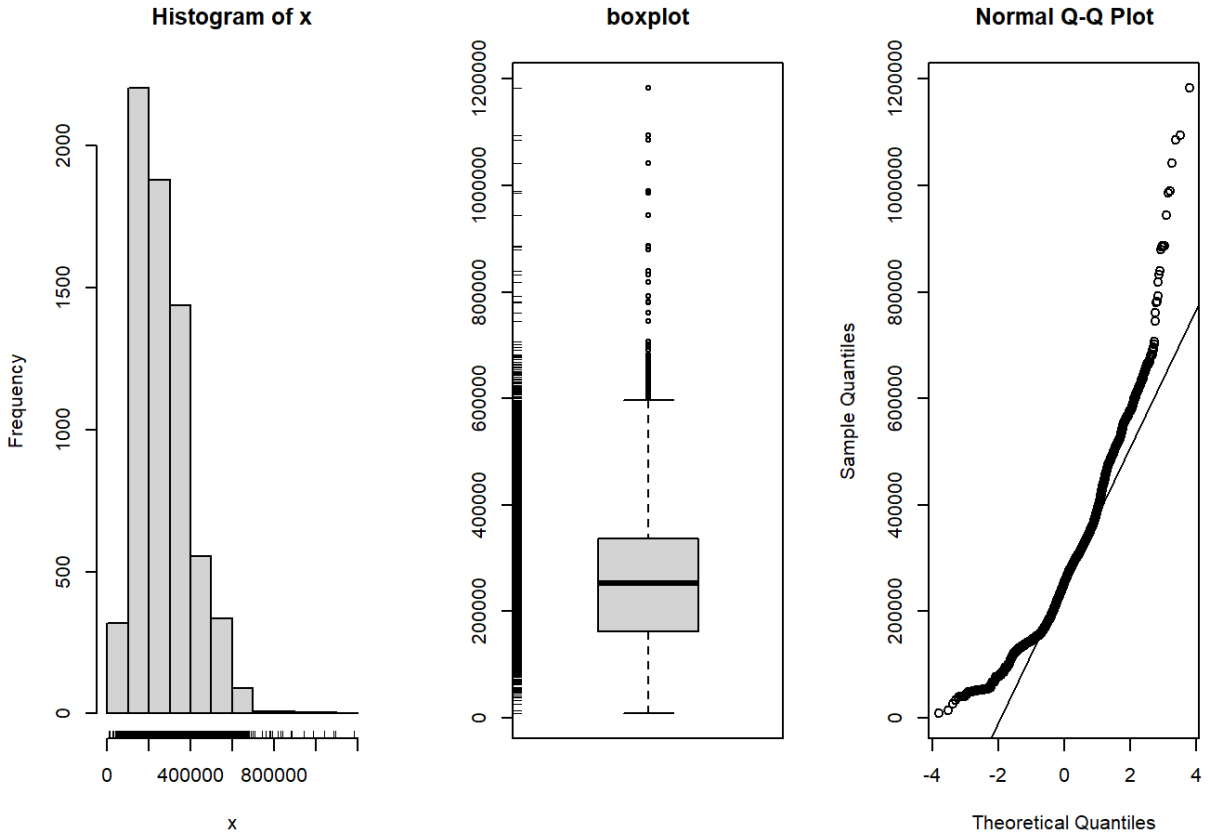
```
> hist(homedata$y2000)
```



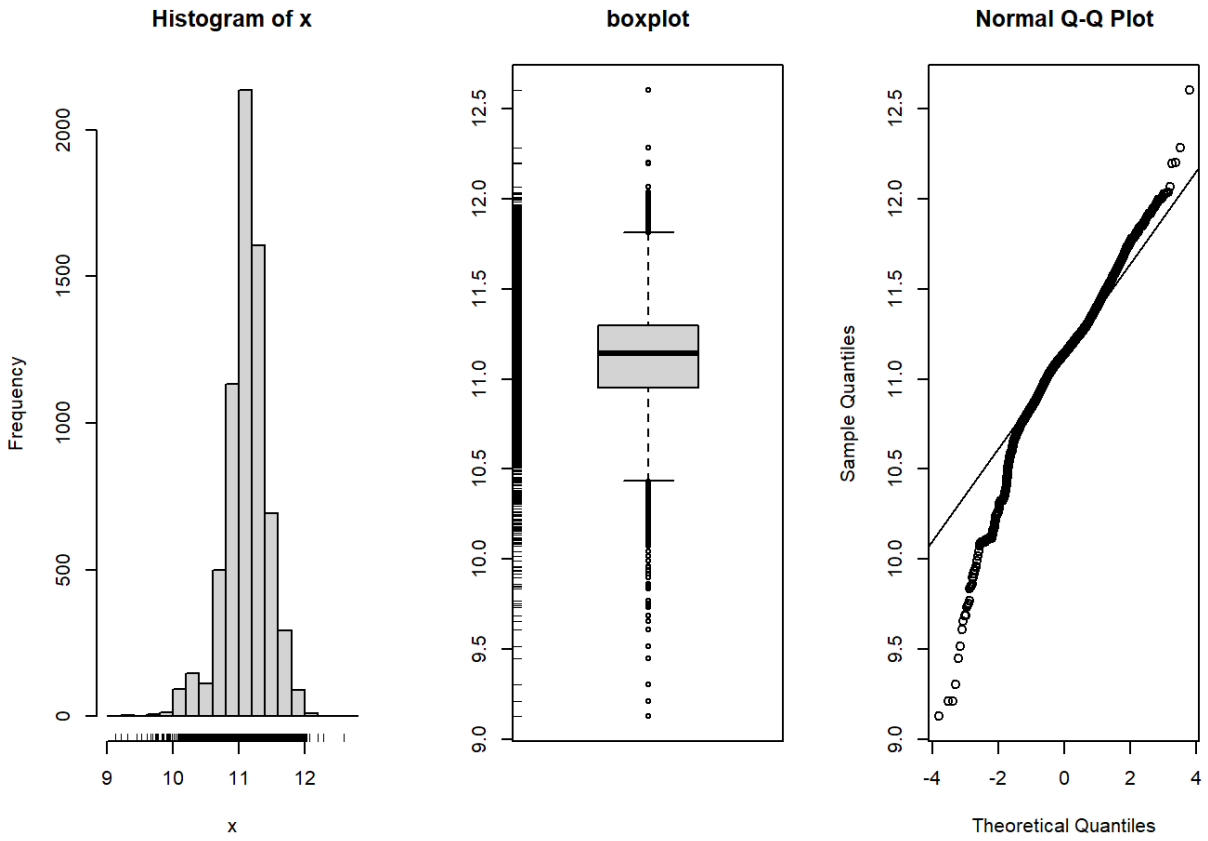

```
> simple.eda(homedata$y1970)
```



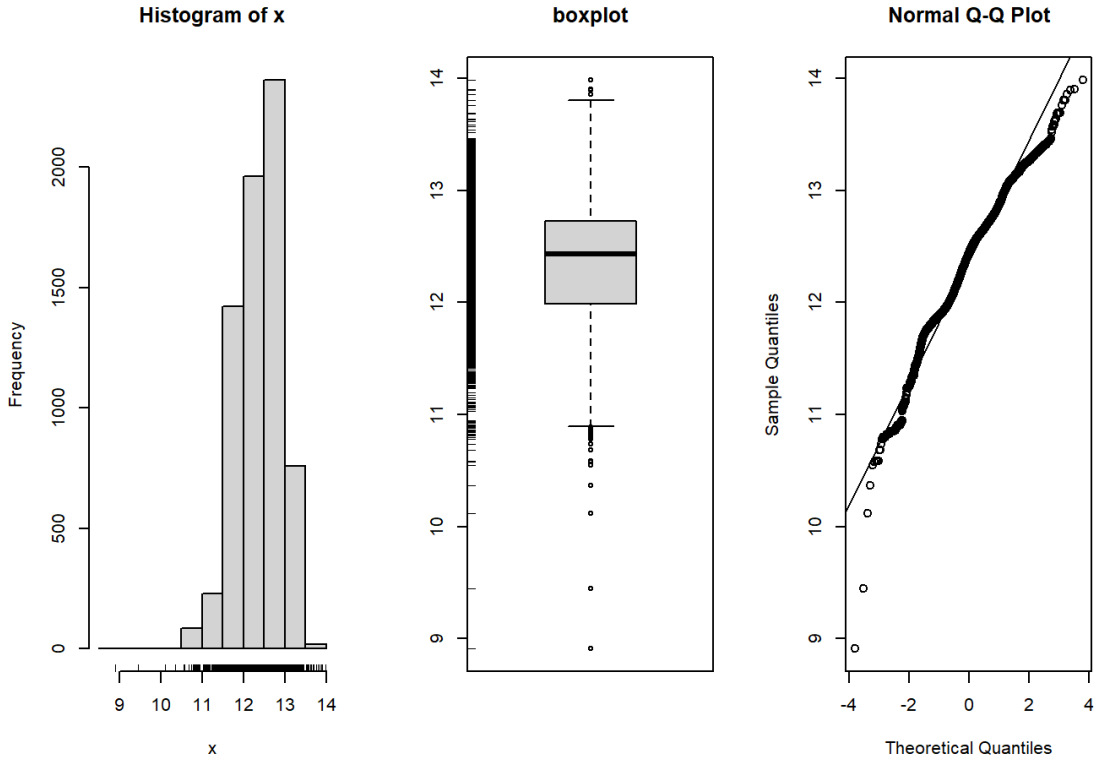
```
> simple.eda(homedata$y2000)
```




```
> simple.eda(log(homedata$y1970[homedata$y1970>0]))
```

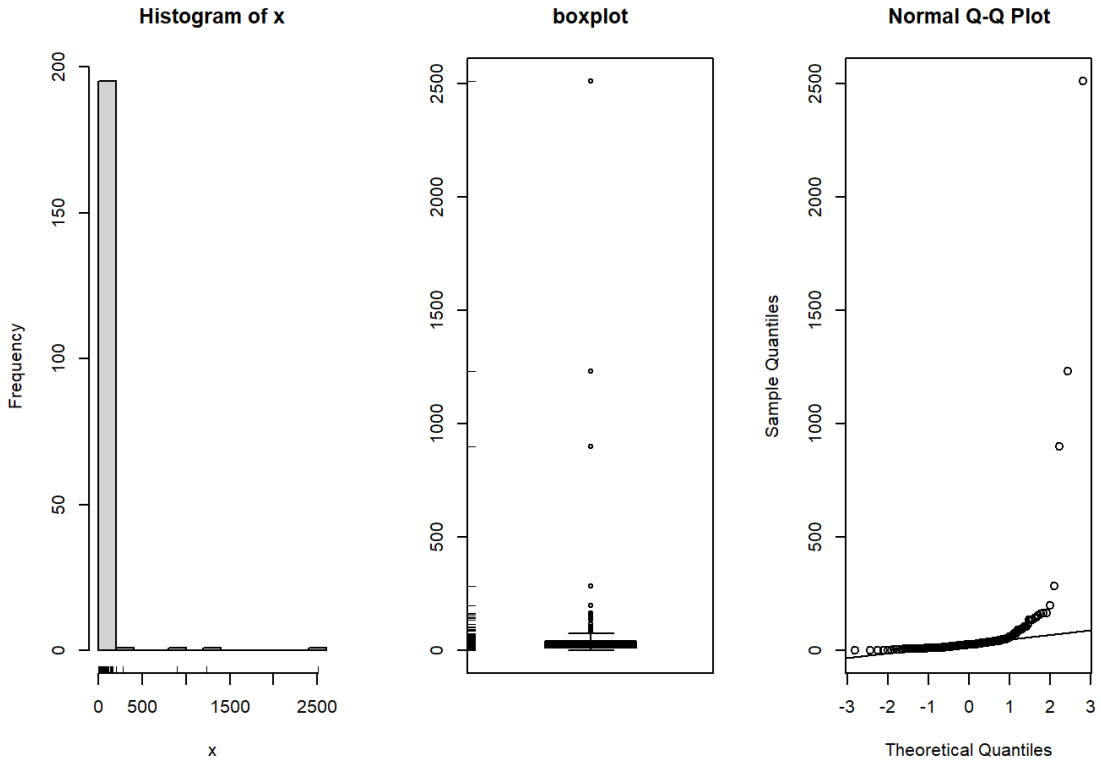


```
> simple.eda(log(homedata$y2000[homedata$y2000>0]))
```

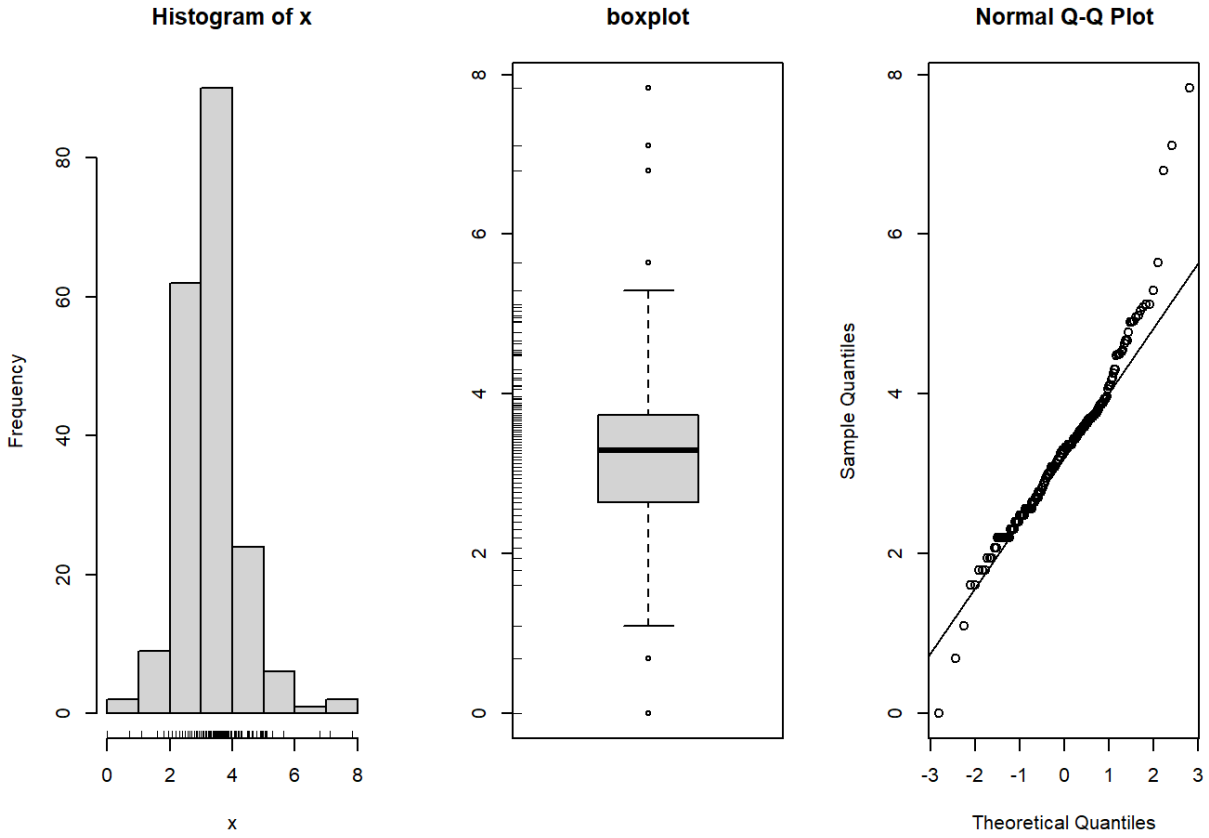


Example - exec.pay

```
> head(exec.pay)
[1] 136  74   8  38  46  43
> simple.eda(exec.pay)
```



```
> simple.eda(log(exec.pay[exec.pay > 0]))
```



Tests for normality

Jarque-Bera test

- If $X \in N(0,\ 1)$ and if we have a sample of n independent observations on X then the **empirical skewness**

$$skew_n := \frac{\frac{1}{n}\sum_{i=1}^{n}(X_i - \overline{X}_n)^3}{s_n^3} \in N(0,\ D_s), \text{ where}$$

$$D_s = \frac{6(n-2)}{(n+1)(n+3)}$$

If the empirical skewness of the sample is outside the interval $(-3\sqrt{D_s},\ 3\sqrt{D_s})$ we can't assume that the observed random variable is normally distributed.

- If $X \in N(0,\ 1)$ and if we have a sample of n independent observations on X then the **empirical kurtosis**

$$kurt_n := \frac{\frac{1}{n}\sum_{i=1}^{n}(X_i - \overline{X}_n)^4}{s_n^4} - 3 \in N(0,\ D_k), \text{ where}$$

$$D_k = \frac{24n(n-2)(n-3)}{(n+1)^2(n+3)(n+5)}$$

If the empirical kurtosis of the sample is outside the interval $(-3\sqrt{D_k},\ 3\sqrt{D_k})$ we can't assume that the observed random variable is normally distributed.

- **Jarque-Bera** use that

$$\lim_{n\to\infty}\frac{D_s}{\frac{6}{n}} = 1, \qquad \frac{skew}{\sqrt{\frac{6}{n}}} \approx N(0,1), \qquad \frac{skew^2}{\frac{6}{n}} \approx \mathcal{X}^2(1)$$

and

$$\lim_{n\to\infty} \frac{D_s}{\frac{24}{n}} = 1, \qquad \frac{skew}{\sqrt{\frac{24}{n}}} \approx N(0,1), \qquad \frac{skew^2}{\frac{24}{n}} \approx \mathcal{X}^2(1)$$

$$JB = \frac{skew^2}{\frac{6}{n}} + \frac{kurt^2}{\frac{24}{n}} = n\left(\frac{skew^2}{6} + \frac{kurt^2}{24}\right) \approx \mathcal{X}^2(2)$$

There for if JB is less than the 0.95-quantile of the $\mathcal{X}^2(2)$ we say that the observed random variable can be assumed to be normally distributed. Otherwise we reject the normality.

As we see **Jarque-Bera test** can be used only to test if the observed random variable is normally distributed.

We can perform the **Jarque-Bera test** by using the function `jarque.bera.test` from the `tseries` package.

```
> x <- rnorm(1000, 3, 4)
> library(tseries)
Warning: package 'tseries' was built under R version
4.0.3
Registered S3 method overwritten by 'quantmod':
  method            from
  as.zoo.data.frame zoo
> jarque.bera.test(x)

    Jarque Bera Test

data:  x
X-squared = 5.2943, df = 2, p-value = 0.07085
> jb <- jarque.bera.test(x)
> class(jb)
[1] "htest"
> attributes(jb)
$names
[1] "statistic" "parameter" "p.value"   "method"
"data.name"

$class
[1] "htest"
```

The output from the `jarque.bera.test` function is a `htest` object. It has `statistic` attribute showing the JB value, `parameter` attribute showing the **degrees of freedom** of the $\mathcal{X}^2$ distribution in this case 2. `p.value` is the most important attribute. When it is less than 0.05 we say that the observed random variable is not normal. `method` attribute shows the name of the test **Jarque-Bera test**.

In our case the $p-value = 0.6944 > 0.05$, therefore, we can assume that the observed random variable is normally distributed.

```
> x <- rexp(1000, 3)
> jarque.bera.test(x)

    Jarque Bera Test

data:  x
X-squared = 1827.2, df = 2, p-value < 2.2e-16
```

In this case the $p-value < 2.2e-16$, therefore, we can't assume that the observed random variable is normally distributed.

Shapiro-Wilk test

Shapiro-Wilk test is similar test only for normality. It can be performed by using the `shapiro.test` function. The output is from `htest` class and has the same attributes.

```
> x <- rnorm(1000, 3, 4)
> shapiro.test(x)

    Shapiro-Wilk normality test

data:  x
W = 0.99857, p-value = 0.6011
```

In our case the $p-value = 0.4401 > 0.05$, therefore, we can assume that the observed random variable is normally distributed.

```
> x <- rexp(1000, 3)
> shapiro.test(x)
```

```
    Shapiro-Wilk normality test

data:  x
W = 0.83137, p-value < 2.2e-16
```

In this case the $p-value < 2.2e-16$, therefore, we can't assume that the observed random variable is normally distributed.

Anderson–Darling test

Anderson–Darling test is similar test for normality, but it can also test for other continuous distributions. It can be performed by using the `ad.test` function from the `nortest` package. The output is again from `htest` class and has the same attributes.

```
> x <- rnorm(1000, 3, 4)
> library(nortest)
Warning: package 'nortest' was built under R version
4.0.3
> ad.test(x)

    Anderson-Darling normality test

data:  x
A = 0.67456, p-value = 0.07794
```

In our case the $p-value = 0.8994 > 0.05$, therefore, we can assume that the observed random variable is normally distributed.

```
> x <- rexp(1000, 3)
> ad.test(x)

    Anderson-Darling normality test

data:  x
A = 43.178, p-value < 2.2e-16
```

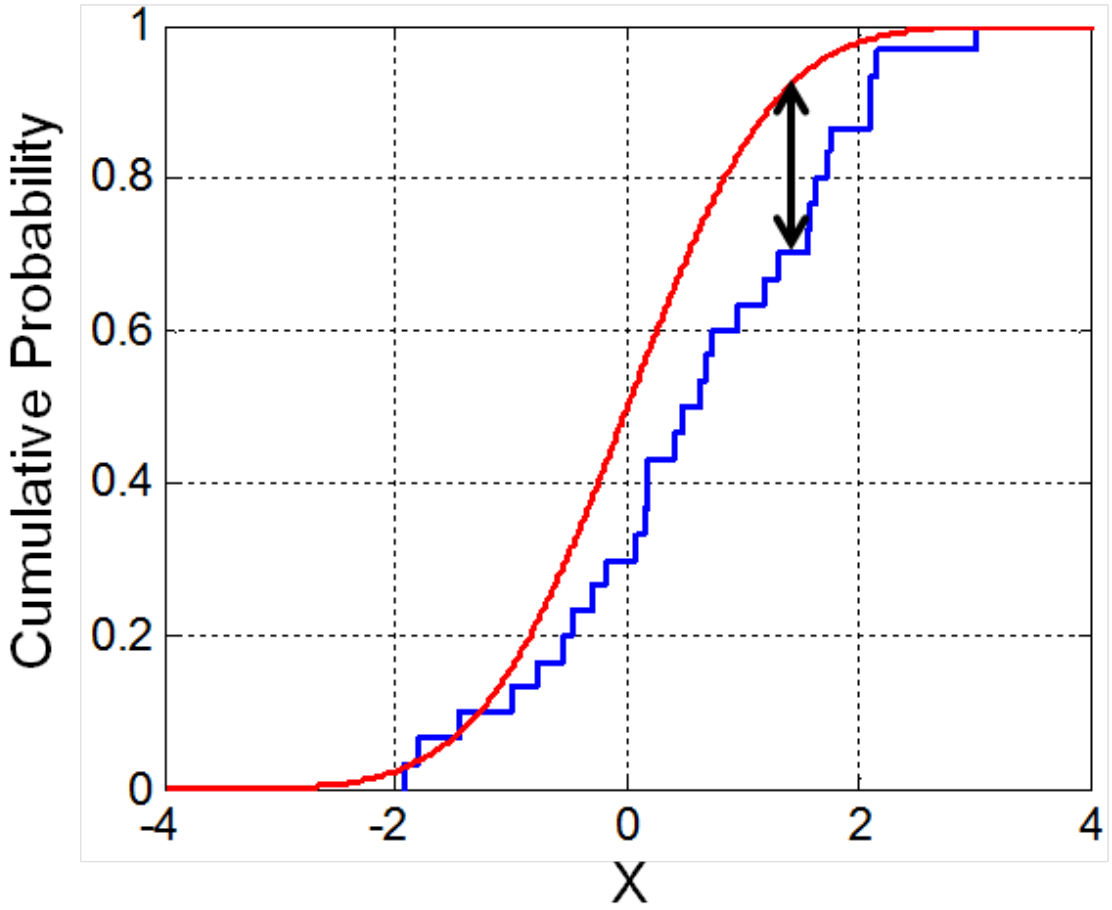
In this case the $p-value < 2.2e-16$, therefore, we can't assume that the observed random variable is normally distributed.

Kolmogorov–Smirnov test

Kolmogorov–Smirnov test can test for any distribution.

- **Kolmogorov** noticed that the supremum in the **Glivenko-Cantelli theorem** is reached in order statistics and finds the distribution of

$$D_n := \sup_{x \in \mathbb{R}} |F_n(x) - F_X(x)| =$$
$$= \max_{i=1,\ ...,\ n} |F_n(x_{(i)}) - F_X(x_{(i)})| =$$
$$= \max_{i=1.\ ...,\ n} \left| \frac{i}{n} - F_X(x_{(i)}) \right|$$



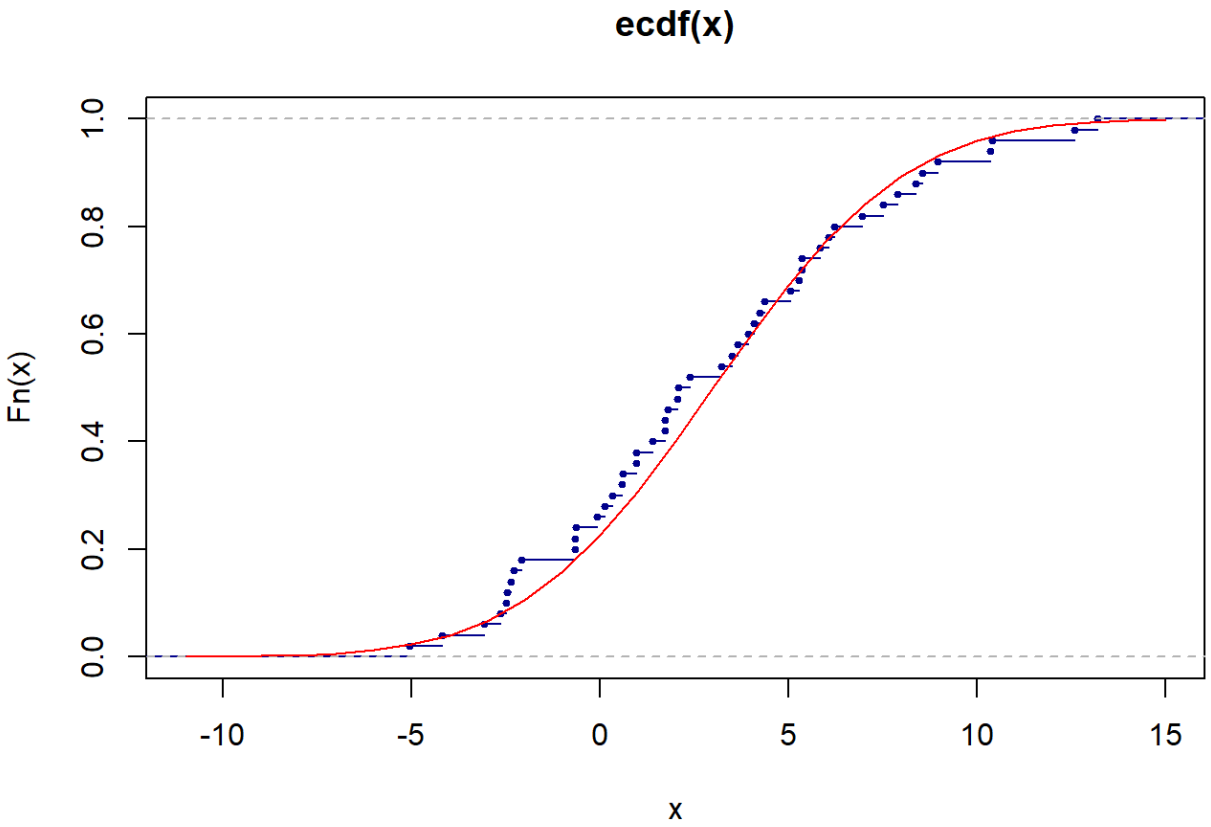
Then he compares $\sqrt{n} D_n$ with quantiles of the **Kolmogorov** distribution

$$\mathbb{P}(K \leq x) = 1 - 2\sum_{k=1}^{\infty}(-1)^{k+1}e^{2k^2x^2},\ x > 0$$

- **Smirnov** generalizes the test to test for equality between distributions of two random variables.

Kolmogorov–Smirnov test can be performed by using the `ks.test` function. The output is again from `htest` class and has the same attributes.

```
> x <- rnorm(50, 3, 4)
> plot(ecdf(x), xlim = c(-11, 15), cex = 0.5, col =
"darkblue")
> xCoord <- -11:15
> x2 <- pnorm(xCoord, 3, 4)
> lines(xCoord, x2, col = "red")
```




```
> ks.test(x, "pnorm", mean(x), sd(x))

    One-sample Kolmogorov-Smirnov test
```

```
data:  x
D = 0.08222, p-value = 0.8604
alternative hypothesis: two-sided
```

The statistic D_n is the biggest distance between the blue and the red line.

In our case the $p-value = 0.7286 > 0.05$, therefore, we can assume that the observed random variable is normally distributed.

```
> x <- rexp(1000, 3)
> ks.test(x, "pnorm", mean(x), sd(x))

    One-sample Kolmogorov-Smirnov test

data:  x
D = 0.1587, p-value < 2.2e-16
alternative hypothesis: two-sided
```

In this case the $p-value = 2.2e-16$, therefore, we can't assume that the observed random variable is normally distributed.