



Тема 1: Основы R



Содержание

- Преимущества R
- Список литературы
- Обзор возможностей R
- Интерфейс R
- Рабочая среда R
- Помощь R
- Пакеты R
- Сохранение результатов
- Использование результатов



Преимущества R

- Бесплатное программное обеспечение
- Доступно на Windows, Unix, Mac OS
- Широкие возможности проведения статистического анализа (в т.ч. методы, недоступные в других пакетах)
- Интуитивно понятный объектно-ориентированный язык высокого уровня
- Широкие возможности графической визуализации результатов анализа



Список литературы

Each of the following tutorials are in PDF format.

- P. Kuhnert & B. Venables, [An Introduction to R: Software for Statistical Modeling & Computing](#)
- J.H. Maindonald, [Using R for Data Analysis and Graphics](#)
- B. Muenchen, [R for SAS and SPSS Users](#)
- W.J. Owen, [The R Guide](#)
- D. Rossiter, [Introduction to the R Project for Statistical Computing for Use at the ITC](#)
- W.N. Venables & D. M. Smith, [An Introduction to R](#)



Список литературы

- Paul Geissler's [excellent R tutorial](#)
- [Excellent Tutorials by David Rossitier](#)
- [Excellent tutorial an nearly every aspect of R](#) (c/o Rob Kabacoff)
- [Introduction to R by Vincent Zoonekynd](#)
- [R Cookbook](#)
- [The Art of R](#)



Список литературы

- R Concepts and Data Types
presentation by Deepayan Sarkar
- The R Wiki
- An Introduction to R
- A Handbook of Statistical Analyses
Using R (Brian S. Everitt and Torsten
Hothorn)



Обзор возможностей R

Можно вводить команды одну за другой в командной строке (`>`) или запустить последовательность команд из файла-источника

Язык R содержит огромное количество различных типов данных, включая векторы (числовые, строковые, логические), матрицы, блоки данных и списки (`vector`, `matrix`, `data.frame`, `list`)

Для выхода из R достаточно ввести команду `>q()`



Обзор возможностей R

Гибкость языка R обеспечивается посредством встроенных и пользовательских функций. Во время работы с R все пользовательские данные хранятся в памяти программы

Базовые функции доступны по умолчанию. Другие функции содержатся в особых статистических пакетах и могут быть загружены во время работы

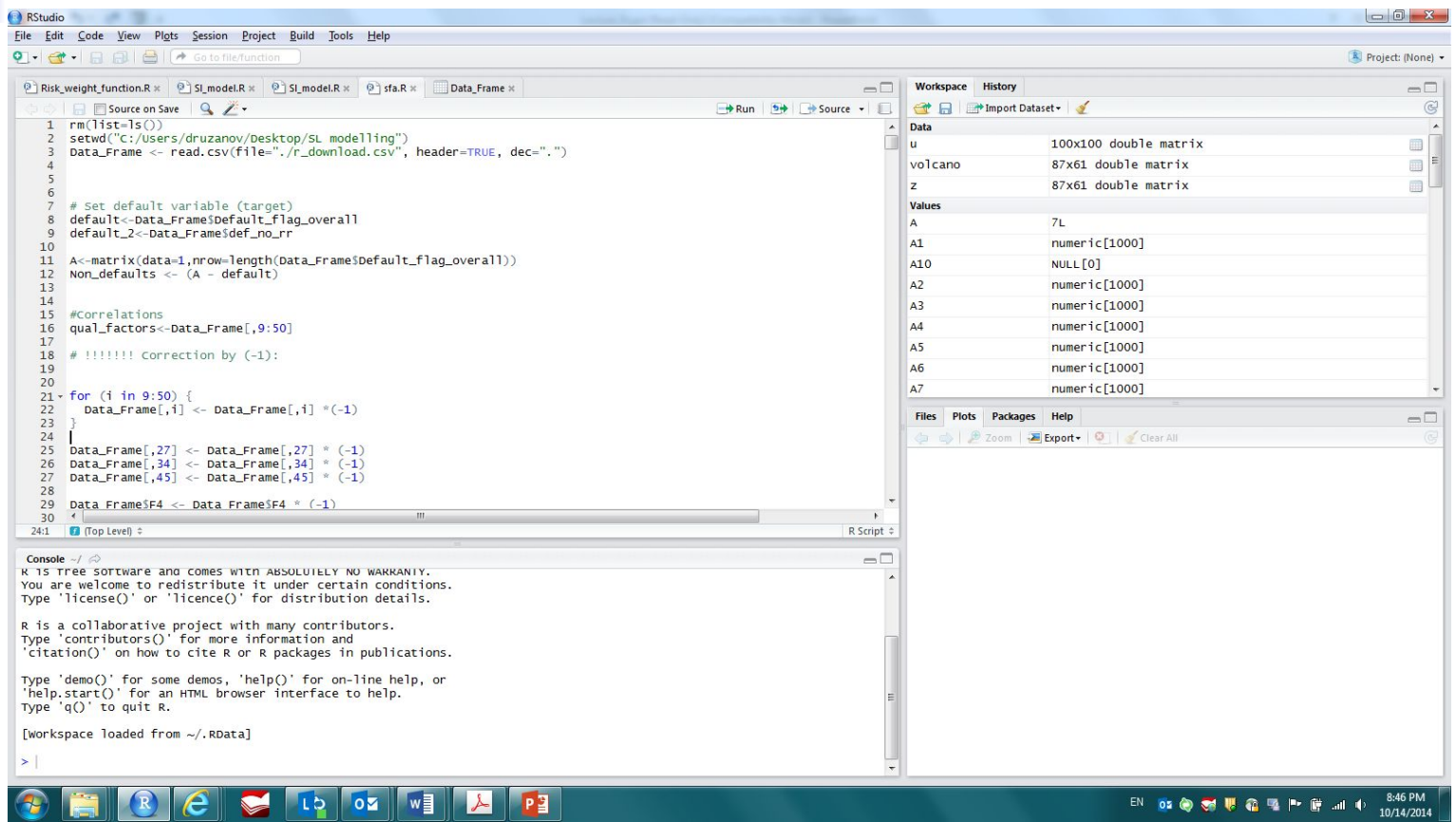


Обзор возможностей R

Основным навыком программирования в R является умение использовать встроенную справочную систему.

Фундаментальным свойством языка R является тот факт, что выходные данные одного объекта могут использоваться в качестве входных данных для другого.

Интерфейс R





Введение в R

- Результаты вычислений могут храниться в различных объектах с использованием операторов присвоения:
 - Стрелка (\leftarrow)
 - Знак равенства ($=$).



Введение в R

- Объекты R могут использоваться в последовательных расчетах. Для того, чтобы обратиться к объекту, необходимо напечатать в рабочей области имя объекта
- Требования к имени объекта
 - Имя объекта не может содержать символы `!`, `+`, `-`, `#`
 - Точка (`.`) и подстрочное подчеркивание (`_`) разрешены
 - Имя объекта может содержать число, но не может начинаться с числа
 - Язык R чувствителен к регистру



Пример

```
> # An example
> x <- c(1:10)
> x[(x>8) | (x<5)]
> # yields 1 2 3 4 9 10
> # How it works
> x <- c(1:10)
> X
> 1 2 3 4 5 6 7 8 9 10
> x > 8
> F F F F F F F T T
> x < 5
> T T T T F F F F F
> x > 8 | x < 5
> T T T T F F F T T
> x[c(T,T,T,T,F,F,F,F,T,T)]
> 1 2 3 4 9 10
```



Введение в R

- Для вывода списка всех объектов, доступных в текущей сессии R, используется функция `ls()`
`> ls()`
`[1] "x" "y"`
- Для того, чтобы обратиться к коду функции, достаточно ввести ее имя в командной строке
- Каждая из функций имеет аргументы, которые могут быть просмотрены при помощи клавиши `tab`

```
> ls(pattern="x")  
[1] "x" "x2"
```



Введение в R

- Если переменной присвоить некоторое значение, то ее предыдущее значение (в случае наличия такового) будет стерто без каких-либо уведомлений
- Для удаления объектов из рабочей среды используется функция `rm`
 - ▢ `rm(x, x2)`
- Удалить все объекты можно при помощи команды
 - ▢ `rm(list=ls())`
- Основная функция, отвечающая за построение графиков – `plot`

```
z2 <- c(1,2,3,4,5,6)
z3 <- c(6,8,3,5,7,1)
plot(z2,z3)
title("My first scatterplot")
```



Введение в R

R чувствителен к
регистру

MODEL, Model, model –
различные объекты



Введение в R

```
> x = sin(9)/75
> y = log(x) + x^2
> x
[1] 0.005494913
> y
[1] -5.203902
> m <- matrix(c(1,2,4,1), ncol=2)
> m
> [,1] [,2]
[1,] 1 4
[2,] 2 1
> solve(m)
[,1] [,2]
[1,] -0.1428571 0.5714286
[2,] 0.2857143 -0.1428571
```



Рабочая среда R

Объекты, создаваемые в R, содержатся в памяти программы (рабочая среда). Объекты рабочей среды НЕ сохраняются после закрытия программы автоматически!

Требуется ручное сохранение



Рабочая среда R

- При работе в R можно изменить имя рабочей папки.

```
## just checking what the current working directory is  
getwd()  
##  
setwd("C:/Users/Desktop/VaR/Gmail")
```



Рабочая среда R

Команды можно вводить в интерактивном режиме в командной строке. Стрелки вверх и вниз позволяют перемещаться по истории команд

Различные проекты R необходимо хранить в различных физических папках



Рабочая среда R

#Просмотр и установка параметров рабочей сессии

`help(options) #`

`options() #` просмотреть текущие параметры

`options(digits=3) #` количество знаков после запятой

Работа с предыдущими командами

`history() #` вывести последние 25 команд

`history(max.show=Inf) #` вывести все команды



Рабочая среда R

- # Сохранить историю команд
`savehistory(file="myfile")` # default is ".Rhistory"
- # Загрузить историю команд
`loadhistory(file="myfile")` # default is ".Rhistory"



Помощь R

В языке программирования R имеется
встроенная система справки **R**

`help.start()` # Общая справка

`help(F)` # Справка по функции *F*

`?F` # То же

`example(F)` # Пример использования функции *F*



Встроенные базы R

R содержит встроенные базы данных, которые можно использовать для обучения

> data() # Загрузка всех доступных пакетов

> help(*datasetname*)



Пакеты R

- Одним из преимуществ R является легкость расширения дистрибутива. Система позволяет создавать новые процедуры и функции, а также подгружать новые пакеты ('R Package' или 'R library')
- Примеры доступных пакетов:
 - Оптимизация портфеля ценных бумаг
 - Анализ временных рядов
 - Нейронные сети
 - GARCH-модели



Сохранение изображений

Для сохранения изображений R имеет особые функции. Для сохранения графика в терминале используется **dev.off()**

Function	Output to
<code>pdf("mygraph.pdf")</code>	pdf file
<code>win.metafile("mygraph.wmf")</code>	windows metafile
<code>png("mygraph.png")</code>	png file
<code>jpeg("mygraph.jpg")</code>	jpeg file
<code>bmp("mygraph.bmp")</code>	bmp file
<code>postscript("mygraph.ps")</code>	postscript file



Сохранение изображений

```
# example - output graph to jpeg file  
jpeg("c:/mygraphs/myplot.jpg")  
plot(x)  
dev.off()
```



Использование результатов модели

Язык программирования R позволяет сохранять результаты анализа и использовать их как входной параметр для дальнейшего исследования

Пример

```
lm(mpg~wt, data=mtcars)
```

Пример рассчитывает параметры линейной регрессии для переменных mpg и wt. Результаты отображены на экране



Использование результатов модели

Пример

```
fit <- lm(mpg~wt, data=mtcars)
```

Создается объект `fit`, содержащий результаты построения линейной регрессии

`str(fit)` # просмотр свойств объекта (его структуры)

При помощи формулы `fit <- lm(mpg~wt, data=mtcars)`

Был создан список (объект класса `list`), содержащий информацию о результатах моделирования (предсказанные значения, остатки, коэффициенты модели)



Использование результатов модели

```
# график остатков и прогнозных значений)  
plot(fit$residuals, fit$fitted.values)
```

Для просмотра выходных данных достаточно открыть по ней справку: **help(lm)**.

Многие функции допускают применение функции `plot` ко всему объекту:

```
# produce diagnostic plots  
plot(fit)
```



Тема 2: Входные данные



Содержание

- Типы данных
- Импорт данных
- Ввод с клавиатуры
- Ввод баз данных в программную среду
- Просмотр данных
- Пропущенные значения
- Дата и время



Типы данных

Данные в **R** могут быть представлены в виде большого числа типов данных (скаляры, векторы, матрицы, базы данных, списки).



Векторы

```
a <- c(1,2,5.3,6,-2,4) # числовой вектор
```

```
b <- c("one","two","three") # строковый вектор
```

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

```
#логический вектор
```

Выбор отдельных элементов вектора

```
a[c(2,4)] # 2-ой и 4-ый элементы вектора a
```



Матрицы

Все колонки матрицы должны быть одного типа и одной и той же длины

Общая форма:

```
mymatrix <- matrix(vector, nrow=r, ncol=c,  
  byrow=FALSE,dimnames=list(char_vector_rownames  
  , char_vector_colnames))
```

byrow=TRUE означает, что матрица заполняется по строкам

byrow=FALSE означает, что матрица заполняется по столбцам

dimnames – опционально: имена строк и столбцов



Матрицы

```
# создание числовой матрицы 5 x 4
y<-matrix(1:20, nrow=5,ncol=4)

# другой пример
cells <- c(1,26,24,68)
rnames <- c("R1", "R2")
cnames <- c("C1", "C2")
mymatrix <- matrix(cells, nrow=2, ncol=2,
  byrow=TRUE, dimnames=list(rnames, cnames))

#Определение отдельных элементов матрицы
x[,4] # 4-ая колонка матрицы
x[3,] # 3-ая колонка матрицы
x[2:4,1:3] # строки 2,3,4, столбцы 1,2,3
```



Массивы

Массивы имеют структуру, схожую с матрицами, но могут иметь более 2 измерений

help(array)



Таблицы данных

Таблица данных по структуре напоминает расширенную матрицу, однако данные в разных колонках могут быть различного типа.

```
d <- c(1,2,3,4)
e <- c("red", "white", "red", NA)
f <- c(TRUE,TRUE,TRUE,FALSE)
mydata <- data.frame(d,e,f)
names(mydata) <- c("ID","Color","Passed")
#имена переменных
```



Таблицы данных

Существует множество способов идентификации элементов таблицы данных

`myframe[3:5]` # колонки 3,4,5

`myframe[c("ID","Age")]` # колонки ID и Age

`myframe$X1` # переменная x1



Списки

Список – нумерованная последовательность объектов (компонентов списка). Список позволяет собрать множество различных объектов (не обязательно одного и того же класса)

```
# Пример списка из 4 компонентов  
w <- list(name="Fred", mynumbers=a,  
          mymatrix=y, age=5.3)
```

```
# Пример списка из двух списков  
v <- c(list1,list2)
```




Списки

Компоненты списка определяются с
использованием следующих знаков: `[]`
`mylist[[2]]` # 2-ой элемент списка



Факторы

Переменную, принимающую конечное число значений, можно преобразовать в фактор. Объект типа «Фактор» хранит возможные значения переменной в виде вектора.

```
# variable gender with 20 "male" entries and  
# 30 "female" entries  
gender <- c(rep("male",20), rep("female", 30))  
gender <- factor(gender)  
summary(gender)
```



Полезные функции

`length(object)` # Число элементов в объекте

`str(object)` # Структура объекта

`class(object)` # Класс объекта

`names(object)` # Имена элементов объекта

`c(object,object,...)` # соединить объекты в вектор

`cbind(object, object, ...)` # объединить объекты как столбцы

`rbind(object, object, ...)` # объединить объекты как строки

`ls()` # Показать список текущих объектов

`rm(object)` # Удалить объект



Импорт данных (.csv)

Первая строка должна содержать имена переменных

```
mydata <- read.table("c:/mydata.csv",  
header=TRUE, sep=";", row.names="id")
```



Ввод данных с клавиатуры

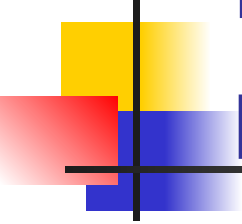
Для создания данных в интерактивном режиме используются следующие функции

```
age <- c(25, 30, 56)
```

```
gender <- c("male", "female", "male")
```

```
weight <- c(160, 110, 220)
```

```
mydata <- data.frame(age,gender,weight)
```



Ввод данных с клавиатуры

```
# Ввод данных в режиме редактора  
mydata <- data.frame(age=numeric(0),  
gender=character(0), weight=numeric(0))  
mydata <- edit(mydata)
```



Просмотр данных

**Существуют различные способы просмотра
имеющихся данных**

Просмотр объектов в рабочей среде
`ls()`

Просмотр переменных объекта
`names(mydata)`

Просмотр структуры объекта
`str(mydata)`

Размерность объекта
`dim(object)`



Просмотр данных

**Существуют различные способы просмотра
имеющихся данных**

Просмотр класса объекта
`class(object)`

Вывод на экран объекта
`mydata`

Вывод первые 10 строк объекта
`head(mydata, n=10)`

Вывод первых 10 строк объекта
`tail(mydata, n=5)`



Пропущенные значения

В **R** пропущенные значения обозначаются символом **NA**. Невозможные значения переменных (например, в результате деления на ноль) обозначаются символом **NaN** (not a number).

Тестирование переменной на пропущенные значения

`is.na(x)` # Возвращает TRUE, если x – пропущенное значение

```
y <- c(1,2,3,NA)
```

```
is.na(y) # возвращает вектор (F F F T)
```



Пропущенные значения

Замена значений переменной на пропущенные

замена числа 99 на пропущенные значения в строке
v1

```
mydata[mydata$v1==99,"v1"] <- NA
```

Исключение пропущенных значений из анализа

Arithmetic functions on missing values yield missing values.

```
x <- c(1,2,NA,3)
```

```
mean(x)          # Результат: NA
```

```
mean(x, na.rm=TRUE) # Результат: 2
```



Пропущенные значения

Функция **complete.cases** возвращает логический вектор, характеризующий заполненные наблюдения

список строк, которые содержат пропущенные значения

```
mydata[!complete.cases(mydata),]
```

Функция **na.omit()** возвращает объект с поэлементным удалением пропущенных значений

Создание таблицы данных без пропущенных значений

```
newdata <- na.omit(mydata)
```



Дата и время

Даты представлены в R как число дней, прошедших с 1970-01-01 (в случае обратного отсчета используются отрицательные значения)

функция `as.Date()` для преобразования формата к дате

```
mydates <- as.Date(c("2007-06-22", "2004-02-13"))
```

```
# число дней между 6/22/07 and 2/13/04  
days <- mydates[1] - mydates[2]
```

`Sys.Date()` – системное время

`Date()` – системная дата и время



Дата и время

Следующие символы могут использоваться для форматирования даты:

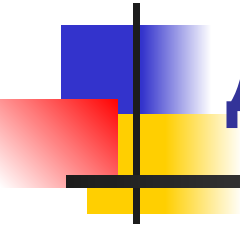
Symbol	Meaning	Example
%d	day as a number (0-31)	01-31
%a	abbreviated weekday	Mon
%A	unabbreviated weekday	Monday
%m	month (00-12)	00-12
%b	abbreviated month	Jan
%B	unabbreviated month	January
%y	2-digit year	07
%Y	4-digit year	2007



Дата и время

```
# print today's date  
today <- Sys.Date()  
format(today, format="%B %d %Y")  
"June 20 2007"
```

Тема 3: Манипулирование данными





Содержание

- Создание новой переменной
- Операторы
- Встроенные функции
- Функции контроля / циклы
- Пользовательские функции
- Сортировка
- Объединение
- Преобразования типов



Создание переменных

- Для создания новых переменных используется оператор `<-`.

#Три примера эквивалентных расчетов

```
mydata$sum <- mydata$x1 + mydata$x2  
mydata$mean <- (mydata$x1 + mydata$x2)/2
```

```
attach(mydata)  
mydata$sum <- x1 + x2  
mydata$mean <- (x1 + x2)/2  
detach(mydata)
```

- ```
mydata <- transform(mydata,
 sum = x1 + x2,
 mean = (x1 + x2)/2
)
```



# Создание переменных

---

## Создание категорий

```
Создание новых категорий
mydata$agecat <- ifelse(mydata$age > 70,
 c("older"), c("younger"))
Создание трех возрастных категорий
attach(mydata)
mydata$agecat[age > 75] <- "Elder"
mydata$agecat[age > 45 & age <= 75] <-
 "Middle Aged"
mydata$agecat[age <= 45] <- "Young"
detach(mydata)
```



# Арифметические операции

---

| Оператор       | Описание                                    |
|----------------|---------------------------------------------|
| +              | суммирование                                |
| -              | разность                                    |
| *              | умножение                                   |
| /              | деление                                     |
| <b>^ or **</b> | возведение в степень                        |
| <b>x % y</b>   | остаток от деления ( $5 \% 2 = 1$ )         |
| <b>x %/ y</b>  | целочисленное деление<br>( $5 \% / 2 = 2$ ) |



# Логические операторы

---

| Оператор  | Описание                     |
|-----------|------------------------------|
| <         | меньше                       |
| <=        | меньше или равно             |
| >         | больше                       |
| >=        | больше или равно             |
| ==        | в точности равно             |
| !=        | не равно                     |
| !x        | не x                         |
| x   y     | x OR y                       |
| x & y     | x AND y                      |
| isTRUE(x) | тестирование x (ИСТИНА/ЛОЖЬ) |



# Структуры контроля/циклы

---

- В **R** реализованы стандартные циклы, которые должны быть заключены в скобки `{}`
- С точки зрения производительности лучше использовать встроенные функции, нежели циклы, когда это возможно



# Контрольные структуры

---

- **if-else**
  - `if (cond) expr`
  - `if (cond) expr1 else expr2`
- **for**
  - `for (var in seq) expr`
- **while**
  - `while (cond) expr`
- **switch**
  - `switch(expr, ...)`
- **ifelse**
  - `ifelse(test,yes,no)`



# Контрольные структуры

---

- # Транспонирование матрицы  
# Альтернативы встроенной функции t()  
mytrans <- function(x) {  
 if (!is.matrix(x)) {  
 warning("argument is not a matrix: returning NA")  
 return(NA\_real\_)  
 }  
 y <- matrix(1, nrow=ncol(x), ncol=nrow(x))  
 for (i in 1:nrow(x)) {  
 for (j in 1:ncol(x)) {  
 y[j,i] <- x[i,j]  
 }  
 }  
 return(y)  
}



# Контрольные структуры

---

- # Пример  
z <- matrix(1:10, nrow=5, ncol=2)  
tz <- mytrans(z)





# Встроенные функции R

---

Практически любая операция в R связана с применением функций.

Функции могут быть применены к **ЛЮБЫМ** объектам (в т. ч. список определенного типа, вектор, матрица, пр.)



# Числовые функции

| Функция                       | Описание                                                        |
|-------------------------------|-----------------------------------------------------------------|
| <b>abs(x)</b>                 | модуль числа                                                    |
| <b>sqrt(x)</b>                | квадратный корень                                               |
| <b>ceiling(x)</b>             | округление вверх $\text{ceiling}(3.475)=4$                      |
| <b>floor(x)</b>               | округление вниз $\text{floor}(3.475)=3$                         |
| <b>trunc(x)</b>               | целая часть числа $\text{trunc}(5.99)=5$                        |
| <b>round(x, digits=n)</b>     | округление $\text{round}(3.475, \text{digits}=2)=3.48$          |
| <b>cos(x), sin(x), tan(x)</b> | также $\text{acos}(x)$ , $\text{cosh}(x)$ , $\text{acosh}(x)$ , |
| <b>log(x)</b>                 | натуральный логарифм                                            |
| <b>log10(x)</b>               | десятичный логарифм                                             |
| <b>exp(x)</b>                 | $e^x$                                                           |

| Функция                                                                                                                          | Описание                                                                                                                                                                                                                                                        |
|----------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>dnorm(x)</b>                                                                                                                  | Normal distribution density (by default m=0 sd=1)<br># plot standard normal curve<br>x <- pretty(c(-3,3), 30)<br>y <- dnorm(x)<br>plot(x, y, type='l', xlab="Normal Deviate", ylab="Density", yaxs="i")                                                         |
| <b>pnorm(q)</b>                                                                                                                  | cumulative normal probability for q<br>(area under the normal curve to the right of q)<br>pnorm(1.96) is 0.975                                                                                                                                                  |
| <b>qnorm(p)</b>                                                                                                                  | normal quantile.<br>value at the p percentile of normal distribution<br>qnorm(.9) is 1.28 # 90th percentile                                                                                                                                                     |
| <b>rnorm(n, m=0,sd=1)</b>                                                                                                        | n random normal deviates with mean m<br>and standard deviation sd.<br>#50 random normal variates with mean=50, sd=10<br>x <- rnorm(50, m=50, sd=10)                                                                                                             |
| <b>dbinom(x, size, prob)</b><br><b>pbinom(q, size, prob)</b><br><b>qbinom(p, size, prob)</b><br><b>rbinom(n, size, prob)</b>     | binomial distribution where size is the sample size<br>and prob is the probability of a heads (pi)<br># prob of 0 to 5 heads of fair coin out of 10 flips<br>dbinom(0:5, 10, .5)<br># prob of 5 or less heads of fair coin out of 10 flips<br>pbinom(5, 10, .5) |
| <b>dpois(x, lamda)</b><br><b>ppois(q, lamda)</b><br><b>qpois(p, lamda)</b><br><b>rpois(n, lamda)</b>                             | poisson distribution with m=std=lamda<br>#probability of 0,1, or 2 events with lamda=4<br>dpois(0:2, 4)<br># probability of at least 3 events with lamda=4<br>1- ppois(2,4)                                                                                     |
| <b>dunif(x, min=0, max=1)</b><br><b>punif(q, min=0, max=1)</b><br><b>qunif(p, min=0, max=1)</b><br><b>runif(n, min=0, max=1)</b> | uniform distribution, follows the same pattern<br>as the normal distribution above.<br>#10 uniform random variates<br>x <- runif(10)                                                                                                                            |

| Function                                 | Description                                                                                                                                                                                       |
|------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>mean(x, trim=0, na.rm=FALSE)</b>      | mean of object x<br># trimmed mean, removing any missing values and<br># 5 percent of highest and lowest scores<br>mx <- mean(x,trim=.05,na.rm=TRUE)                                              |
| <b>sd(x)</b>                             | standard deviation of object(x). also look at var(x) for variance and mad(x) for median absolute deviation.                                                                                       |
| <b>median(x)</b>                         | median                                                                                                                                                                                            |
| <b>quantile(x, probs)</b>                | quantiles where x is the numeric vector whose quantiles are desired and probs is a numeric vector with probabilities in [0,1].<br># 30th and 84th percentiles of x<br>y <- quantile(x, c(.3,.84)) |
| <b>range(x)</b>                          | range                                                                                                                                                                                             |
| <b>sum(x)</b>                            | sum                                                                                                                                                                                               |
| <b>diff(x, lag=1)</b>                    | lagged differences, with lag indicating which lag to use                                                                                                                                          |
| <b>min(x)</b>                            | minimum                                                                                                                                                                                           |
| <b>max(x)</b>                            | maximum                                                                                                                                                                                           |
| <b>scale(x, center=TRUE, scale=TRUE)</b> | column center or standardize a matrix.                                                                                                                                                            |



# Другие полезные функции

---

| Функция                                            | Описание                                                                                                                 |
|----------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------|
| <b>seq</b> ( <i>from</i> , <i>to</i> , <i>by</i> ) | Создание последовательности чисел<br><code>indices &lt;- seq(1,10,2)</code><br><code>#indices is c(1, 3, 5, 7, 9)</code> |
| <b>rep</b> ( <i>x</i> , <i>ntimes</i> )            | Копирование объекта <i>x</i> <i>n</i> раз<br><code>y &lt;- rep(1:3, 2)</code><br><code># y is c(1, 2, 3, 1, 2, 3)</code> |
| <b>cut</b> ( <i>x</i> , <i>n</i> )                 | Трансформация непрерывной переменной на фактор с <i>n</i> уровнями<br><code>y &lt;- cut(x, 5)</code>                     |



# Сортировка

---

- Сортировка данных производится при помощи функции **order( )** function. По умолчанию сортировка проводится по возрастанию (ASCENDING).
- # Пример сортировки

```
data(mtcars)
sort by mpg
newdata = mtcars[order(mtcars$mpg),]
sort by mpg and cyl
newdata <- mtcars[order(mtcars$mpg, mtcars$cyl),]
sort by mpg (ascending) and cyl (descending)
newdata <- mtcars[order(mtcars$mpg, -mtcars$cyl),]
```



# Объединение

---

Для объединения двух таблиц данных используется функция **merge**. В большинстве случаев можно объединить две таблицы данных по одному или нескольким ключам (inner join).

# Объединение таблиц данных по ID

```
total <- merge(dataframeA,dataframeB,by="ID")
```

# Объединение таблиц данных по ID и стране

```
total <-
```

```
merge(dataframeA,dataframeB,by=c("ID","Country"))
```



# Объединение

---

## Добавление строк

Для добавление строк к таблице данных используется функция **rbind**. Таблицы данных должны иметь одинаковые переменные, но не обязательно одинаковую размерность

```
total <- rbind(dataframeA, dataframeB)
```





# Преобразования данных

---

- **Полезные функции:**

**is.numeric(), is.character(), is.vector(),  
is.matrix(), is.data.frame()  
as.numeric(), as.character(), as.vector(),  
as.matrix(), as.data.frame)**