Введение в R

R — язык программирования для статистической обработки данных и работы с графикой, а также свободная программная среда вычислений с открытым исходным кодом в рамках проекта GNU. R широко используется как статистическое программное обеспечение для анализа данных и фактически стал стандартом для статистических программ.

Коротко говоря, R применяется везде, где нужна работа с данными. Это не только статистика в узком смысле слова, но и «первичный» анализ (графики, таблицы сопряженности) и продвинутое математическое моделирование. В принципе, R может использоваться и там, где в настоящее время принято использовать специализированные программы математического анализа, такие как MATLAB или Octave. Но, разумеется, более всего его применяют для статистического анализа — от вычисления средних величин до вейвлет-преобразований и временных рядов. Географически R распространен тоже очень широко. Трудно найти американский или западноевропейский университет, где бы не работали с R. Очень многие серьезные компании (скажем, Boeing) устанавливают R для работы.

У R два главных **преимущества**: неимоверная гибкость и свободный код. Гибкость позволяет создавать приложения (пакеты) практически на любой случай жизни. Нет, кажется, ни одного метода современного статистического анализа, который бы не был сейчас представлен в R.

У R есть и немало **недостатков**. Самый главный из них — это трудность обучения программе. Второй недостаток R — относительная медлительность. Некоторые функции, особенно использующие циклы, и виды объектов, особенно списки и таблицы данных, «работают» в десятки раз медленнее, чем их аналоги в коммерческих пакетах.

Установка R (https://losst.ru/ustanovka-r-v-ubuntu)

Среды разработки

R kernel for Jupyter Notebook (https://irkernel.github.io/)

RStudio (https://www.rstudio.com/)

Смена ядра в Jupyter: Kernel -> Change kernel.

Изучение R (на русском)

(описание на сайте (http://tukachev.flogiston.ru/blog/?p=1352))

Роберт И. Кабаков «R в действии. Анализ и визуализация данных на языке R»

Алексей Шипунов, Е. Балдин «Наглядная статистика. Используем R!»

Сергей Мастицкий, Владимир Шитиков «Статистический анализ и визуализация данных с помощью R»

Stepic: Анализ данных в R (https://stepic.org/course/129)

Stepic: Основы программирования на R (https://stepic.org/course/497)

Заметки об R (http://mpoctok.narod.ru/r/intro.htm)

Изучение R (на английском)

An Introduction to R (https://cran.r-project.org/doc/manuals/r-release/R-intro.pdf) (от разработчиков)

https://www.rdocumentation.org/ (https://www.rdocumentation.org/)

https://www.r-project.org/ (https://www.r-project.org/)

https://rdrr.io/ (https://rdrr.io/)

==========

Векторы

Векторы и их конкатенации. Число - тоже вектор.

In [1]:

```
c(1, 2, 3, 4, 5)
c(1)
c(c(1, 2), c(3, 4), c(5))
c('a', 'b', 'c')
```

```
1  2  3  4  5
```

```
1
```

```
1  2  3  4  5
```

```
'a'  'b'  'c'
```

Создание векторов. Правая граница **включается**.

In [2]:

```
1:5
5:1
seq(from = 1, to = 10, by = 2)
seq(from = 10, to = 1, by = -3)
rep(x = 0, times = 10)  # 10 раз повторить вектор из элемента 0
rep(x = 1:3, times = 2)  # 2 раза повторить вектор 1:3
```

```
1  2  3  4  5

5  4  3  2  1

1  3  5  7  9

10  7  4  1

0  0  0  0  0  0  0  0  0  0

1  2  3  1  2  3
```

Длина вектора

In [3]:

```
length(1:5)
```

```
5
```

Арифметические операции над векторами

In [4]:

```
1:5 + 5:1
1:5 * 5:1
1:5 - 5:1
1:5 / 5:1
1:5 %% 5:1  # остаток от деления
1 + 1:5
2 * 1:5
```

```
6  6  6  6  6

5  8  9  8  5

-4  -2  0  2  4

0.2  0.5  1  2  5

1  2  0  0  0

2  3  4  5  6

2  4  6  8  10
```

In [5]:

```
(1:5) ^ 2
(1:5) ** 2
1:5 ^ 2  # эквивалентно 1:(5 ^ 2)
1:5 ** 2  # эквивалентно 1:(5 ** 2)
```

```
1  4  9  16  25

1  4  9  16  25

1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22
23  24  25

1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22
23  24  25
```

Циклическое дополнение объекта меньшего размера до объектов большего размера

In [6]:

```
1:10 + 1:5
rep(x = 0, times = 10) + 1:3
```

```
2  4  6  8  10  7  9  11  13  15
```

```
Warning message in rep(x = 0, times = 10) + 1:3:
“длина большего объекта не является произведением длины меньшего объек
та”
```

```
1  2  3  1  2  3  1  2  3  1
```

Описательные статистики

In [7]:

```
sum(1:10)
min(1:10)
max(1:10)
mean(1:10)  # выборочное среднее
var(1:10)  # выборочная дисперсия
sd(1:10)  # выборочное стандартное отклонение
median(1:10)  # выборочная медиана
```

```
55
```

```
1
```

```
10
```

```
5.5
```

```
9.16666666666667
```

```
3.02765035409749
```

```
5.5
```

Индексы. Индексация происходит **с единицы**. В качестве индексов можно передавать вектор индексов. Можно передавать отрицательные индексы - означает "все кроме".

In [8]:

```
x <- 1:10
x[c(2, 5, 3)]
x[3:6]
x[-c(2, 5, 3)]  # исключаются элементы с индексами 2,3,5
```

```
2  5  3
```

```
3  4  5  6
```

```
1  4  6  7  8  9  10
```

Упражнение. Создайте векторы $(21, 22, \ldots, 33)$ и $(67, 64, \ldots, 43)$. Какова их длина? Можно ли их сложить поэлементно? Чему равны минимальный и максимальный элемент такой суммы?

In [9]:

```r
a <- 21:33
b <- seq(from = 67, to = 43, by = -3)
length(a)
length(b)
d <- a + b
d
c(min(d), max(d))
```

```
13

9

Warning message in a + b:
“длина большего объекта не является произведением длины меньшего объек
та”

88  86  84  82  80  78  76  74  72  97  95  93  91


72  97
```

Матрицы

Создание матриц. Как и раньше, если данных не хватает до требуемого размера матрицы, то они будут циклически размножены. Если же данных больше, то они будут обрезаны.

In [10]:

```
matrix(1:6, nrow = 2)
matrix(1:5, nrow = 2, ncol = 4)  # заполнение по столбцам
matrix(1:5, nrow = 2, ncol = 4, byrow = TRUE)  # заполнение по строкам
matrix(1:50, nrow = 2, ncol = 4)
```

A matrix:
2 × 3 of
type int

1	3	5
2	4	6

```
Warning message in matrix(1:5, nrow = 2, ncol = 4):
“длина данных [5] не является множителем количества строк [2]”
```

A matrix: 2 ×
4 of type int

1	3	5	2
2	4	1	3

```
Warning message in matrix(1:5, nrow = 2, ncol = 4, byrow = TRUE):
“длина данных [5] не является множителем количества строк [2]”
```

A matrix: 2 ×
4 of type int

1	2	3	4
5	1	2	3

```
Warning message in matrix(1:50, nrow = 2, ncol = 4):
“длина данных [50] не является множителем количества столбцов [4]”
```

A matrix: 2 ×
4 of type int

1	3	5	7
2	4	6	8

In [11]:

```
x <- matrix(1:20, nrow=4)
print(x)
```

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

Индексация в матрицах

In [12]:

```
x[1,]  # первая строка
x[,1]  # первый столбец
x[1, 1]  # первый элемент

# подматрицы
x[2:3, 2:4]
x[c(1, 4), c(2, 5)]
x[-c(1, 4), c(2, 5)]
```

1 5 9 13 17

1 2 3 4

1

A matrix: 2 × 3 of type int

6	10	14
7	11	15

A matrix: 2 × 2 of type int

5	17
8	20

A matrix: 2 × 2 of type int

6	18
7	19

Если передать просто индекс, то матрица сначала как бы преобразуется в вектор, из которого будет взят соответствующий элемент

In [13]:

```
x[10]
x[-10]
```

10

1 2 3 4 5 6 7 8 9 11 12 13 14 15 16 17 18 19 20

Конкатенация матриц

In [14]:

```r
x <- matrix(1:10, nrow = 2)
y <- matrix(11:20, nrow = 2)
x
y
```

A matrix: 2 × 5 of
type int

1	3	5	7	9
2	4	6	8	10

A matrix: 2 × 5 of type
int

11	13	15	17	19
12	14	16	18	20

In [15]:

```r
rbind(x, y)  # по строкам
cbind(x, y)  # по столбцам
```

A matrix: 4 × 5 of type
int

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

A matrix: 2 × 10 of type int

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

Упражнение. Создайте матрицу $\begin{pmatrix} 1 & 4 & 9 \\ 16 & 25 & 36 \\ 49 & 64 & 81 \end{pmatrix}$. Сложите верхнюю левую матрицу 2x2 с матрицей, образованной 1 и 3 строками и 2 и 3 столбцами. Эти две матрицы также сконкатенируйте разными способами.

In [16]:

```
A <- matrix((1:9)^2, nrow = 3, byrow = TRUE)
A
```

A matrix: 3 × 3
of type dbl

1	4	9
16	25	36
49	64	81

In [17]:

```
B <- A[1:2, 1:2]
C <- A[-2, -1]
B + C
```

A matrix: 2
× 2 of type
dbl

5	13
80	106

In [18]:

```
rbind(B, C)
cbind(B, C)
```

A matrix:
4 × 2 of
type dbl

1	4
16	25
4	9
64	81

A matrix: 2 × 4 of
type dbl

1	4	4	9
16	25	64	81

Остальное

Логические операторы

In [19]:

```
1 < 2
1 > 2
TRUE
```

TRUE

FALSE

TRUE

Условный оператор

In [20]:

```
if (1 > 2) {
    print('true')
} else {
    print('false')
}
```

```
[1] "false"
```

Циклы

In [21]:

```
s <- 0
for (i in 1:10) {
    s <- s + i
}
print(s)
```

```
[1] 55
```

Функции

In [22]:

```
f <- function(x, y, z) {
    s <- x + y * z
    return(s)
}

f(1, 2, 3)
```

7

Вероятностные распределения

Distribution	R name	additional arguments
beta	`beta`	`shape1, shape2, ncp`
binomial	`binom`	`size, prob`
Cauchy	`cauchy`	`location, scale`
chi-squared	`chisq`	`df, ncp`
exponential	`exp`	`rate`
F	`f`	`df1, df2, ncp`
gamma	`gamma`	`shape, scale`
geometric	`geom`	`prob`
hypergeometric	`hyper`	`m, n, k`
log-normal	`lnorm`	`meanlog, sdlog`
logistic	`logis`	`location, scale`
negative binomial	`nbinom`	`size, prob`
normal	`norm`	`mean, sd`
Poisson	`pois`	`lambda`
signed rank	`signrank`	`n`
Student's t	`t`	`df, ncp`
uniform	`unif`	`min, max`
Weibull	`weibull`	`shape, scale`
Wilcoxon	`wilcox`	`m, n`

Как использовать? Пусть *name* - имя распределения в R.

- `d` *name* --- плотность (pdf)
- `p` *name* --- функция распределения (cdf)
- `q` *name* --- квантили (ppf)
- `r` *name* --- генерация выборки (rvs)

Другие параметры:

- `log` и `log.p` --- вычисления логарифмах
- `lower.tail` --- левостороннее накопление

Например, `p` *name* `(x, ..., lower.tail = FALSE, log.p = TRUE)` вернет $\log(1 - F(x))$, где $F(x)$ --- функция распределения.

In [23]:

```
dnorm
pnorm
qnorm
rnorm
```

```
function (x, mean = 0, sd = 1, log = FALSE)
.Call(C_dnorm, x, mean, sd, log)

function (q, mean = 0, sd = 1, lower.tail = TRUE, log.p = FALSE)
.Call(C_pnorm, q, mean, sd, lower.tail, log.p)

function (p, mean = 0, sd = 1, lower.tail = TRUE, log.p = FALSE)
.Call(C_qnorm, p, mean, sd, lower.tail, log.p)

function (n, mean = 0, sd = 1)
.Call(C_rnorm, n, mean, sd)
```

In [24]:

```
x <- rnorm(n = 10)
x
```

-2.14678563081019 -0.99796299360256 -1.54369202108689 0.463497539613111
-2.14812142745174 0.210975388261784 1.10609475422221 -0.919236184128946
0.175688956508896 -0.630129024762095

In [25]:

```
dnorm(0)
dnorm(0, mean = 1)
dnorm(x)
```

0.398942280401433

0.241970724519143

0.0398241086330779 0.242463619750895 0.121185716713721 0.358311156508065
0.0397100344323835 0.390161766316671 0.216392511855597 0.261469898157721
0.392832550554691 0.327106384174176

In [26]:

```
pnorm(0)
pnorm(1)
pnorm(1, lower.tail = FALSE)
```

0.5

0.841344746068543

0.158655253931457

In [27]:

```
qnorm(0.5)
qnorm(0.9)
```

0

1.2815515655446

In [28]:

```
dnorm(50)
dnorm(50, log = TRUE)
```

0

-1250.9189385332

Другое

In [29]:

```
sample(1:5, 20, replace=TRUE)  # выборка размера 20 из 1:5
sample(1:50, 20)  # без повторений
sample(1:10)  # перемешивание
```

3 2 5 2 4 5 4 1 3 2 4 4 3 2 2 1 3 3 5 4

39 46 6 4 40 11 23 31 5 24 36 38 16 32 47 44 28 12 41 30

4 1 9 6 5 8 10 7 3 2

Многомерное нормальное распределение

In [30]:

```
#install.packages('MASS')
library('MASS')

mu <- c(0, 0)
Sigma <- matrix(c(3, 2, 2, 3), nrow = 2)
mvrnorm(n = 10, mu, Sigma)
```

A matrix: 10 × 2 of type dbl

-1.5708994	-2.5220140
-1.6675433	-1.9674761
-2.4882015	-2.2244520
-1.4189301	-0.9577702
-0.1703379	1.2914930
-2.6990265	-5.4013001
-0.8877016	-1.9148014
2.0683134	1.1738751
-1.9835552	-1.6939353
3.7127874	0.7409479

Упражнение. Сгенерировать выборку размера 10 из равномерного распределения на отрезке [0, 5]. Посчитать по ней логарифмическую функцию правдоподобия для модели $Exp(2)$.

In [31]:

```
sample <- runif(10, min = 0, max = 5)
loglikelihood <- sum(dexp(sample, rate = 2, log = TRUE))
loglikelihood
```

-34.1560548012621

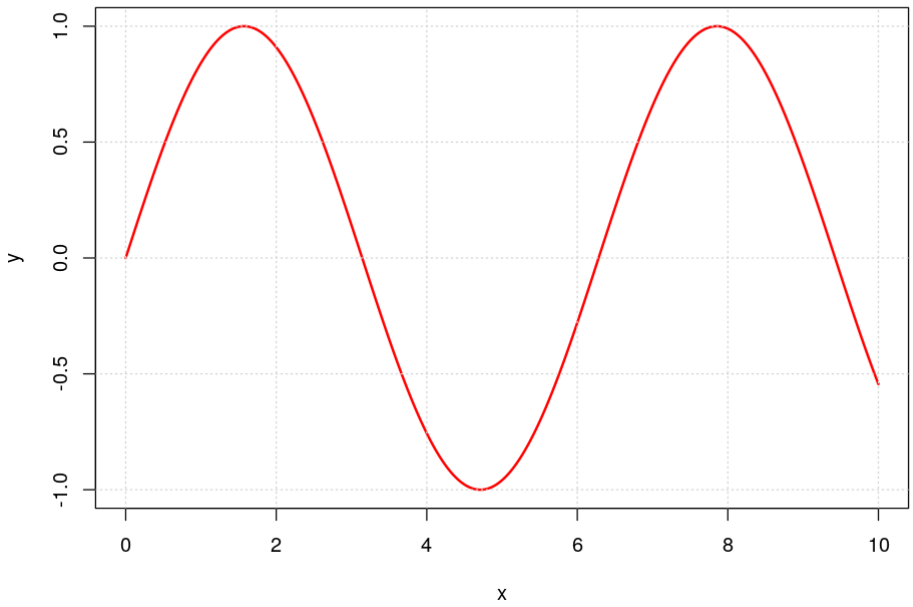
Графики

In [32]:

```
options(repr.plot.width = 8, repr.plot.height = 6)
```

In [33]:

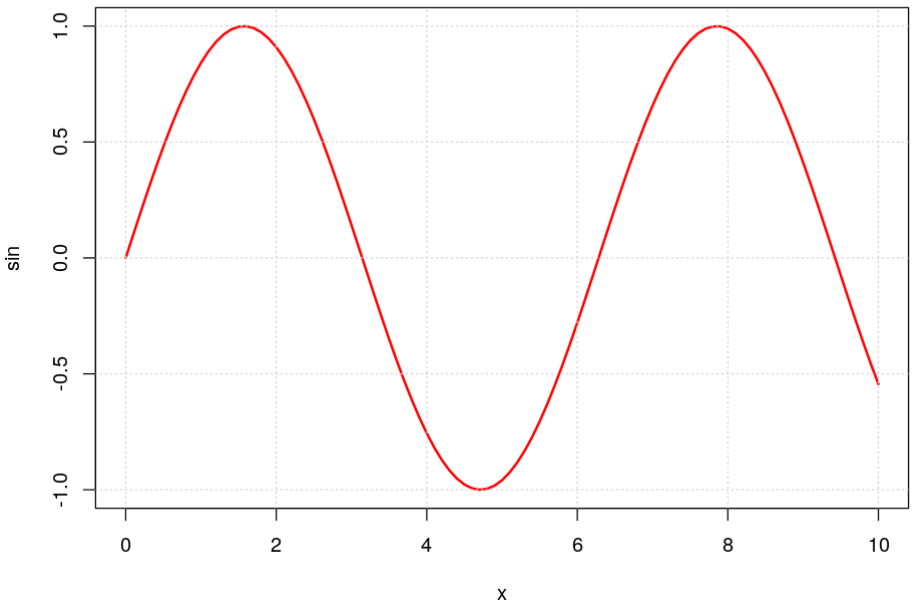
```
x <- seq(from = 0, to = 10, by = 0.01)
y <- sin(x)

plot(x, y, type = 'l', col = 'red', lwd = 2)
grid()
```



Проще:

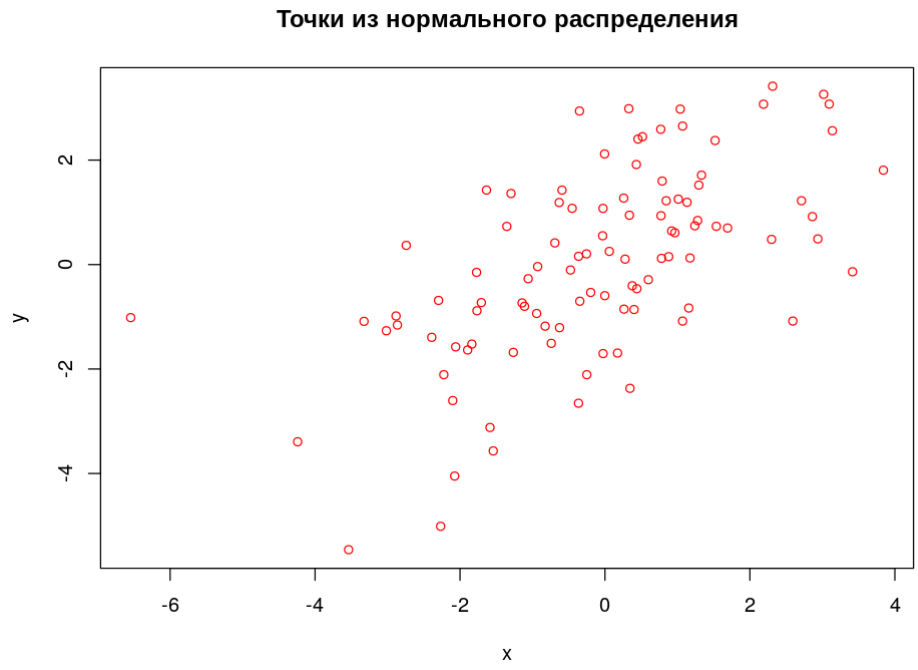
In [34]:

```
plot(sin, 0, 10, type = 'l', col = 'red', lwd = 2)
grid()
```



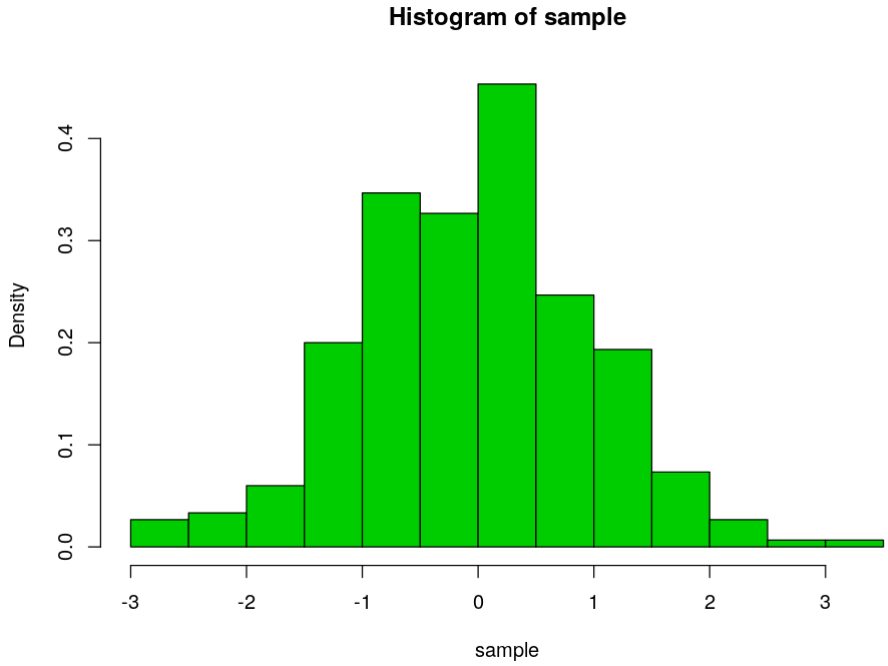
In [35]:

```
sample <- mvrnorm(n = 100, mu, Sigma = Sigma)
plot(sample[,1], sample[,2], cex = 0.9, col = 'red',
     xlab = 'x', ylab = 'y',
     main = 'Точки из нормального распределения')
```



In [36]:

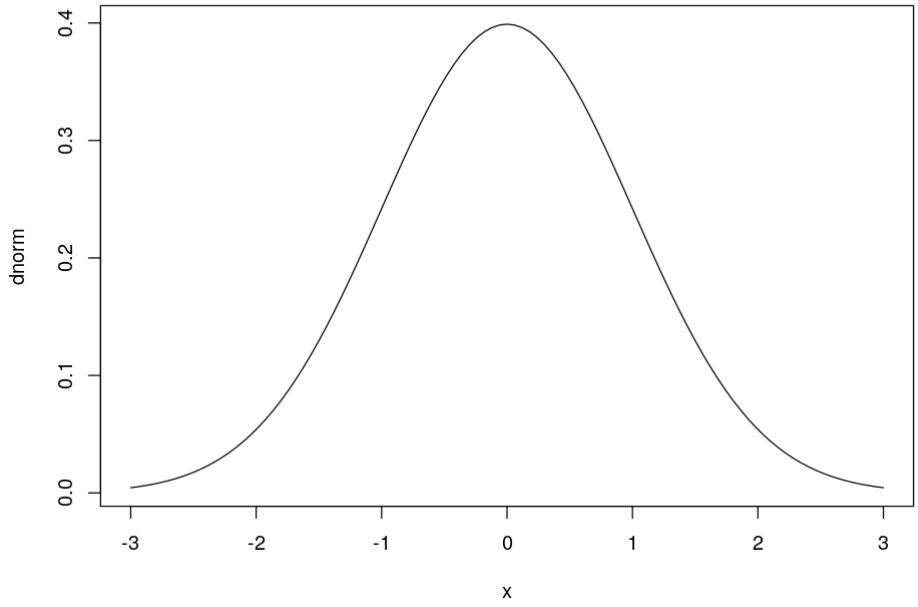
```
sample <- rnorm(n = 300)
hist(sample, freq = FALSE, col = 3, nclass = 10)
```



Упражнение. Построить график плотности нормального распределения на отрезке $[-3, 3]$.

In [37]:

```
1 plot(dnorm, -3, 3)
```



Прикладная статистика и анализ данных, 2019

Никита Волков

https://mipt-stats.gitlab.io/ (https://mipt-stats.gitlab.io/)