Липецкий государственный технический университет

Кафедра прикладной математики

МЕТОДЫ ПРИКЛАДНОЙ СТАТИСТИКИ

Лекция 1

Что такое данные? Зачем и как их обрабатывать?

Составитель - Сысоев А.С., к.т.н., доцент

Липецк - 2021

Outline

1.1. Откуда берутся данные?

Способов получения данные много. Главные - эксперимент и наблюдение:

- **наблюдение** - такой способ получения данных, при котором воздействие наблюдателя на наблюдаемый объект сведено к минимуму;
- **эксперимент** включает наблюдение, но сначала на наблюдаемый объект оказывается заранее рассчитанное воздействие.

Задача: исследовать, чем питается какое-то редкое животное.

Наблюдение «в чистом виде» более или менее неосуществимо, поскольку всегда будет внесено какое-нибудь воздействие.

1.2. Генеральная совокупность и выборка

«Статистика знает всё...»
И. Ильф, Е. Петров «Двенадцать стульев»

Задача: предпочтения покупателями мороженого.

Всех продавцов не проконтролируешь, но ведь нескольких-то можно. Надо выбрать из общего множества несколько торговых точек (*как* выбирать - особая наука) и проконтролировать тамошние продажи силами самой фирмы или такими нанятыми людьми, которым можно доверять.

Самый главный вопрос: можно ли этот результат распространить на всю совокупность продаж? Можно, поскольку на основе теории вероятностей уже много лет назад была создана **теория выборочных исследований**. Ее-то и называют чаще всего математической статистикой, или просто **статистикой**.

Задача: сплошная перепись населения России 1897 г.

Процесс создания выборки может являться источником ошибок. Их принято называть **«ошибками репрезентативности».** Однако правильная организация выборки позволяет их избежать.

1.3. Как получить данные?

Два основных принципа составления выборки: повторность и рандомизация:

- **принцип повторностей** предполагает, что один и тот же эффект будет исследован несколько раз. Повторности должны быть независимы друг от друга (**задача**: лягушки).

Сколько надо собрать данных? Ответы: (1) чем больше, тем лучше и (2) 30. Считается, что выборки, меньшие 30, следует называть малыми, а большие - большими.

- **принцип рандомизации**: каждый объект должен иметь абсолютно те же самые шансы быть выбранным, что и все прочие объекты (**задача**: деревья).

1.4. Что ищут в данных?

Анализ данных необходим всегда, когда результат неочевиден, и часто даже тогда, когда он кажется очевидным.

Основные направления анализа данных:

- **общие характеристики для больших выборок** (центральная тенденция (точка, набор точек), разброс, ...);
- **сравнение разных выборок**. Сравнение данных при помощи *статистических тестов* позволяет выяснить, насколько велика вероятность, что различия между группами вызваны случайными причинами;
- **сведения о взаимосвязи**:
  - **соответствия**;
  - **корреляции**. Корреляции показывают силу взаимосвязи, но не могут определить ее направления;
  - **зависимости**. Можно измерить и силу, и направление, и оценить, насколько вероятно то, что они - результат случайных причин. Можно предсказать, как будет «вести» себя зависимая переменная в каких-нибудь до сих пор не опробованных условиях.
- **установление структуры** (многомерная статистика, DataMining, классификация, кластеризация, ...).

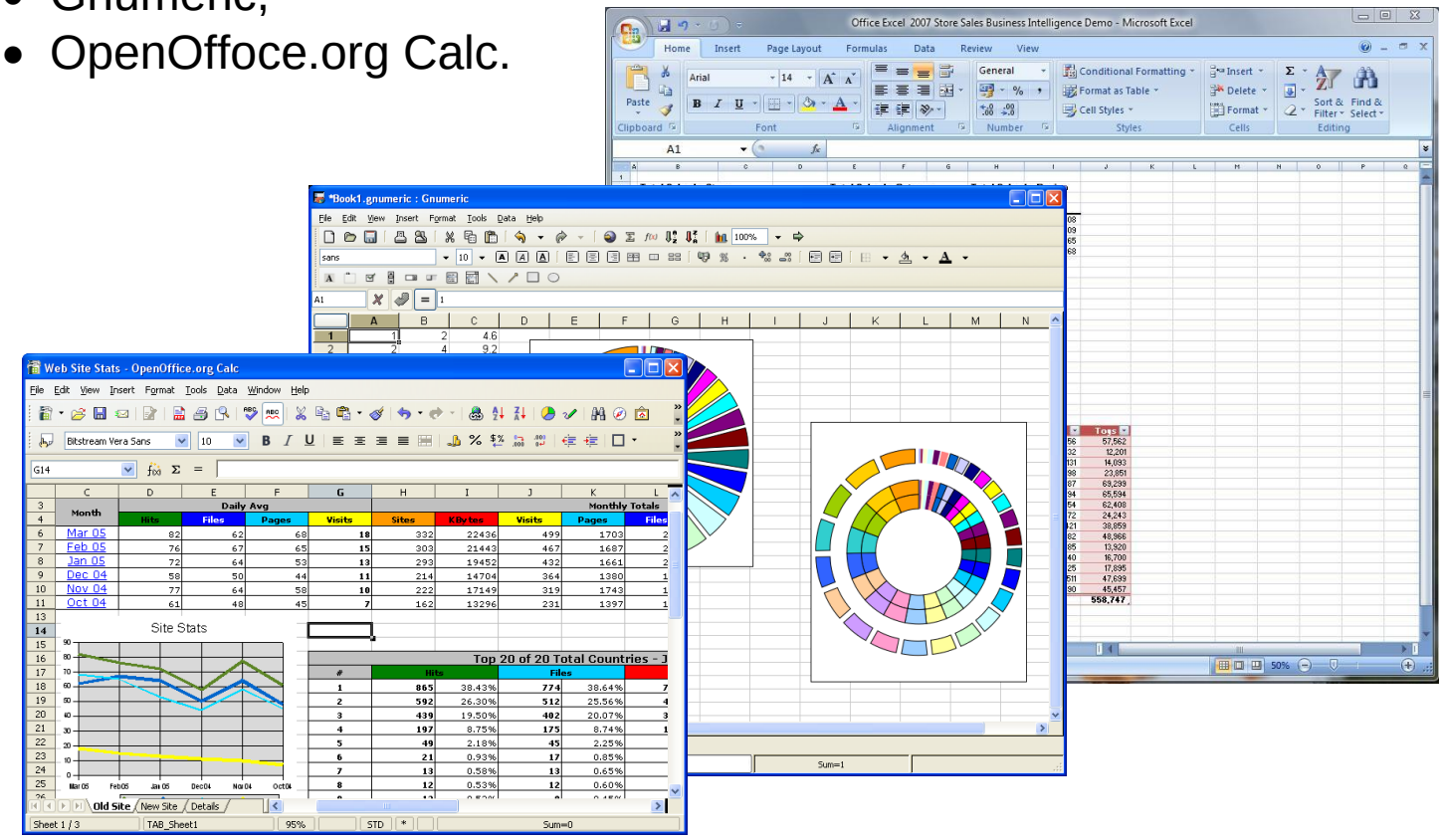
Другой подход - предсказательные и описательные методы.

1.5. Как обрабатывать данные?

1.5.1. Неспециализированные программы

Электронные таблицы, представляющие скудный статистический потенциал:

- Microsoft Office Excel,
- Gnumeric,
- OpenOffoce.org Calc.

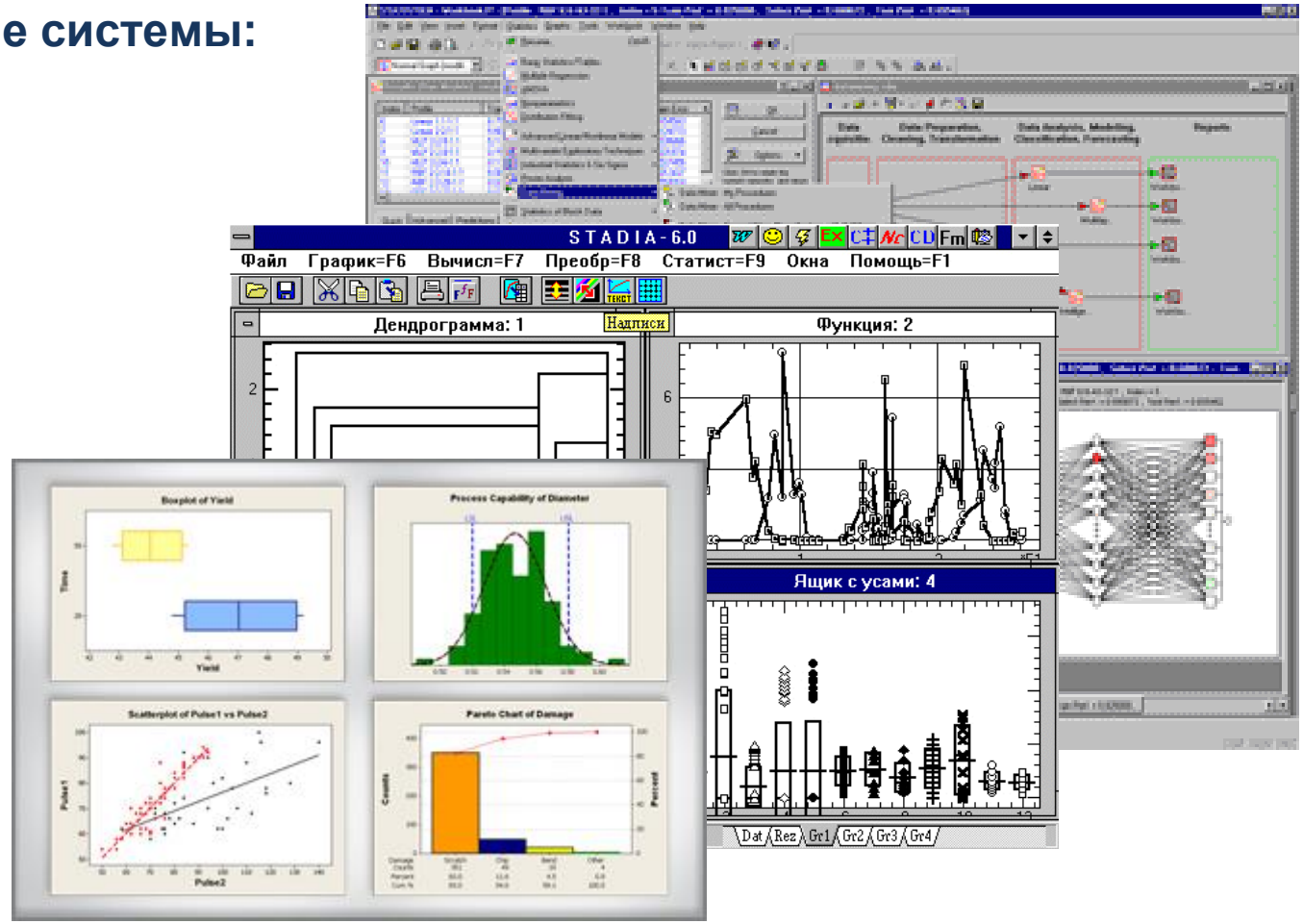


1.5. Как обрабатывать данные?

1.5.2. Специализированные программы

Оконно-кнопочные системы:

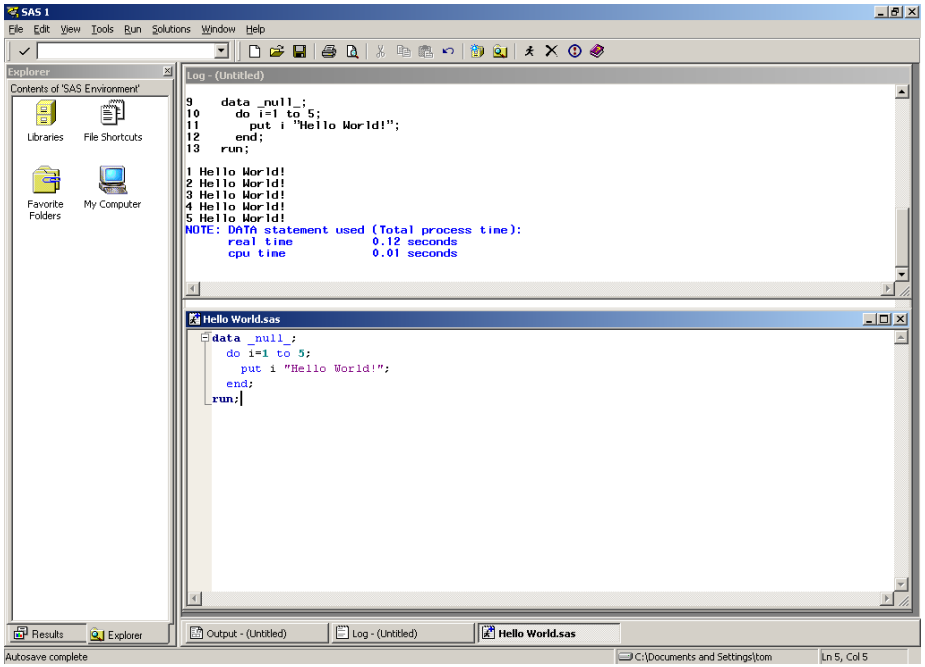
- STATISTICA,
- STADIA,
- SPSS,
- MiniTab



1.5.2. Специализированные программы

Статистические среды. Эта группа программ использует в основном интерфейс командной строки. Пользователь получает полный контроль над системой: он может комбинировать любые типы анализа, записывать процедуры в скрипты, которые можно запустить в любое время, модифицировать вывод графиков, сохранять их в любые графические форматы, легко писать расширения для системы.

Пример: SAS.



1.5.3. Из истории R и S

R - это среда для статистических расчетов. R задумывался как свободный аналог среды S-Plus, которая, в свою очередь, является коммерческой реализацией языка расчетов S. Язык S возник в 1976 году в компании Bell Labs и был назван, естественно, «по мотивам» языка C.

В августе 1993 г. двое молодых новозеландских ученых анонсировали свою новую разработку, которую они назвали R (буква «R» была выбрана просто потому, что она стоит перед «S», тут есть аналогия с языком программирования C, которому предшествовал язык B).

Ross Ihaka *Robert Gentleman*

1.5.4. Применение, преимущества и недостатки R

Область применения - от вычисления средних величин до вейвлет-преобразований и временных рядов.

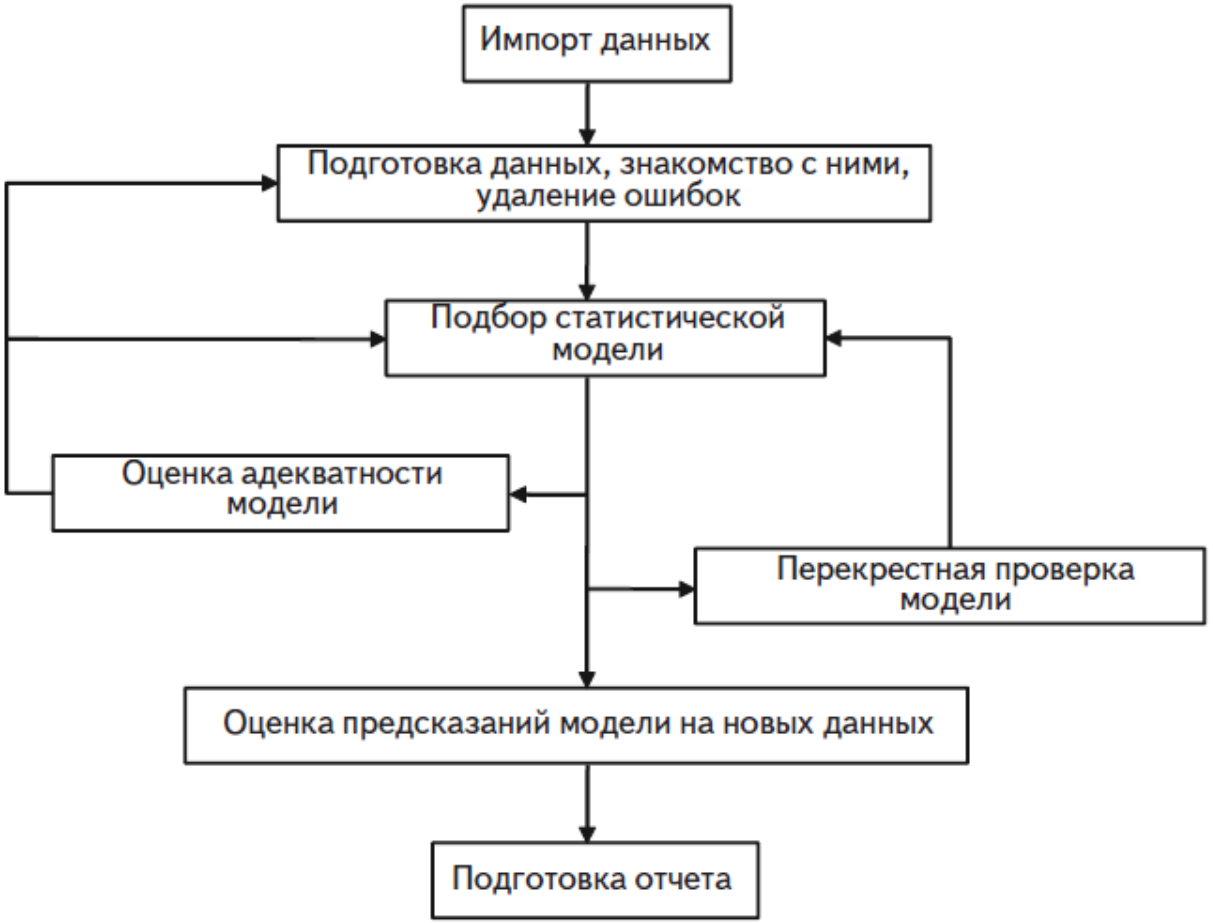
Преимущества:

- **гибкость** - позволяет создавать различные приложения;
- **свободный код** - возможность разобраться, как именно происходит анализ (при наличии ошибок - их исправление).

Недостатки:

- **трудность обучения пользователей** (множество команд, отсутствие меню). Выход - команда `help(`*`название функции`*`)` =)
- **относительная медлительность** - функции, использующие циклы, списки и большие таблицы, выполняются в десятки раз медленнее, чем в коммерческих пакетах.

1.6. Анализ данных. Что это и как его выполнять?



ЭТАПЫ ТИПИЧНОГО АНАЛИЗА ДАННЫХ

1.6. Анализ данных. Что это и как его выполнять?

Задача: изучить физическое развитие и собрали данные о возрасте и весе 10 младенцев первого года жизни. Получить распределение значений веса и их зависимостью от возраста.

Возраст (месяцы)	Вес (кг)	Возраст (месяцы)	Вес (кг)
01	4.4	09	7.3
03	5.3	03	6.0
05	7.2	09	10.4
02	5.2	12	10.2
11	8.5	03	6.1

```
> age <- c(1,3,5,2,11,9,3,9,12,3)
> weight <- c(4.4,5.3,7.2,5.2,8.5,7.3,
+ 6.0,10.4,10.2,6.1)
> mean(weight)
[1] 7.06
> sd(weight)
[1] 2.077498
> cor(age, weight)
[1] 0.9075655
> plot(age, weight)
> q()
```

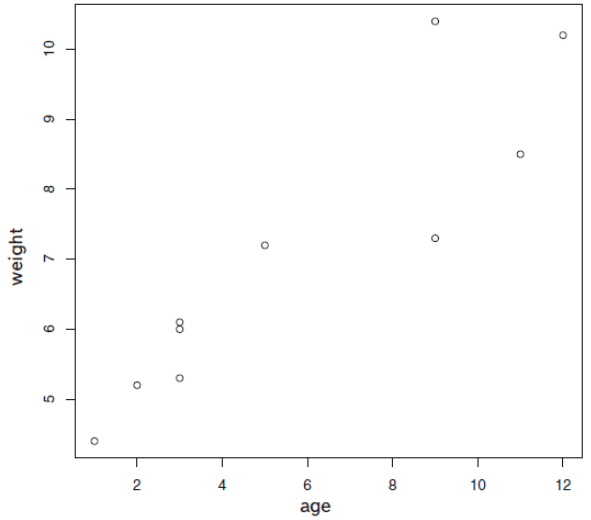


Диаграмма рассеяния веса младенцев

1.7. Рабочее пространство R. Пакеты.

Рабочее пространство – это текущая рабочая среда R в памяти компьютера, которая включает в себя любые созданные пользователем объекты.

Текущая рабочая директория – это та директория, где находятся файлы данных и куда по умолчанию сохраняются результаты.

Функция	Действие
`getwd()`	Вывести на экран название текущей рабочей директории
`setwd("моя_директория")`	Назначить *моя_директория* текущей рабочей директорией
`ls()`	Вывести на экран список объектов в текущем рабочем пространстве
`rm("список_объектов")`	Удалить один или несколько объектов
`help(options)`	Справка о возможных опциях
`options()`	Посмотреть или установить текущие опции
`history(#)`	Вывести на экран последние # команд (по умолчанию 25)

1.7. Рабочее пространство R. Пакеты.

Ввод

Функция `source("filename")` запускает скрипт. Если не прописан путь к файлу, подразумевается, что он находится в текущей рабочей директории.

Текстовый вывод

Функция `sink("имя_файла")` выводит все результаты в файл с названием `имя_файла`.

Графический вывод

Функция	Вывод (формат графического файла)
`pdf("filename.pdf")`	PDF
`win.metafile("filename.wmf")`	Windows metafile
`png("filename.png")`	PNG
`jpeg("filename.jpg")`	JPEG
`bmp("filename.bmp")`	BMP
`postscript("filename.ps")`	PostScript

1.7. Рабочее пространство R. Пакеты.

Пакеты – это собрания функций R, данных и скомпилированного программного кода в определенном формате.

Установка пакета

Для установки пакета используется команду `install.packages()`.Пакет нужно установить только один раз. Для обновления всех установленных пакетов используется команда `update.package()`. Для использования пакета в текущей сессии программы необходимо загрузить его при помощи команды library().

Функция `help(package="название_пакета")` выводит короткое описание этого пакета и алфавитный указатель всех входящих в него функций и наборов данных.

2.1. Типы данных в языке R

Все объекты данных в R можно разделить на следующие **классы** (т.е. типы объектов):

- **numeric** – объекты, к которым относятся целочисленные (integer) и действительные числа (double);
- **logical** – логические объекты, которые принимают только два значения: FALSE (F) и TRUE (T);
- **character** – символьные объекты (значения переменных задаются в двойных, либо одинарных кавычках).

```
is.numeric(<имя_объекта>)
as.integer(<имя>)
```

В R существует ряд специальных объектов:

- **Inf** – положительная или отрицательная бесконечность (обычно результат деления вещественного числа на 0);
- **NA** – "отсутствующее значение" (Not Available);
- **NaN** – "не число" (Not a Number).

Выражение (expression) языка R представляет собой сочетание таких элементов, как оператор присваивания, арифметические или логические операторы, имена объектов и имена функций. (=, <-, ->)

2.1. Типы данных в языке R

2.1.1. Векторы и матрицы

Вектор представляет собой поименованный одномерный объект, содержащий набор однотипных элементов (числовые, логические, либо текстовые значения – никакие их сочетания не допускаются). Для создания векторов небольшой длины в R используется *функция конкатенации*.

```
my.vector <- c(1, 2, 3, 4, 5)
my.vector
[1] 1 2 3 4 5
```

Альтернатива - `scan()`

Последовательность

```
S <- seq(1,7)
S
[1] 1 2 3 4 5 6 7
```

Повторные значения

```
Text <- rep("test", 5)
Text
[1] "test" "test" "test" "test" "test"
```

Упорядочивание элементов вектора

```
sort(z)      # по умолчанию decreasing = FALSE
[1] 0.3 0.5 0.6
sort(z, decreasing = TRUE)
[1] 0.6 0.5 0.3
```

2.1. Типы данных в языке R

2.1.1. Векторы и матрицы

Матрица представляет собой двумерный вектор.

```
my.mat <- matrix(seq(1, 16), nrow = 4, ncol = 4)
my.mat
     [,1] [,2] [,3] [,4]
[1,]    1    5    9   13
[2,]    2    6   10   14
[3,]    3    7   11   15
[4,]    4    8   12   16
```

!!! Заполнение матрицы происходит по столбцам

```
rownames(my.mat) <- c("A", "B", "C", "D")
```

Матрицу можно собрать также из нескольких векторов, используя функции `cbind()` или `rbind()`.

```
a <- c(1, 2, 3, 4)
b <- c(5, 6, 7, 8)
d <- c(9, 10, 11, 12)
e <- c(13, 14, 15, 16)
cbind(a, b, d, e)
rbind(a, b, d, e)
```

Транспонирование

```
t(my.mat)
```

2.1. Типы данных в языке R

2.1.2. Факторы

В статистике данные очень часто группируют в соответствии с тем или иным признаком, например, полом, социальным положением, стадией болезни, местом отбора проб и т.п. В R существует специальный класс векторов – **факторы (factors)**, которые предназначены для хранения кодов соответствующих уровней номинальных признаков.

Задача: в эксперименте по испытанию эффективности нового медицинского препарата было задействовано 10 пациентов-добровольцев, из которых шесть пациентов принимали новый препарат, а четверо остальных – плацебо (например, таблетки активированного угля). Для обозначения членов этих двух групп мы можем использовать коды 1 (препарат) и 0 (плацебо).

```
treatment <- c(1, 1, 1, 1, 1, 1, 0, 0, 0, 0)
treatment
[1] 1 1 1 1 1 1 0 0 0 0
treatment <- factor(treatment, levels = c(0, 1))
treatment
[1] 1 1 1 1 1 1 0 0 0 0
Levels: 0 1
levels(treatment) <- c("no", "yes")
treatment
[1] yes yes yes yes yes yes no no no no
Levels: no yes
```

2.1. Типы данных в языке R

2.1.3. Списки и таблицы

В отличие от вектора или матрицы, которые могут содержать данные только одного типа, в **список (list)** или **таблицу (data frame)** можно включать сочетания любых типов данных. Это позволяет эффективно, т.е. в одном объекте, хранить разнородную информацию.

```
vector1 <- c("A", "B", "C")
vector2 <- seq(1, 3, 0.5)
vector3 <- c(FALSE, TRUE)
my.list <- list(Text=vector1, Number=vector2, Logic=vector3)
my.list
$Text
[1] "A" "B" "C"
$Number
[1] 1.0 1.5 2.0 2.5 3.0
$Logic
[1] FALSE TRUE
```

Для выяснения структуры

```
str(my.list)
List of 3
$ Text : chr [1:3] "A" "B" "C"
$ Number: num [1:5] 1 1.5 2 2.5 3
$ Logic : logi [1:2] FALSE TRUE
```

2.1. Типы данных в языке R

2.1.3. Списки и таблицы

Таблица данных является частным случаем списка, в котором все компоненты-векторы имеют одинаковый размер). Таблицы данных – это основной класс объектов R, используемых для хранения данных.

```
city <- c("City1", "City1", "City2", "City2", "City3", "City3")
sex <- c("Male", "Female", "Male", "Female", "Male", "Female")
number <- c(12450, 10345, 5670, 5800, 25129, 26000)
CITY <- data.frame(City = city, Sex = sex, Number = number)
CITY
    City    Sex     Number
1  City1   Male     12450
2  City1 Female     10345
3  City2    Male     5670
4  City2 Female      5800
5  City3    Male    25129
6  City3 Female     26000
```

2.1. Типы данных в языке R

2.1.4. Импортирование данных в R

Особенности:

- В импортируемой таблице с данными не должно быть пустых ячеек. Если некоторые значения по тем или иным причинам отсутствуют, вместо них следует ввести NA.
- Импортируемую таблицу с данными рекомендуется преобразовать в простой текстовый файл с одним из допустимых расширений. На практике обычно используются файлы с расширением .txt, в которых значения переменных разделены знаками табуляции (tab-delimited files), а также файлы с расширением .csv (comma separated values), в которых значения переменных разделены запятыми или другим разделяющим символом.
- В качестве первой строки в импортируемой таблице рекомендуется ввести заголовки столбцов-переменных. Все последующие строки файла в качестве первого элемента могут содержать заголовки строк (если таковые предусмотрены), после которых следуют значения каждой из имеющихся в таблице переменных.

	Group	Variable1	Variable2	Variable3
Ivan	A	102	1.3	14
Vitaliy	A	98	1.4	11
Sergey	B	45	NA	8
Mikhail	B	50	3.2	6

2.1. Типы данных в языке R

2.1.4. Импортирование данных в R

Основной функцией для импортирования данных в рабочую среду R является `read.table()`.

Аргумент	Назначение
File	file = "C:/Temp/MyData.dat" file = "http://somesite.net/YourData.csv"
header	Служит для сообщения программе о наличии в загружаемом файле строки с заголовками столбцов. По умолчанию принимает значение FALSE. Если строка с заголовками столбцов имеется, этому аргументу следует присвоить значение TRUE.
row.names	Служит для указания номера столбца, в котором содержатся имена строк
Sep	sep = "" sep = ","
Dec	dec = "." dec = ","
Nrows	Выражается целым числом, указывающим количество строк, которое должно быть считано из загружаемой таблицы.
Skip	Выражается целым числом, указывающим количество строк в файле, которое должно быть пропущено перед началом импортирования.

```
chem <- read.table(file = file.choose(), header = TRUE, sep = ",")
```

2.2. Представление даты и времени; временные ряды

ФОРМАТЫ ПРЕДСТАВЛЕНИЯ ДАТЫ И ВРЕМЕНИ

Анализ данных, содержащих даты и время, может иногда сопровождаться рядом проблем:

- разные годы начинаются в разные дни недели;
- високосные годы имеют дополнительный день в феврале;
- американцы и европейцы по разному представляют даты (например, 8/9/2011);
- страны различаются по временным поясам и в ряде случаев применяют переход на "зимнее" и "летнее" время.

```
Sys.time()
[1] "2011-09-06 00:38:04 EEST"
```

ВЫЧИСЛЕНИЯ С ДАТАМИ И ВРЕМЕНЕМ

В R можно выполнять следующие типы операций с датами и временем:

- число + время;
- время – число;
- время1 – время2
- время1 "логический оператор" время2 (в качестве логического оператора могут использоваться ==, !=, <=, <, > или >=).

2.2. Представление даты и времени; временные ряды

ВЫЧИСЛЕНИЯ С ДАТАМИ И ВРЕМЕНЕМ (продолжение)

`proc.time()` - продолжительность вычислительного процесса

Задача: сколько потребуется времени, чтобы вычислить 10 000 значений arctg.

```
t1 <- proc.time()
for (x in 1:10000) y <- atan(x)
time.result <- proc.time() - t1
time.result["elapsed"]
elapsed
0.02
```

ВРЕМЕННЫЕ РЯДЫ

Пример: ежемесячные данные по рождаемости в г. Нью-Йорк, собранные в период с января 1946 г. по декабрь 1959 г. (A Little Book of R for Time Series)

```
birth <- scan("http://robjhyndman.com/tsdldata/data/nybirths.dat")
Read 168 items
```

Преобразовать данные во временной ряд

```
birth.ts <- ts(birth, start = c(1946, 1), frequency = 12)
```

2.3. Организация вычислений: функции, ветвления, циклы

Вызов функции и описание	Пример и результат
Арифметические функции	
abs(x) – модуль величины x	abs(-1) ⇒ 1
ceiling(x) – округление до целого в большую сторону	ceiling(9.435) ⇒ 10
floor(x) – округление до целого в меньшую сторону	floor(2.975) ⇒ 2
round(x, digits=n) – округление до указанного числа digits знаков после десятичной точки	round(5.475, 2) ⇒ 5.48
signif(x, digits=n) # округление до указанного числа digits значащих цифр	signif(3.475, 2) ⇒ 3.5
trunc(x) – округление до целого числа	trunc(4.99) ⇒ 4
exp(x) – e^x	exp(2.87) ⇒ 17.637
log(x) – логарифм натуральный x **log10**(x) – логарифм десятичный x	log(3.12) ⇒ 1.137 log(3.12) ⇒ 0.494
sqrt(x) # корень квадратный x	sqrt(2.12) ⇒ 1.456
cos(x) **sin**(x) **tan**(x) **acos**(x) **cosh**(x) **acosh**(x) – тригонометрические функции от x	cos(1.27*pi) ⇒ -0.661

2.3. Организация вычислений: функции, ветвления, циклы

Функции для работы с символьными типами данных	
grep(pattern,x,ignore.case=FALSE,fixed=FALSE) – возврат индекса первого найденного элемента pattern в x	grep("A",c("x","y","A", "z"),fixed=TRUE) ⇒ 3
substr(x, start=n1, stop=n2) – выбор или замена символов в строках символьного вектора x	substr("язык R", 2, 4) ⇒ "зык"
paste(..., sep="") – объединение символов или строк через значение разделителя sep	paste("x",1:3,sep="") ⇒ "x1" "x2" "x3"
strsplit(x, split) – разделяет элементы вектора по разделителям split	strsplit("абв","") ⇒ "а" "б" "в"
toupper(x) и **tolower**(x) – преобразуют буквы текстового вектора x в прописные и обратно	toupper("Мал") ⇒ "МАЛ" toupper("БАЛ") ⇒ "бал"

СОЗДАНИЕ СОБСТВЕННЫХ ФУНКЦИЙ

```
имя_функции <- function(arg1, arg2,...) {
группа_выражений
return(object) }
```

где имя_функции – имя создаваемой функции, arg1, arg2,... – формальные аргументы функции. Оператор return() нужен в случаях, когда группа выражений не возвращает целевого результата.

УСЛОВИЯ И ЦИКЛЫ

```
if( логическое_выражение )
{ группа_выражений_1 если логическое_выражение равно TRUE }
else { группа_выражений_2 в противном случае }

compare <- function(x, y){ n1 <- length(x) ; n2 <- length(y)
    if(n1!= n2){
        if(n1 > n2){ z=(n1 - n2)
            cat("Первый вектор имеет на ",z," элементов больше \n") }
        else{ z=(n2 - n1)
            cat("Второй вектор имеет на ",z," элементов больше \n")}}
    else{cat("Количество элементов одинаково ",n1,"\n") }
}

x <- c(1:4)
y <- c(1:9)
compare(x, y)
Первый вектор имеет на 5 элементов больше
```

```
ifelse(логическое_выражение, группа_выражений_1, группа_выражений_2)
```

Повторение в цикле одних и тех же вычислительных операций осуществляется с использованием конструкций for(), while() или repeat()

```
for (index in for_object) { группа_выражений }

while(логическое_выражение) { группа_выражений }

repeat { группа_выражений ; break }
```

2.4. Базовые графические возможности

2.4.1. Функция plot() и ее параметры

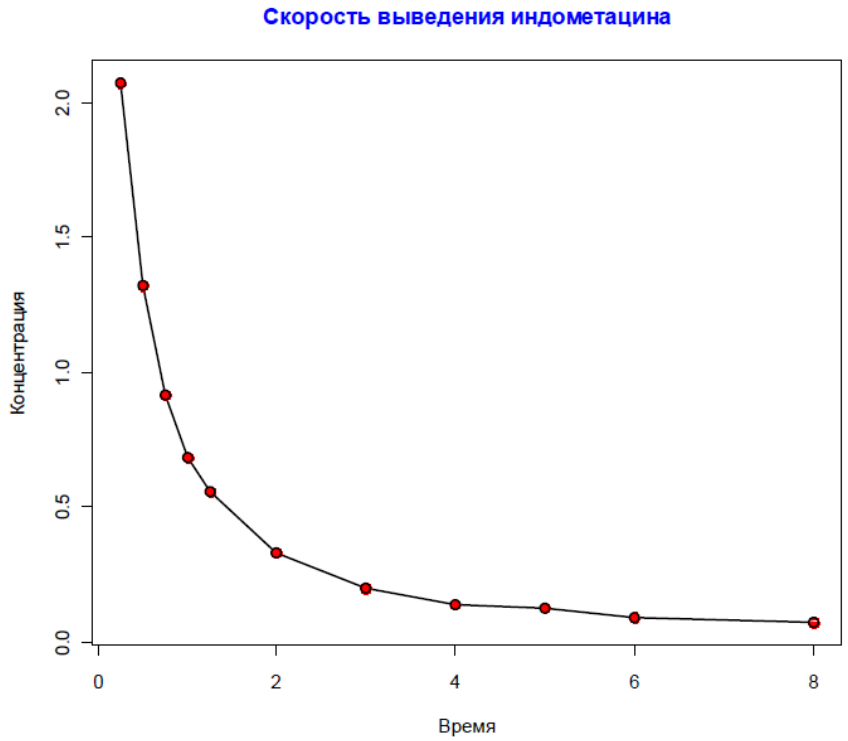
Задача: исследовать скорость выведения из организма человека индометацина – одного из наиболее активных противовоспалительных препаратов. В эксперименте приняли участие шесть испытуемых.

```
names(Indometh)
[1] "Subject" "time" "conc"
```

```
plot(time, conc)
```

Аргументы

1. Параметры `xlab` и `ylab`
2. Параметр `type`
3. Параметры `xlim` и `ylim`
4. Параметры `axes` и `ann`
5. Параметр `log`
6. Параметр `main`



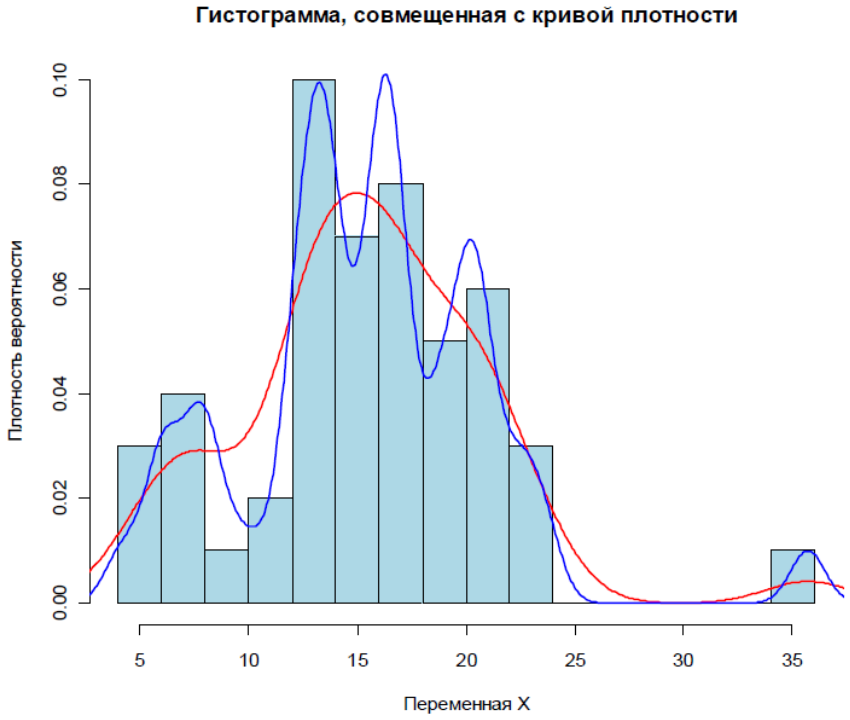
2.4. Базовые графические возможности

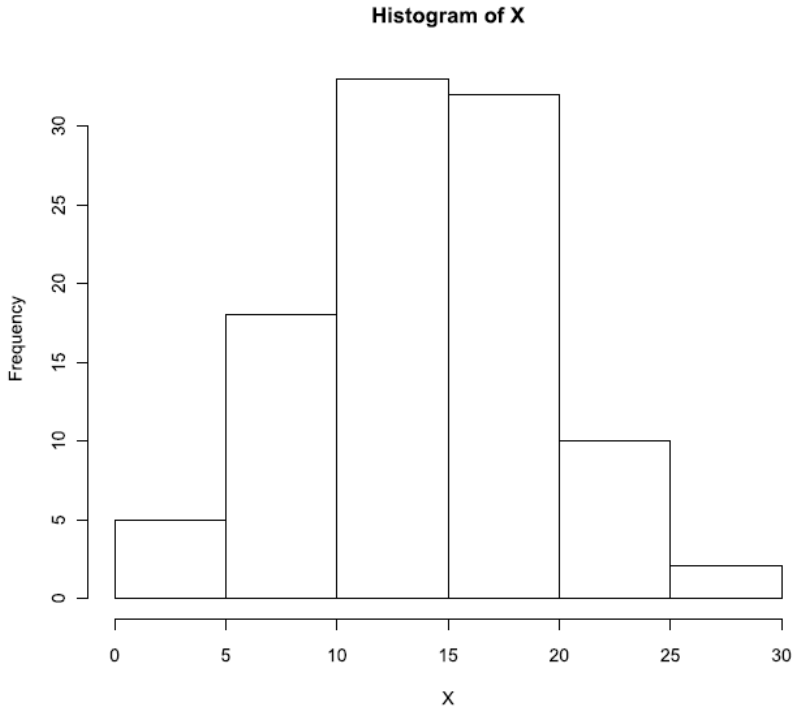
2.4.2. Гистограммы

В системе R для построения гистограмм служит функция `hist()`.

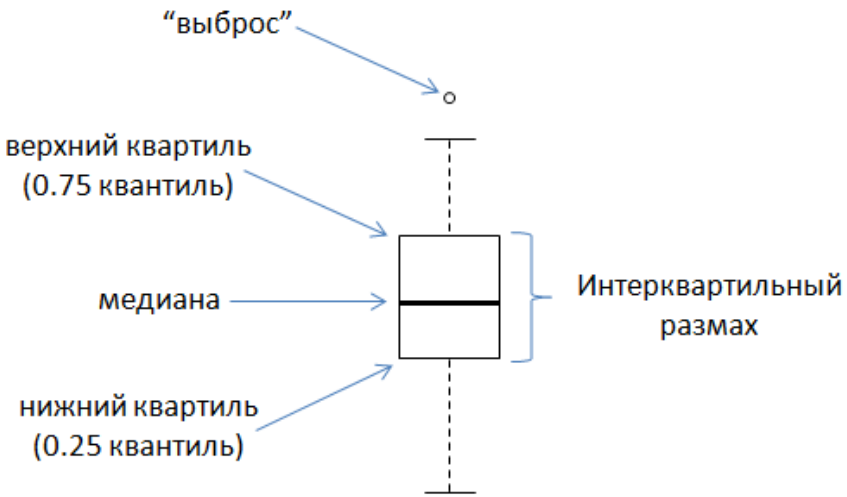
```
X <- rnorm(n = 100, mean = 15, sd = 5)
hist(X)
```

Оценка плотности вероятности выполняется при помощи функции `density()`

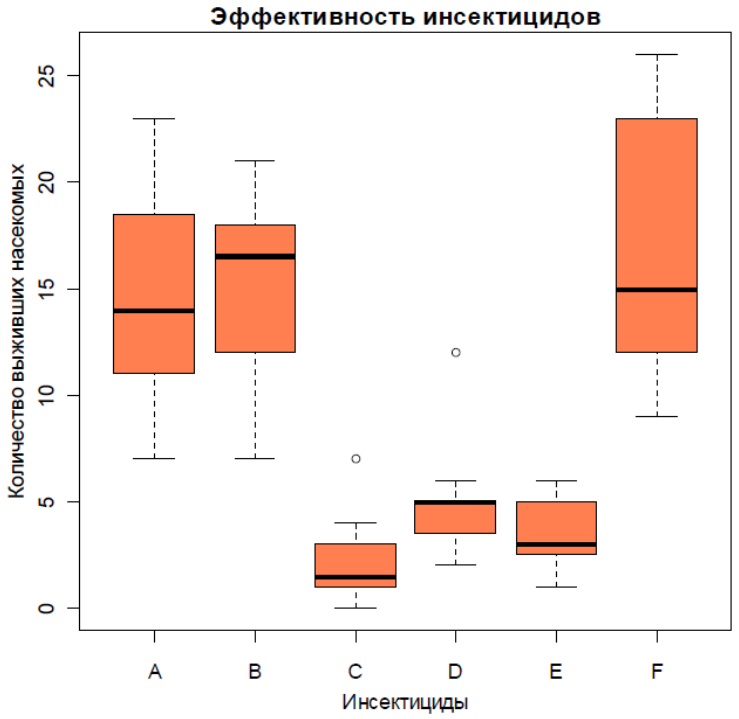




Диаграммы размахов, или "ящики с усами" (англ. box-whisker plots)




```
boxplot(count ~ spray,
     xlab = "Инсектициды",
     ylab = "Количество выживших насекомых",
     main = "Эффективность инсектицидов",
     col = "coral", data = InsectSprays)
```



2.4. Базовые графические возможности

2.4.4. Круговые и столбиковые диаграммы

Функция `pie()`:

- `x` – вектор из положительных чисел, на основе которых строится диаграмма;
- `labels` – текстовый вектор, содержащий подписи секторов диаграммы; если значения x уже имеют атрибут `names` (имена), то аргумент `labels` указывать не обязательно;
- `radius` – изменяет размер квадрата, внутри которого строится диаграмма
- `init.angle` – угол поворота диаграммы;
- `col` – вектор (числовой или текстовый), содержащий коды цветов для заливки секторов диаграммы;
- `main` – текстовый вектор, содержащий заголовок диаграммы;
- ... – другие графические параметры (например, параметры, определяющие размер подписей секторов диаграммы, цвет линий, и т.п.).

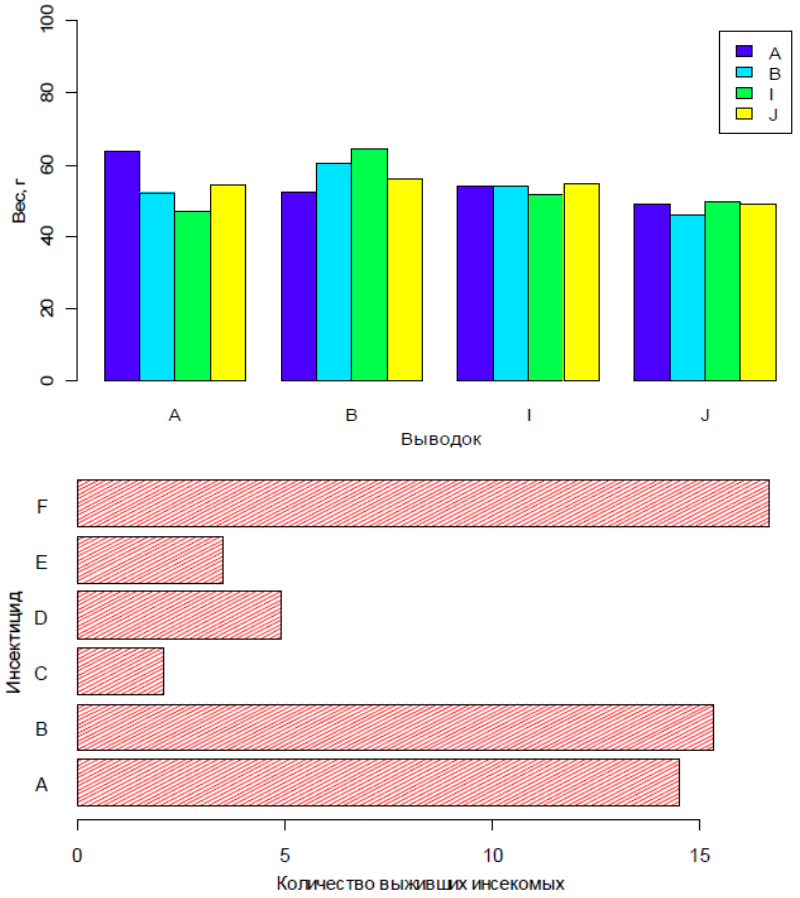
```
pie(percent.voted, radius = 0.9, cex = 0.6,
 + main = "Явка", col = c("black", "gray80"))
pie(votes, cex = 0.6, radius = 0.9,
 + init.angle = -10, main = "Распределение
 + голосов", col = c(2:8))
```

2.4. Базовые графические возможности

2.4.4. Круговые и столбиковые диаграммы

Функция `barplot()`:

- `hight` ("высота")
  `width` ("ширина")
- `space` ("пространство") – величина зазора между столбцами
- `names.arg` – текстовый вектор, содержащий подписи (вдоль оси X) для каждого столбца или группы столбцов.
- `legend.text` – вектор, содержащий текстовые элементы легенды графика.
- `horiz` – принимает логическое значение: TRUE для горизонтального расположения столбцов и FALSE – для вертикального.
- `density` – числовой вектор, задающий плотность заштриховки столбцов.
- `angle` – угол наклона штрихов (в градусах).
- `col` – вектор цветовых кодов для столбцов или их элементов.
- `border` – код цвета для обводки столбцов.
- ... – другие графические параметры



2.4. Базовые графические возможности

2.4.5. Категоризированные графики

Функция **coplot()**:

`formula` – формула, описывающая взаимодействие между анализируемыми переменными;

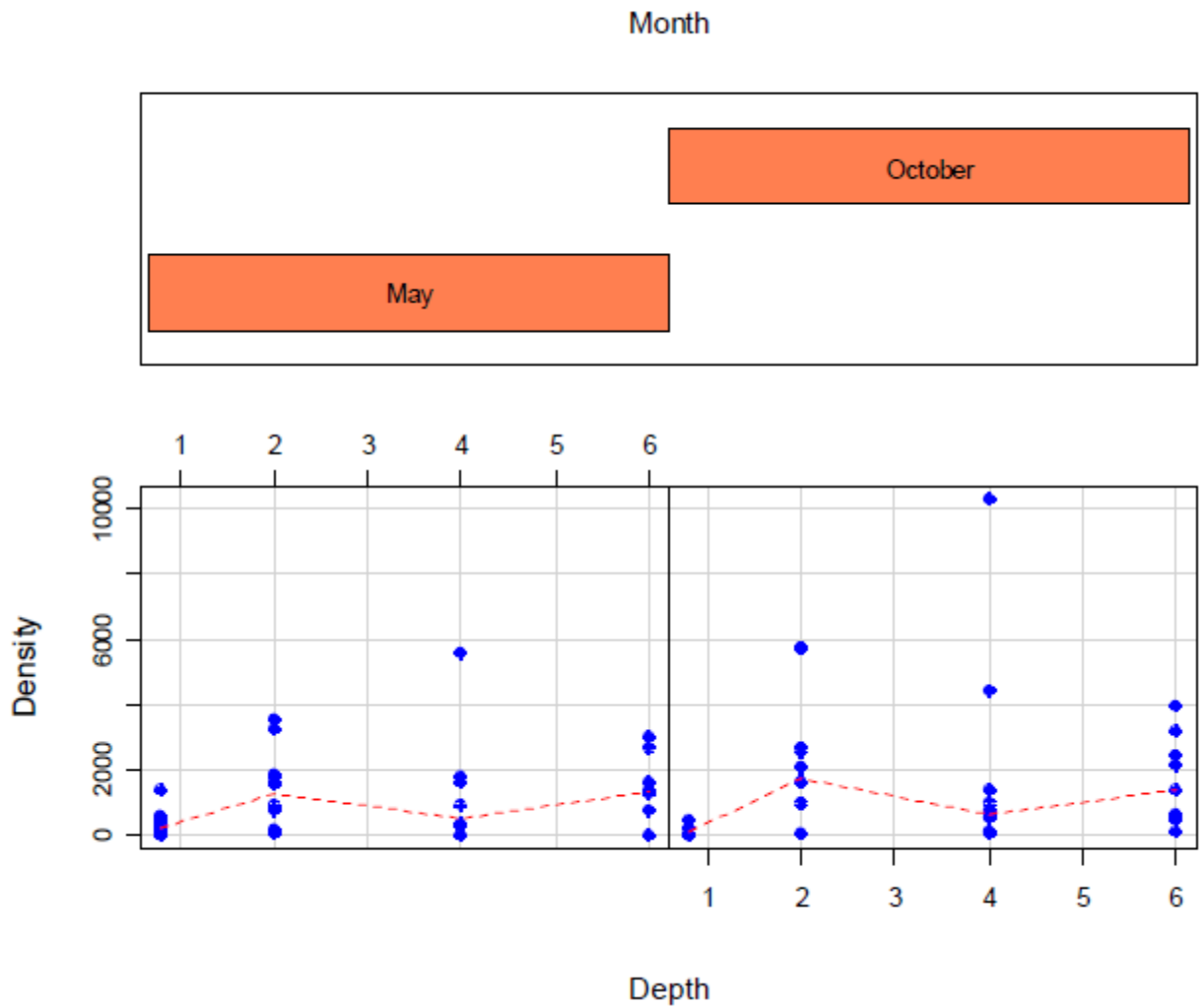
- `data` – таблица данных, содержащая значения переменных, указанных в `formula`;
- `panel` – функция, позволяющая задать тип и настроить внешний вид отдельных панелей категоризованного графика;
- `rows` и `columns`
- `show.given`
- `number` – количество интервалов, на которые разбиваются переменные a и b в случаях, если эти переменные не являются факторами.

```
density <- read.delim(file =
"http://dl.dropbox.com/u/7521662/Dreissena_in_Naroch_Lake.txt",
  header = TRUE)
str(density)
'data.frame': 79 obs. of  4 variables:
 $ Transect: Factor w/ 3 levels "A","B","C": 1 1 1 1 1 1 1 1 1 1 ...
 $ Month : Factor w/ 2 levels "May","October": 1 1 1 1 1 1 1 1 1 1 ...
 $ Depth   : num  0.8 0.8 0.8 0.8 0.8 0.8 0.8 0.8 2 2 ...
 $ Density : int  36 340 40 40 24 56 28 0 1560 40 ...
```

```
coplot(Density ~ Depth | Month, data = density, panel = function(x, y, ...) {
panel.smooth(x, y, lty = 2, pch = 19, col = "blue", span = 0.6)}, bar.bg = c(fac =
"coral"), pch = 19, col = "blue", xlab = c("Depth", "Month"), ylab = "Density")
```

2.4. Базовые графические возможности

2.4.5. Категоризированные графики



Откуда скачать R?

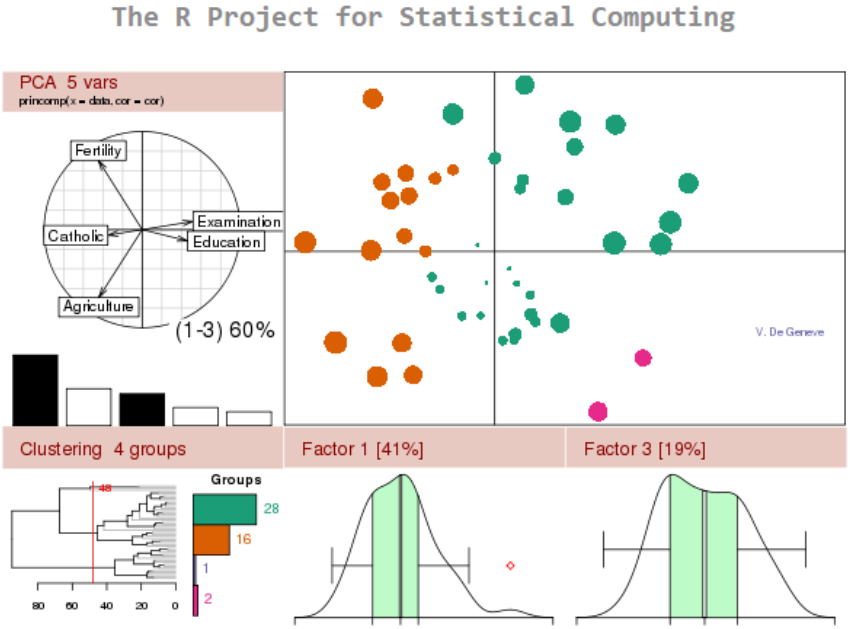
http://www.r-project.org/

About R
What is R?
Contributors
Screenshots
What's new?

Download, Packages
CRAN

R Project
Foundation
Members & Donors
Mailing Lists
Bug Tracking
Developer Page
Conferences
Search

Documentation
Manuals
FAQs
The R Journal
Wiki
Books
Certification
Other



Getting Started:

- R is a free software environment for statistical computing and graphics. It compiles and runs on a wide variety of UNIX platforms, Windows and MacOS. To **download R**, please choose your preferred CRAN mirror.
- If you have questions about R like how to download and install the software, or what the license terms are, please read our answers to frequently asked questions before you send an email.

Литература по R

Мастицкий С. Э., Шитиков В. К. (2014) Статистический анализ и визуализация данных с помощью R. - Электронная книга, 400 с

Буховец А. Г., Москалев П. В., Богатова В. П., Бирючинская Т. Я. (2010) Статистический анализ данных в системе R. Учебное пособие. Воронеж: ВГАУ, 124 с.

Зарядов И. С. (2010) Введение в статистический пакет R: типы переменных, структуры данных, чтение и запись информации, графика. М.: Издательство Российского университета дружбы народов, 207 с.

Зарядов И. С. (2010) Статистический пакет R: теория вероятностей и математическая статистика. М.: Издательство Российского университета дружбы народов, 141 с.

Шитиков В. К., Розенберг Г. С. (2012) Рандомизация, бутстреп и методы Монте-Карло. Примеры статистического анализа данных по биологии и экологии. Тольятти: Ин-т экологии Волжского бассейна.

Шипунов А. Б., Балдин Е. М., Волкова П. А., Коробейников А. И., Назарова С. А., Петров С. В., Суфиянов В. Г. (2012) Наглядная статистика. Используем R! - М.: ДМК Пресс, 298 с.

Кабаков Р. К. (2014) R в действии. Анализ и визуализация данных на языке R Издательство: ДМК Пресс, 580 с.

Joseph Adler (2009) R in a Nutshell