

Администрация городского округа Самара
Муниципальное бюджетное образовательное учреждение
высшего образования
«Самарская академия государственного и муниципального управления»
Кафедра «Общественные дисциплины и право»

В. Н. Кожухова, А. А. Коробецкая, В. К. Семенычев

Свободная программная среда R

Практикум
для студентов, обучающихся по направлению
38.04.01 «Экономика» (уровень подготовки магистратуры)

Самара
Издательство «САГМУ»
2016

УДК 519.682.4+330.43

ББК 65в631

К 58

*Печатается по решению Учебно-методического совета
Муниципального бюджетного образовательного учреждения
высшего образования
«Самарская академия государственного и муниципального управления»*

*Рецензент: Митрофанов А. Н. — д-р техн. наук, профессор, декан факультета
«Экономика и управление» Самарской академии государственного
и муниципального управления*

Кожухова В. Н., Коробецкая А. А., Семенычев В. К.

К 58 Свободная программная среда R: практикум для студентов, обучающихся по направлению 38.04.01 «Экономика» (уровень подготовки магистратуры). — Самара: САГМУ, 2016. — 48 с.

ISBN 978-5-94189-154-2

Данный практикум предназначен студентам, обучающимся по направлению подготовки магистратуры 38.04.01 «Экономика» для дисциплины «Эконометрика (продвинутый курс)».

Практикум формирует следующие профессиональные компетенции: ПК-8 — способность готовить аналитические материалы для оценки мероприятий в области экономической политики и принятия стратегических решений на микро- и макроуровне; ПК-10 — способность составлять прогноз основных социально-экономических показателей деятельности предприятия, отрасли, региона и экономики в целом.

В практикуме предлагается цикл лабораторных работ, направленных на изучение приемов работы в свободной среде R. Особенностью данного практикума является изучение современных методов решения нелинейного МНК, таких как генетические алгоритмы.

Практикум может быть полезен и для тех, кто изучает экономику и математическое моделирование по другим направлениям подготовки, а также аспирантам, исследователям и преподавателям.

УДК 519.682.4+330.43

ББК 65в631

ISBN 978-5-94189-154-2

© Кожухова В. Н., Коробецкая А. А.,
Семенычев В. К., 2016

© Муниципальное бюджетное образовательное
учреждение высшего образования «Самарская
академия государственного и муниципального
управления», 2016

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ	4
ЛАБОРАТОРНАЯ РАБОТА 1. ОСНОВЫ РАБОТЫ С R.	
ОБРАБОТКА СТАТИСТИЧЕСКИХ ДАННЫХ	8
Цель работы	8
Задание	8
Указания к выполнению работы.....	8
Консольный режим	8
Написание скриптов	11
Установка пакетов	14
Контрольные вопросы	19
ЛАБОРАТОРНАЯ РАБОТА 2. ЛИНЕЙНАЯ РЕГРЕССИЯ.....	19
Цель работы	19
Задание	19
Указания к выполнению работы.....	19
Парная линейная регрессия	19
Множественная линейная регрессия.....	23
Контрольные вопросы	27
ЛАБОРАТОРНАЯ РАБОТА 3. ВРЕМЕННЫЕ РЯДЫ	27
Цель работы	27
Задание	27
Указания к выполнению работы.....	28
Анализ временных рядов	28
Загрузка данных из различных источников в R.....	33
Контрольные вопросы	35
ЛАБОРАТОРНАЯ РАБОТА 4. ГЕНЕТИЧЕСКИЕ АЛГОРИТМЫ	36
Цель работы	36
Задание	36
Указания к выполнению работы.....	36
Генетические алгоритмы	36
Генерация данных	42
Контрольные вопросы	43
ЛИТЕРАТУРА	44
ВАРИАНТЫ ЗАДАНИЙ	45

ВВЕДЕНИЕ

Данный практикум призван познакомить читателя с некоторыми особенностями и возможностями популярного и активно развивающегося программного средства — языка R.

Классическая статистика в основном основывается на проверке статистических гипотез — априорных предположений о свойствах исследуемых данных.

Появление компьютеров позволило существенно повысить эффективность вычислений. Но в первое время, пока компьютеры были еще очень слабы в области отображения информации, особенно графической, практика статистики, как и подготовка статистиков, ориентировалась на построение моделей и на проверку гипотез.

Подобный подход реализован в программных средствах, подобных SPSS, которые базируются на электронных таблицах и управляются с помощью меню. Фактически первые версии программных продуктов SPSS и SAS Analytics состояли из подпрограмм, которые можно было вызвать из основной программы (на Fortran или на другом языке) с целью подгонки и проверки модели из имеющегося набора моделей.

Альтернативой этому подходу стала концепция разведочного анализа данных (Exploratory Data Analysis, EDA), предложенная Джоном Тьюки (John Tukey). Разведочный анализ данных применяется для нахождения связей между переменными в ситуациях, когда отсутствуют (или недостаточны) априорные представления о природе этих связей. Как правило, при разведочном анализе учитывается и сравнивается большое число переменных, а для поиска закономерностей используются самые разные методы.

R соединяет в себе эти концепции, позволяя использовать как основные статистические методы, так и более сложные, специально разработанные методы многомерного анализа, предназначенные для отыскания закономерностей в многомерных данных.

R, в отличие от SAS и SPSS, — **это универсальный язык программирования**, разработанный для применения в таких областях, как разведочный анализ данных, классические статистические тесты и высокоуровневая графика.

R обладает рядом существенных достоинств.

R — свободная программная среда вычислений **с открытым исходным кодом**. Это реализация языка S с дополнительными модулями, разработанными в языке S-Plus. В некоторых случаях мо-

делями в обоих языках занимались одни и те же люди. R доступен в соответствии с лицензией GNU.

На этом фундаменте R продолжает развиваться, в значительной степени посредством добавления пакетов. R-пакет представляет собой коллекцию наборов данных, функций языка R, документации и динамически загружаемых элементов на языке C или Fortran. Посредством этих пакетов исследователи могут с легкостью обмениваться вычислительными методами со своими коллегами.

R уже **имеет собственную обширную и непрерывно расширяющуюся библиотеку пакетов**, содержащую большое количество готовых решений различных задач.

Некоторые пакеты имеют ограниченную область применения, другие представляют целые области статистики, а некоторые отражают новейшие разработки. Многие новые разработки в области статистики, эконометрики, биологии, химии, медицины, географии и других прикладных областей сначала появляются как R-пакеты, и только потом реализуются в коммерческих программных продуктах.

R — это мощный скриптовый язык. Скриптом называется программный сценарий, любая исполняемая процедура, которая запускается автоматически или же с помощью команды пользователя. Скрипты используют не только в программировании, область их применения шире. Скрипт может быть использован для повторяемых и сложных для запоминания пользователю операций.

Для обработки неупорядоченных данных требуются возможности языка программирования: продукты SAS и SPSS имеют скриптовые языки для решения отдельных задач, однако R был создан именно как язык программирования и поэтому является более подходящим средством для этой цели.

Таким образом, R является полезным инструментом в области анализа больших массивов данных. Он уже **интегрирован в ряд коммерческих пакетов**, таких как IBM SPSS и InfoSphere, а также Mathematica.

R — язык, ориентированный на статистику, который можно рассматривать в качестве конкурента для таких аналитических систем, как SAS Analytics, не говоря уже о таких более простых пакетах, как StatSoft STATISTICA или Minitab.

R **органично интегрируется с системами публикации** документов, что позволяет встраивать статистические результаты и графику из среды R в документы публикационного качества.

Эта возможность нужна не всем, однако если вы хотите написать книгу о своем анализе данных или просто не любите копиро-

вать свои результаты в документы текстового процессора, то самый короткий и самый элегантный маршрут состоит в использовании R.

В качестве языка программирования R подобен многим другим языкам. Любой человек, который когда-либо писал программный код, найдет в R множество знакомых моментов. Отличительные особенности R лежат в статистической философии, которую он исповедует.

Язык R имеет легкий синтаксис — это универсальный инструмент, разработанный специально для работы с данными. R также включает в себя чрезвычайно мощные графические возможности.

В данном практикуме рассматривается работа с R из-под операционной системы Windows.

Для выполнения практикума читателю необходимо установить:

1. Язык R:
 - классический <https://cran.rstudio.com/bin/windows/base/>;
 - или PRO <https://mran.revolutionanalytics.com/download/>.
2. Графический интерфейс пользователя RStudio: <https://www.rstudio.com/products/rstudio/download/> — набор интегрированных инструментов, разработанных, чтобы наиболее продуктивно использовать R.

RStudio включает в себя консоль, редактор с подсветкой синтаксиса, поддерживающий как прямое выполнение кода, так и инструменты для построения графиков, сохранение истории команд, отладку и управление рабочим пространством.

При установке R будет лучше, если имя пользователя Windows будет написано латиницей. Если имя пользователя набрано кириллицей (например, "Иван"), то необходимо создать нового пользователя с именем без кириллицы (например, "Ivan"), а установку программ и дальнейшую работу рекомендуется проводить, войдя в систему под «английским» пользователем.

На время установки R и RStudio также рекомендуется отключить антивирусную программу. Другим вариантом избегания ошибок, связанных с кириллицей, является установка R и дальнейшая работа от имени администратора.

При сохранении *.R файлов не следует использовать названия файлов или папок, содержащих русские буквы или пробелы, во избежание ошибок.

Для работы в R потребуется стабильное Интернет-соединение, т. к. может потребоваться онлайн-загрузка дополнительных пакетов для обработки данных.

Англоязычных источников по R чрезвычайно много. При возникновении вопросов, ответов на которые нет в справке R, можно использовать следующие ресурсы:

- <http://rseek.org/> — Google, модифицированный под поиск по источникам, связанным с R;
- <http://stats.stackexchange.com/> — обращаться с вопросами по поводу статистических методов и их реализации в R;
- <http://stackoverflow.com/> — обращаться с вопросами по поводу программирования в R;
- <http://vk.com/rstatistics> — группа в социальной сети «ВКонтакте», участники которой могут ответить на возникшие вопросы;
- <http://r-analytics.blogspot.ru/> — один из самых масштабных русскоязычных проектов по R.

Данный курс предполагает знание читателями базового курса эконометрики и основ программирования на любом языке высокого уровня.

ЛАБОРАТОРНАЯ РАБОТА 1. ОСНОВЫ РАБОТЫ С R. ОБРАБОТКА СТАТИСТИЧЕСКИХ ДАННЫХ

Цель работы

Ознакомиться с интерфейсом RStudio, научиться работать в режиме консоли и путем написания скриптов, а также подключать внешние пакеты, изучить основные методы обработки статистических данных.

Задание

1. Загрузить данные для своего варианта в переменную-вектор.
2. Получить справочную информацию по своим данным, просмотреть их содержимое.
3. Проверить, есть ли среди данных пропуски.
4. Создать новую переменную-вектор, в которой будут 1, если значение в исходном векторе больше среднего, и -1, если значение переменной меньше среднего, и 0, если значение равно среднему.
5. Вывести описательную статистику.
6. Построить графики абсолютных частот и плотности распределения.

Указания к выполнению работы

Консольный режим

После запуска RStudio пользователь попадает в консольный режим работы (рис. 1). Любая команда, написанная пользователем, будет сразу выполнена R по нажатию Enter.

Рассмотрим основные выражения в R: числа, строки и логические переменные.

Можно использовать R как калькулятор, например:

```
> 1 + 1
[1] 2
> 6 * 7
[1] 42
> sqrt(16)
[1] 4
```

Результат сразу появится в консоли.

Строки печатаются в кавычках: двойных или одинарных:

```
> "Hello world!"
[1] "Hello world!"
```



```
> 'Hello world!'
[1] 'Hello world!'
```

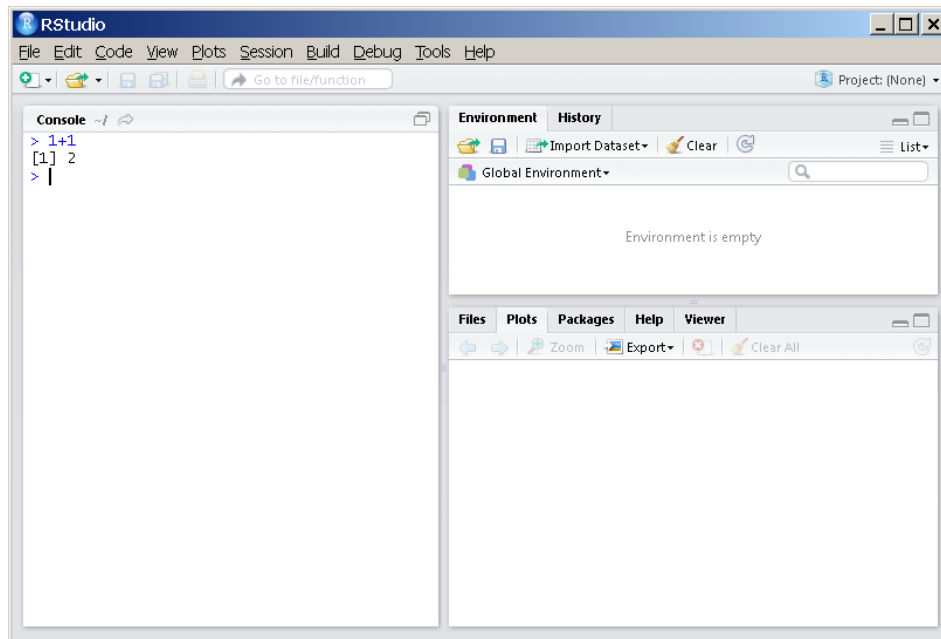


Рис. 1. Консольный режим в RStudio

Логические выражения возвращают TRUE или FALSE:

```
> 3 < 4
[1] TRUE
```

Чтобы сравнить два выражения, используется двойной знак равенства:

```
> 2 + 2 == 5
[1] FALSE
```

Как и в других языках программирования, можно сохранять значения в переменную. Сохраним 42 в переменную `x`:

```
> x <- 42
```

И в обратную сторону:

```
> 5 -> x
```

Можно распечатать значение переменной в любое время, просто набрав ее имя в консоли. Попробуем напечатать текущее значение `x`:

```
> x
[1] 42
```

Можно так же повторно назначить любое значение переменной в любое время.

R чувствителен к регистру: переменные `x` и `X` — это разные переменные:

```
> X
[1] 5
```

Чтобы вызвать функцию, нужно обратиться к ней по имени, указав в скобках нужные аргументы. Например, функция суммы:

```
> sum(1, 3, 5)
[1] 9
```

Получить помощь по функции можно командой `help(functionname)` или `?functionname`.

В правом нижнем углу на вкладке Help появится справка (рис. 2):

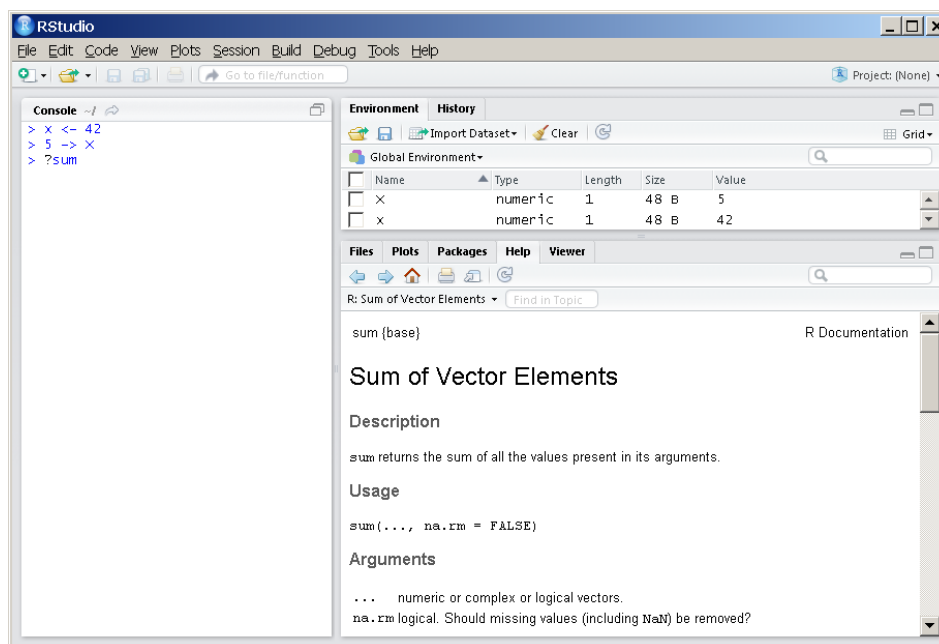


Рис. 2. Справка по функции `sum`

Зададим вектор с помощью функции `c` (сокр. от англ. **C**ombine):

```
> y <- c(-3, 2, NA, 5)
> y
[1] -3 2 NA 5
```

`NA` — это пропущенное наблюдение (от англ. **N**ot **A**vailable). Его не следует путать с `NaN` (**N**ot a **N**umber — «не число», неопределенность):

```
> 0/0
[1] NaN
```

Попробуем просуммировать элементы вектора `y`:

```
> sum(y)
[1] NA
```

Необязательным аргументом функции `sum` является `na.rm` (сокр. от англ. **R**emove **N**A), по умолчанию равный `FALSE`.

Если указать для него значение «истина», то функция суммы будет складывать все элементы вектора, исключая пропущенные:

```
> sum(y, na.rm = TRUE)
[1] 4
```

Последовательность чисел можно задать двумя способами: `start:end` либо функцией `seq()`:

```
> 5:9
[1] 5 6 7 8 9
> seq(5,9)
[1] 5 6 7 8 9
> seq(10,50, by = 10)
[1] 10 20 30 40 50
```

Обращаться к элементам вектора можно, используя квадратные скобки:

```
> sentence <- c('mack', 'the', 'knife')
> sentence[3]
[1] "knife"
> sentence[c(1,3)]
[1] "mack" "knife"
```

Либо можно задать элементам вектора имена:

```
> ranks <- 1:3
> names(ranks) <- c("first", "second", "third")
> ranks
  first second  third
    1      2      3
> ranks["first"]
first
    1
```

Написание скриптов

В R удобнее писать не по одной команде, а сразу целый набор команд и потом запускать их все на выполнение. Для этого нужны скрипты.

Чтобы создать скрипт, следует выбрать `File -> New File -> R Script`. Откроется новая область, в которой можно писать команды. Комментарии, которые не будет выполнять R, пишутся со знака `#` (рис. 3).

По нажатию `Enter` команды в скрипте выполняться не будут, а будет лишь осуществлен переход на новую строку.

Чтобы выполнить команду в режиме скрипта, следует поставить курсор на нужную строку и нажать `Ctrl+Enter`. Если команда занимает более одной строки, то необходимо ставить знак `+` в конце каждой строки. Команда выполняется построчно, в каждой строке требуется нажимать `Ctrl+Enter`, либо выделить всю команду целиком и нажать `Ctrl+Enter` один раз.

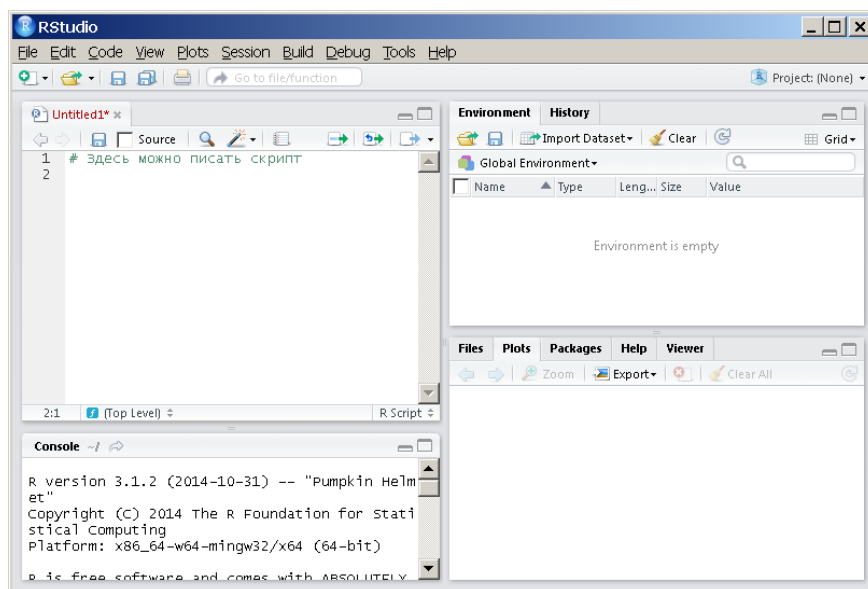


Рис. 3. Создание нового скрипта

R может подсказывать, какие команды доступны, если начать вводить первые символы и нажать либо Tab, либо Ctrl+Enter.

В основном в R работают с наборами данных. Такая структура носит в R название `data.frame` и представляет собой таблицу, в которой каждый столбец — это некоторая переменная, а каждая строка — это одно наблюдение.

Создадим в режиме скрипта `data.frame`. Пусть имеются наблюдения за ростом и весом некоторых людей. Зададим два вектора:

```
rost <- c(160, 175, 155, 190, NA)
ves <- c(NA, 70, 48, 85, 60)
```

И объединим их в набор данных, который поместим в переменную `df`, а затем выведем на экран:

```
df <- data.frame(rost, ves)
df
```

В консоли получим следующую таблицу:

```
  rost ves
1  160  NA
2  175  70
3  155  48
4  190  85
5   NA  60
```

Обращаться к конкретным наблюдениям `df` можно, используя квадратные скобки:

```
> df[3,1]
[1] 155
```

Обращаться к переменным можно, используя знак `$` или указывая столбец с пропуском номера строки:

```
> df$rost
[1] 160 175 155 190  NA
```

или

```
> df[,1]
[1] 160 175 155 190 NA
```

Обращаться к наблюдениям можно, указывая конкретную строку и пропуская номер столбца:

```
> df[4,]
  rost ves
4  190  85
```

Основные описательные статистики (среднее, стандартное отклонение и медиану) можно получить с помощью функций `mean`, `sd` и `median`:

```
mean(df$rost, na.rm = T)
[1] 170
sd(df$rost, na.rm = T)
[1] 15.81139
median(df$rost, na.rm = T)
[1] 167.5
```

Можно сохранить скрипт, нажав `File -> Save`. При первом сохранении R предложит выбрать кодировку. Рекомендуется указать UTF-8 (рис. 4), чтобы русские буквы (например, в комментариях) отображались корректно. Затем необходимо выбрать директорию и задать имя файла, который будет сохранен с расширением `*.R`.

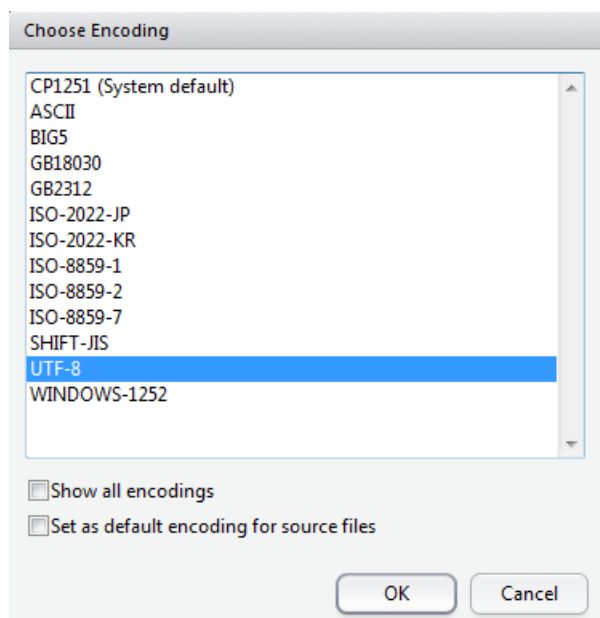


Рис. 4. Выбор кодировки при сохранении

Установка пакетов

В R существует базовый набор пакетов (библиотек), содержащих самые необходимые функции [2]. Для реализации задач эконометрики требуются дополнительные пакеты.

Для работы с внешними пакетами необходимо выполнить два действия — установку и подключение нужного пакета. Установку необходимо выполнить один раз, а подключать — в каждой рабочей сессии.

Для установки пакетов существует функция `install.packages`. Также автоматически будут доустановлены связанные пакеты.

Для подключения установленного пакета следует воспользоваться функцией `library` [1].

Например, установим следующие пакеты:

`psych` — содержит функции для расчета описательных статистик;

`dplyr` — содержит функции для работы с `data.frame`;

`ggplot2` — самый мощный пакет для построения красивых графиков, диаграмм, карт и т. д.

```
install.packages(c("psych", "dplyr", "ggplot2"))
```

Прямым сообщением об ошибке установки является только слово `Error`, появляющееся в консоли. Все остальные сообщения `Warning` являются просто предупреждениями о чем-либо.

Для того чтобы определить, какие пакеты нужны для работы, можно воспользоваться поиском в сети Интернет, задав вопрос на английском языке. Например, чтобы найти, в каком пакете находится алгоритм Левенберга–Марквардта для расчета нелинейного МНК, можно набрать в поиске «levenberg-marquardt algorithm in r» и первой же ссылкой будет пакет `minpack.lm` в R.

Для выполнения данной работы понадобится подключить следующие пакеты:

```
library("psych") # описательные статистики
library("lmtest") # тестирование гипотез в линейных моделях
library("ggplot2") # графики
library("dplyr") # манипуляции с данными
library("MASS") # подгонка распределений
```

Получим, например, описание набора данных по автомобилям `cars` командой:

```
help(cars)
```

Результат выполнения команды (в правом нижнем углу на вкладке Help) показан на рисунке 5.

В этом наборе данных 50 наблюдений и две переменных (скорость, миль/час и длина тормозного пути в футах).

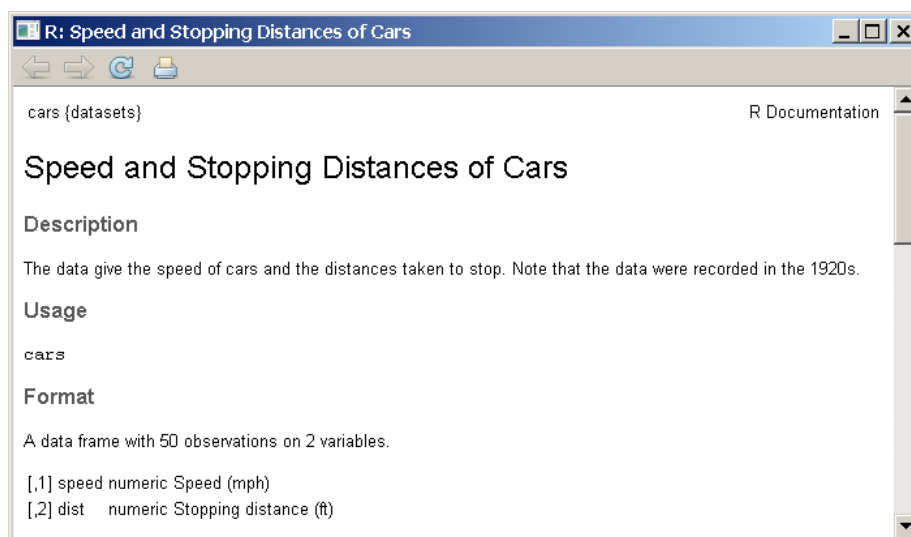



Рис. 5. Справка по набору данных cars

Поместим в переменную `d` встроенный в R набор данных по автомобилям¹:

```
d <- cars # этот набор данных находится в базовом пакете datasets
```

Теперь `d` имеет тип данных `data.frame` (набор данных), в чем можно удостовериться, посмотрев в правом верхнем углу окна таблицу среды Environment (рис. 6).

Для этого должен быть выбран режим Grid. С помощью кнопки  можно просмотреть содержимое набора данных в виде таблицы.

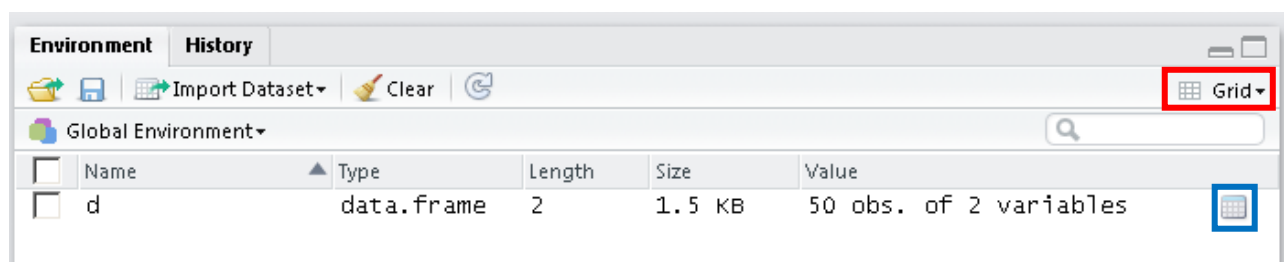


Рис. 6. Описание набора данных `d` в таблице среды Environment

Следующей командой можно посмотреть на этот набор данных, в результате чего будут перечислены все переменные и типы данных:

```
glimpse(d) # функция из пакета dplyr
```

¹ В дальнейшем можно работать как с переменной `d`, так и непосредственно с `cars`, но в последнем случае есть риск испортить исходные данные.

Результат выполнения команды появится в консоли:

```
> d <- cars
> glimpse(d)
Observations: 50
Variables: 2
$ speed (dbl) 4, 4, 7, 7, 8, 9, 10, 10, 10, 11, 11, 12, 12, 12, 12, 13, ...
$ dist (dbl) 2, 10, 4, 22, 16, 10, 18, 26, 34, 17, 28, 14, 20, 24, 28, ...
```

Переменные `speed` и `dist` имеют тип данных `dbl` (`double`) и содержат по 50 наблюдений. Для других типов данных используются следующие сокращения: `chr` (`character/string`), `int` (`integer`), `fctr` (`factor`), `tims` (`time`), `lgl` (`logical`).

Посмотрим на первые шесть наблюдений набора данных `d`:

```
> head(d) # функция из базового пакета utils
  speed dist
1     4    2
2     4   10
3     7    4
4     7   22
5     8   16
6     9   10
```

и последние шесть наблюдений:

```
> tail(d) # функция из базового пакета utils
  speed dist
45    23   54
46    24   70
47    24   92
48    24   93
49    24  120
50    25   85
```

Получим таблицу с описательными статистиками: среднее, мода, медиана, стандартное отклонение, минимум/максимум, асимметрия, эксцесс и т. д.:

```
> describe(d) # функция из пакета psych
      vars  n mean   sd median trimmed  mad min max range  skew kurtosis
speed    1 50 15.40  5.29     15   15.47  5.93   4  25   21 -0.11   -0.67
dist     2 50 42.98 25.77     36   40.88 23.72   2 120  118  0.76    0.12
      se
speed 0.75
dist  3.64
```

Построим гистограмму абсолютных частот для переменной `dist` (длины тормозного пути).

Воспользуемся функцией `qplot`, задав источник данных `d` (аргумент `data`), переменную для построения графика (`dist`), подпи-

шем оси (параметры функции `xlab` и `ylab`) и название графика (параметр `main`):

```
# функция из пакета ggplot2
qplot(data=d, dist, xlab="Длина тормозного пути (футы)", ylab="Число автомобилей", main="Данные по автомобилям 1920х")
```

Результат выполнения данной функции показан на рисунке 7.

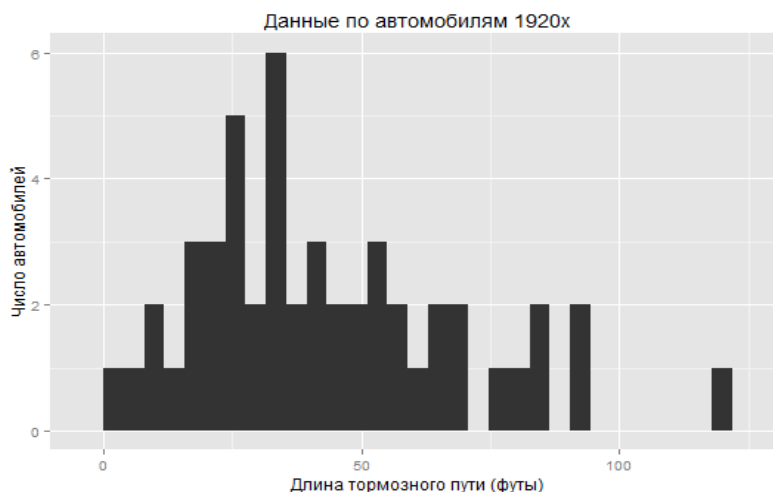


Рис. 7. Гистограмма абсолютных частот для переменной `dist`

Можно построить также гистограмму плотности распределения (рис. 8):

```
# функция из базового пакета graphics
hist(d$dist, probability = TRUE, col="grey")
```

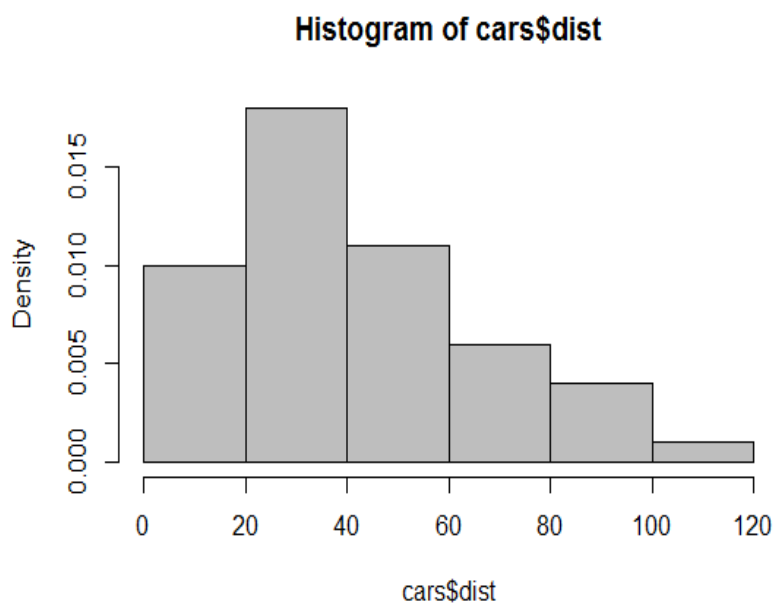
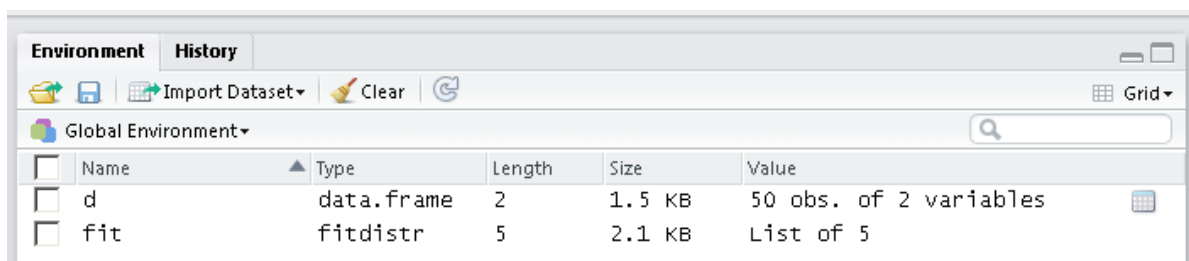


Рис. 8. Гистограмма плотности распределения для переменной `dist`

Подгоним плотность распределения Вейбулла, поместив результат (оценки параметров распределения) в переменную `fit`:

```
fit <- fitdistr(d$dist, "weibull") # функция из пакета MASS
```

Переменная `fit` теперь представляет собой список (List) из 5 элементов, что можно увидеть в Environment (рис. 9).



Name	Type	Length	Size	Value
d	data.frame	2	1.5 KB	50 obs. of 2 variables
fit	fitdistr	5	2.1 KB	List of 5

Рис. 9. Описание переменной `fit` в таблице среды Environment

Доступ к элементам списка можно получить через значок доллара `$`.

Оценки двух параметров распределения Вейбулла были рассчитаны методом максимального правдоподобия.

Посмотрим их, обратившись к элементу списка `fit`:

```
> fit$estimate
  shape    scale 
1.72371 48.15234
```

Покажем на том же графике теоретическую плотность распределения Вейбулла (рис. 10):

```
xvals <- seq(0, 120, .20) # значения по оси абсцисс от 0 до 120 с шагом 0,2
lines(xvals, dweibull(xvals, shape=fit$estimate[1],
scale=fit$estimate[2]))
```

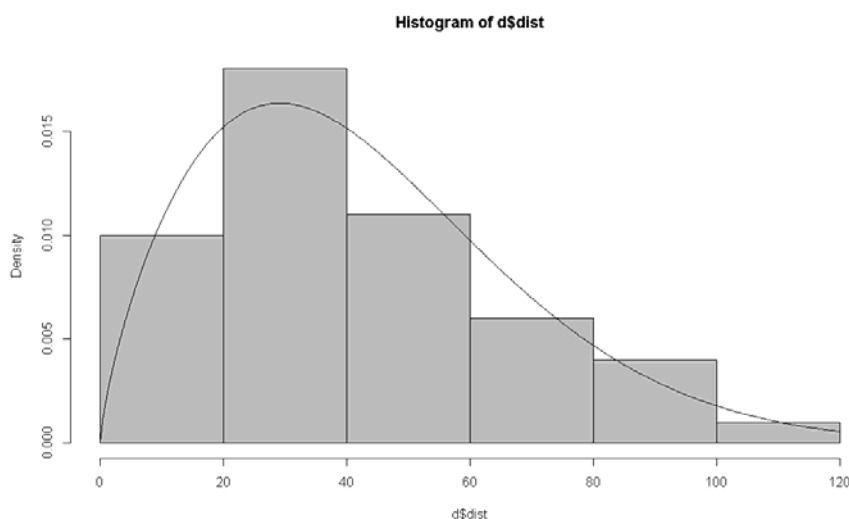


Рис. 10. Гистограмма распределения переменной `dist`

Первый аргумент функции `lines` — это значения по оси абсцисс, на основе которых будет построен график.

Далее указывается функция плотности `dweibull`. Для нее нужно указать значения аргумента для расчета и значения двух параметров распределения: коэффициент формы (`shape`) и масштаба (`scale`).

Контрольные вопросы

1. Как найти среднее выборочное значение и стандартное отклонение?
2. Как проверить наличие пропусков в исходных данных?
3. Каким образом функция `qplot` разбивает выборку на интервалы?

ЛАБОРАТОРНАЯ РАБОТА 2. ЛИНЕЙНАЯ РЕГРЕССИЯ

Цель работы

Изучить приемы исследования корреляционной зависимости, построения парной и множественной линейной регрессии.

Задание

1. Загрузить набор данных для своего варианта, ознакомиться с его содержимым.
2. Построить график корреляционного поля для каждого фактора.
3. Построить уравнение парной линейной регрессии для каждого фактора.
4. Проверить значимость каждого из полученных уравнений регрессии. Показать уравнения регрессии с заданным в варианте доверительным интервалом на графиках.
5. Построить прогнозы по каждому из уравнений парной регрессии для заданных в варианте значений факторов.
6. Построить уравнение множественной линейной регрессии и получить корреляционную матрицу.
7. Построить прогноз по уравнению множественной регрессии для заданных в варианте значений факторов.

Указания к выполнению работы

Парная линейная регрессия

Рассмотрим построение парной линейной регрессии на встроенном наборе данных `cars`.

```
d <- cars
```

Будем рассматривать зависимость длины тормозного пути (переменная `dist`) от скорости (переменная `speed`).

Построим график зависимости длины тормозного пути от скорости автомобиля:

```
qplot(data=d, speed, dist)
```

Чтобы настроить внешний вид графика, необходимо использовать функцию `ggplot` (рис. 11).

```
ggplot() +  
  geom_point(aes(x=d$speed, y=d$dist), size = 2) + theme_bw(base_size =  
18) +  
  xlab("Скорость, миль/ч") + ylab("Длина тормозного пути, футы") +  
  labs(title = "Корреляционное поле")
```

Данная функция имеет множество других настроек, с которыми можно ознакомиться в справке [6]. Оценим модель линейной регрессии длины тормозного пути на скорость автомобиля.

Для этого командой `lm` поместим в переменную `model` модель линейной регрессии, указав `dist` в качестве зависимой переменной, и через значок `~` переменную `speed` в качестве регрессора:

```
model <- lm(data=d, dist~speed) # базовый пакет stats
```

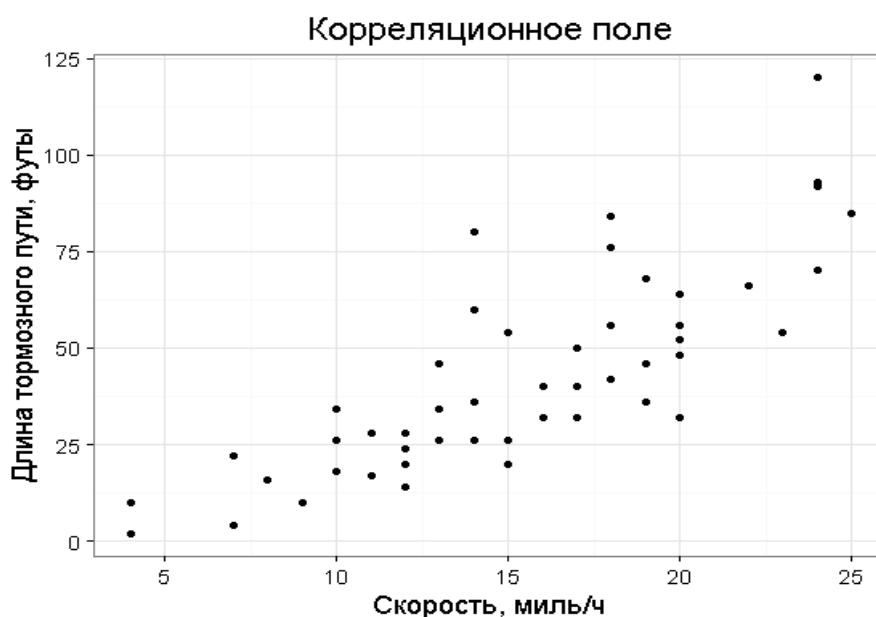


Рис. 11. График зависимости длины тормозного пути `dist` от скорости автомобиля `speed`

Тип `lm` представляет собой список из 12 элементов (рис. 12).

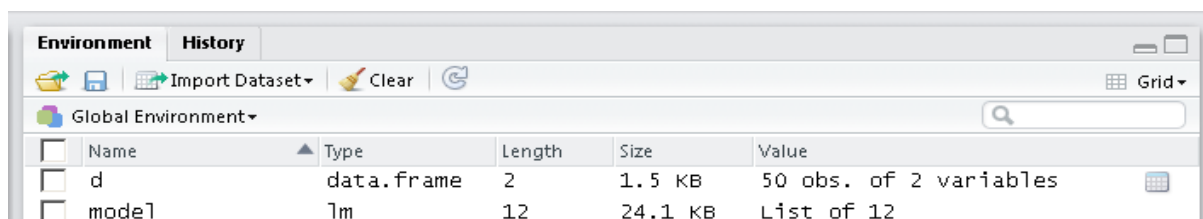


Рис. 12. Переменная-список `lm` в в таблице среды Environment

Посмотрим на коэффициенты уравнения линейной регрессии:

```
model$coefficients
```

Результат в консоли:

```
> model$coefficients
(Intercept)      speed 
-17.579095      3.932409
```

(Intercept) — это константа в уравнении регрессии, speed — коэффициент регрессии.

Таким образом, уравнение регрессии имеет вид:

$$dist_i^m = -17.579 + 3.9324 \cdot speed_i$$

Также можно посмотреть значения вектора ошибок модели — разницу между реальной длиной тормозного пути `dist` и полученной по модели $dist_i^m$. Выведем первые 10 значений этого вектора с точностью две цифры после запятой:

```
model$residuals[1:10]
options(digits = 3)
      1      2      3      4      5      6      7      8      9     10
 3.85 11.85 -5.95 12.05  2.12 -7.81 -3.74  4.26 12.26 -8.68
```

Более полный набор расчетов по модели можно получить командой `summary`:

```
summary(model) # базовый пакет base
```

Call:

```
lm(formula = dist ~ speed, data = d)
```

Residuals:

```
      Min       1Q   Median       3Q      Max 
-29.069  -9.525  -2.272   9.215  43.201
```

Coefficients:

```
              Estimate Std. Error t value Pr(>|t|)
(Intercept) -17.5791      6.7584  -2.601   0.0123 *
speed         3.9324      0.4155   9.464 1.49e-12 ***
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Residual standard error: 15.38 on 48 degrees of freedom
 Multiple R-squared: 0.6511, Adjusted R-squared: 0.6438
 F-statistic: 89.57 on 1 and 48 DF, p-value: 1.49e-12

Помимо коэффициентов регрессии, R выводит:

- стандартные ошибки коэффициентов (Std. Error);
- наблюдаемые значения t-критерия при проверке значимости коэффициентов регрессии (t value);
- Р-значения для коэффициентов регрессии (P-value).

Звездочками или точками в столбце справа R показывает значимость или незначимость коэффициентов: *** — значимы на уровне значимости менее 0.001; ** — значимы на уровне значимости 0.001; * — значимы на уровне значимости 0.01; . — значимы на уровне значимости 0.05 и т. д. Эти обозначения приведены в разделе Signif. codes.

Коэффициент детерминации (Multiple R-squared) равен 0.6511; скорректированный коэффициент детерминации (Adjusted R-squared) равен 0.6438. Наблюдаемое значения F-критерия проверки значимости уравнения в целом и Р-значение:

F-statistic: 89.57 on 1 and 48 DF, p-value: 1.49e-12

Таким образом, уравнение регрессии получилось значимым.

Проведем на графике полученную линию регрессии с 95% доверительными интервалами (рис. 13):

```
qplot(data = d, speed, dist) + stat_smooth(method="lm", level = 0.95) +  
  theme_bw(base_size = 18)
```

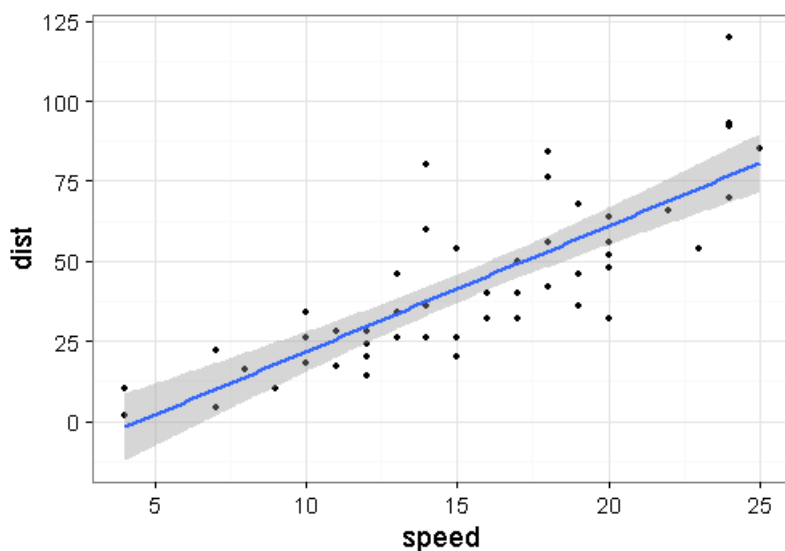


Рис. 13. График линейной регрессии с доверительными интервалами

Рассчитаем также 95% доверительные интервалы для параметров линейной регрессии:

```
> confint(model, level = 0.95) # базовый пакет stats
```

```

                2.5 %      97.5 %
(Intercept) -31.167850 -3.990340
speed        3.096964  4.767853

```

Рассчитанные по модели значения $dist_i^m = -17.579 + 3.9324 \cdot speed_i$ можно получить командой

```
fitted(model) # базовый пакет stats
```

Необъясненная сумма квадратов отклонений:

```
RSS <- deviance(model) # базовый пакет stats
```

Можно также рассчитать полную сумму квадратов, воспользовавшись уже известными функциями `sum` и `mean`:

```
TSS <- sum((y-mean(y))^2)
```

Для того чтобы построить прогноз по полученной модели, нужно задать значения регрессора и поместить их в новый `data.frame`.

```
# создаем новый набор данных
nd <- data.frame(speed=c(40,60))
```

Строим прогноз функцией `predict`:

```
> predict(model, nd)
      1      2
139.7173 218.3654
```

Множественная линейная регрессия

Рассмотрим встроенный набор данных по социально-экономическим показателям в 47 провинциях Швейцарии в 1888 г.

```
t <- swiss # встроенный набор данных по Швейцарии
```

Этот набор данных содержит 6 переменных по 47 наблюдений, каждая из которых измеряется в процентах (`help(swiss)`):

Fertility — рождаемость;

Agriculture — % мужчин, занятых в сельском хозяйстве;

Examination — % призывников, получивших высшую оценку на экзамене в армии;

Education — % призывников, имеющих образование помимо начального;

Catholic — % католиков среди населения;

Infant.Mortality — % детей, умерших до года.

Посмотрим на этот набор данных:

```
> glimpse(t)
Observations: 47
Variables: 6
$ Fertility      (dbl) 80.2, 83.1, 92.5, 85.8, 76.9, 76.1, 83.8, 92.4...
$ Agriculture    (dbl) 17.0, 45.1, 39.7, 36.5, 43.5, 35.3, 70.2, 67.8...
$ Examination    (int) 15, 6, 5, 12, 17, 9, 16, 14, 12, 16, 14, 21, 1...
$ Education      (int) 12, 9, 5, 7, 15, 7, 7, 8, 7, 13, 6, 12, 7, 12,...
$ Catholic       (dbl) 9.96, 84.84, 93.40, 33.77, 5.16, 90.57, 92.85,...
$ Infant.Mortality (dbl) 22.2, 22.2, 20.2, 20.3, 20.6, 26.6, 23.6, 24.9...
```

Встроенный пакет `graphics` содержит функцию `pairs`, позволяющую получить все возможные диаграммы рассеяния на одном графике, а также выполнить их сглаживание с помощью опции `panel.smooth`:

```
pairs(swiss, panel = panel.smooth)
```

Результатом будет график, показанный на рисунке 14.

Функция `cor` позволяет как вычислить корреляцию между двумя выборками, так и получить корреляционную матрицу для всех переменных из набора данных:

```
> cor(swiss)
```

	Fertility	Agriculture	Examination	Education	Catholic
Fertility	1.000	0.3531	-0.646	-0.6638	0.464
Agriculture	0.353	1.0000	-0.687	-0.6395	0.401
Examination	-0.646	-0.6865	1.000	0.6984	-0.573
Education	-0.664	-0.6395	0.698	1.0000	-0.154
Catholic	0.464	0.4011	-0.573	-0.1539	1.000
Infant.Mortality	0.417	-0.0609	-0.114	-0.0993	0.175

	Infant.Mortality
Fertility	0.4166
Agriculture	-0.0609
Examination	-0.1140
Education	-0.0993
Catholic	0.1755
Infant.Mortality	1.0000

Корреляционную матрицу можно получить и в других видах, например, с помощью функции `sjp.corr` из пакета `sjPlot` (рис. 15):

```
library("sjPlot")
sjp.corr(t)
```

Существует еще одна функция, позволяющая получить корреляционную матрицу, диаграммы рассеяния и сглаженные распределения одновременно (рис. 16):

```
library("GGally")
ggpairs(t) # функция из пакета GGally
```


Чтобы оценить регрессию рождаемости на остальные переменные, можно воспользоваться уже знакомой функцией `lm`, а регрессоры перечислить через знак «плюс»:

```
model2 <- lm(data=t, Fertility~Agriculture+Education+Catholic)
```

В данном случае регрессорами стали % занятых в с/х; % католического населения и % имеющих образование выше начального.

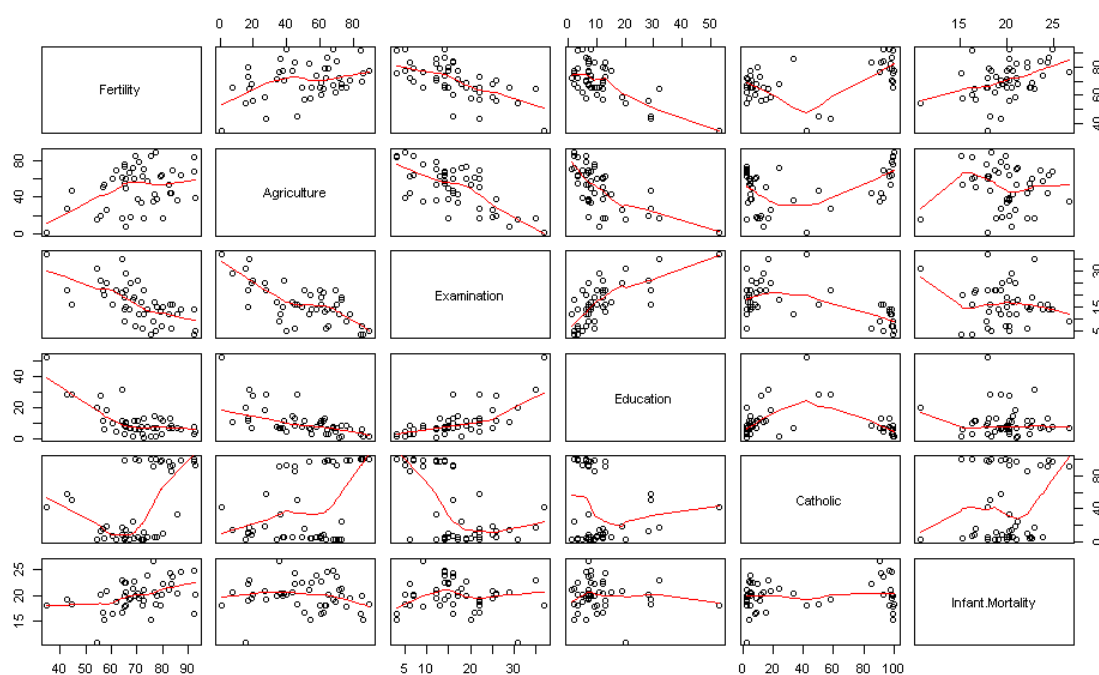


Рис. 14. Диаграммы рассеяния, полученные с помощью функции `pairs`

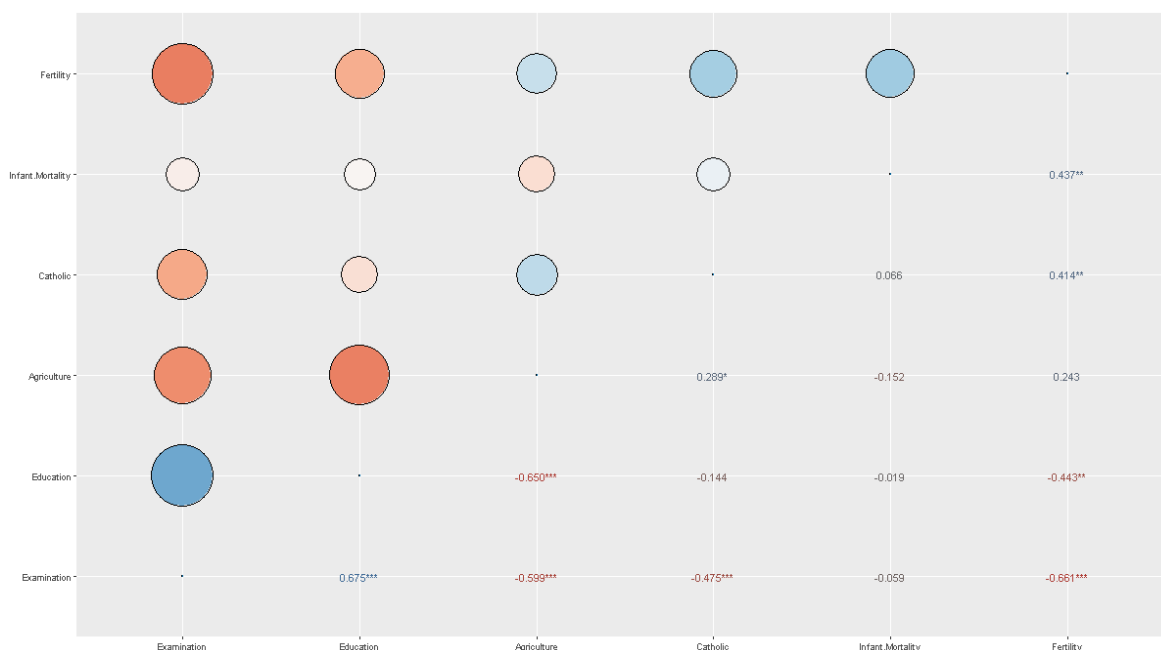


Рис. 15. Корреляционная матрица, полученная с помощью функции `sjp.corr`

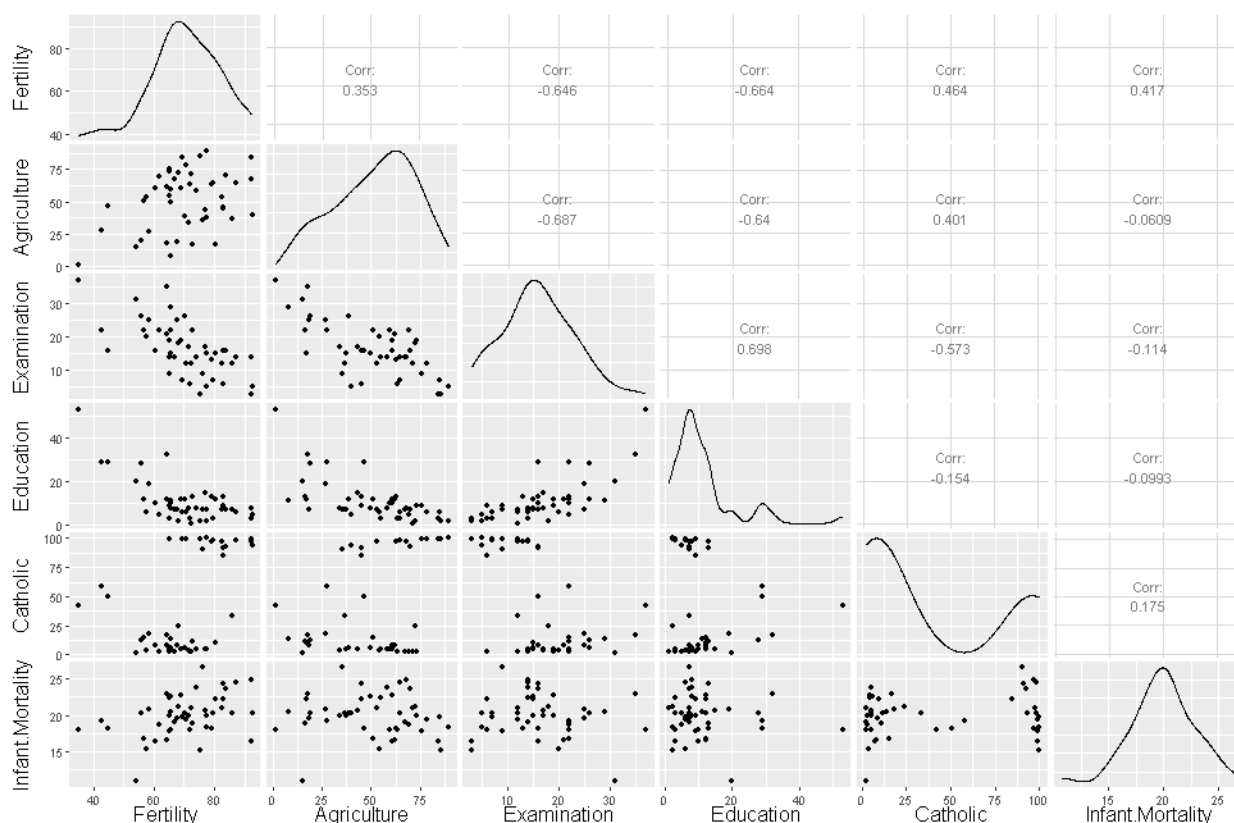


Рис. 16. Корреляционная матрица, диаграммы рассеяния и сглаженные распределения, полученные с помощью функции `ggpairs`

Получить оценки коэффициентов уравнения регрессии, а также проверить основные гипотезы поможет функция `summary`:

```
> summary(model2)
```

Call:

```
lm(formula = Fertility ~ Agriculture + Education + Catholic,
    data = t)
```

Residuals:

Min	1Q	Median	3Q	Max
-15.178	-6.548	1.379	5.822	14.840

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	86.22502	4.73472	18.211	< 2e-16 ***
Agriculture	-0.20304	0.07115	-2.854	0.00662 **
Education	-1.07215	0.15580	-6.881	1.91e-08 ***
Catholic	0.14520	0.03015	4.817	1.84e-05 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 7.728 on 43 degrees of freedom

Multiple R-squared: 0.6423, Adjusted R-squared: 0.6173

F-statistic: 25.73 on 3 and 43 DF, p-value: 1.089e-09

Построим прогноз по аналогии с парной линейной регрессией. Отличие заключается лишь в том, что в наборе данных необходимо указать значения каждого фактора:

```
# создаем новый набор данных
nd2 <- data.frame(Agriculture=0.5,Catholic=0.5, Education=20)
> predict(model2,nd2)
      1
64.75316
```

Построение прогноза по нескольким точкам выполняется с помощью векторов значений:

```
# создаем новый набор данных
nd2 <- data.frame(Agriculture=c(0.5,0.8),Catholic=c(0.5, 0.65),
                  Education=c(20, 25))
# прогнозируем
predict(model2,nd2)
      1      2
64.75316 59.35330
```

Контрольные вопросы

1. Что такое Intercept в парной линейной регрессии?
2. Как проверить значимость уравнения регрессии в целом?
3. Как узнать разброс значений коэффициентов регрессии?

ЛАБОРАТОРНАЯ РАБОТА 3. ВРЕМЕННЫЕ РЯДЫ

Цель работы

Изучить базовые способы анализа, моделирования и прогнозирования временных рядов.

Задание

1. Загрузить данные из временного ряда для своего варианта за последние полгода.
2. Построить график временного ряда с графиками автокорреляционной и частной автокорреляционной функций.
3. Подобрать вручную порядок ARMA-модели. Для подбора использовать визуальный анализ графика.
4. Подобрать ARMA-модель автоматически.
5. Построить прогноз на указанное в варианте число шагов. Показать прогноз на графике.

Указания к выполнению работы

Анализ временных рядов

Для работы подключим следующие пакеты:

```
library("lubridate") # работа с датами
library("zoo") # работа с временными рядами
library("xts") # дополнительные функции для работы с временными рядами
library("dplyr") # работа с наборами данных
library("ggplot2") # графики
library("forecast") # прогнозы
library("lmtest") # тестирование гипотез в линейных моделях
library("quantmod") # загрузка данных с различных источников
```

Для начала посмотрим, как сгенерировать тестовые выборки. Для этого воспользуемся функцией `arima.sim` из базового пакета `stats`.

Выполним симуляцию AR(1)-процесса в 100 наблюдений по модели: $y_t = 0.6y_{t-1} + \varepsilon_t$ и поместим в переменную `y`.

```
y <- arima.sim(n=100, list(ar=0.6))
```

В этой функции необходимо указать объем выборки, а также в параметре `list` коэффициенты модели.

Можно построить полученный ряд на графике (рис. 17):

```
plot(y)
```

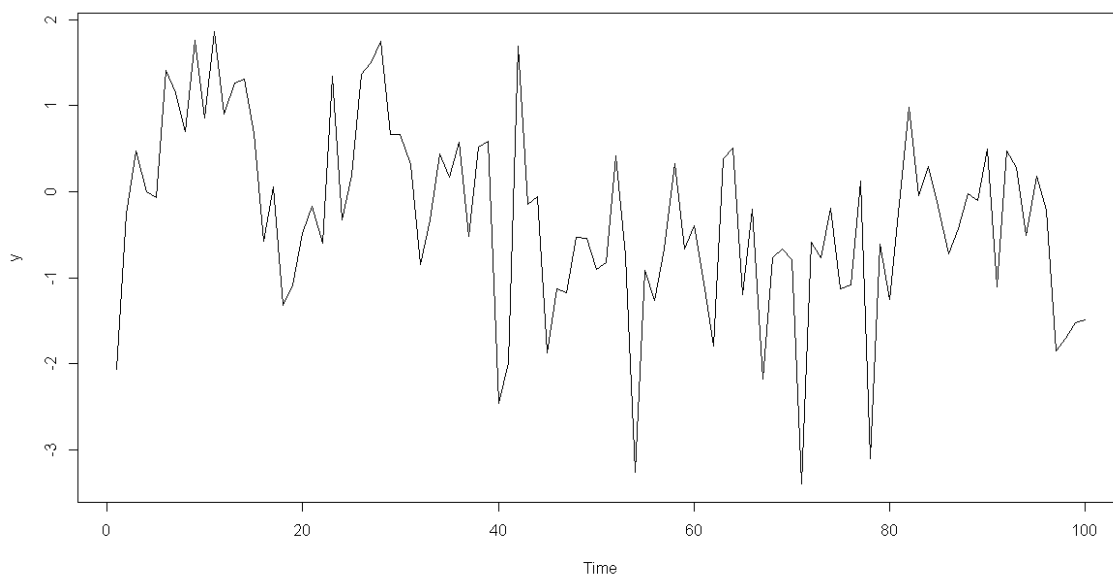


Рис. 17. График временного ряда

Графики автокорреляционной и частной автокорреляционной функций для процесса `y` вызываются командами соответственно:

```
Acf(y)
Pacf(y)
```

Все три графика одновременно могут быть вызваны командой:

```
tsdisplay(y)
```

Результат выполнения команды показан на рисунке 18.

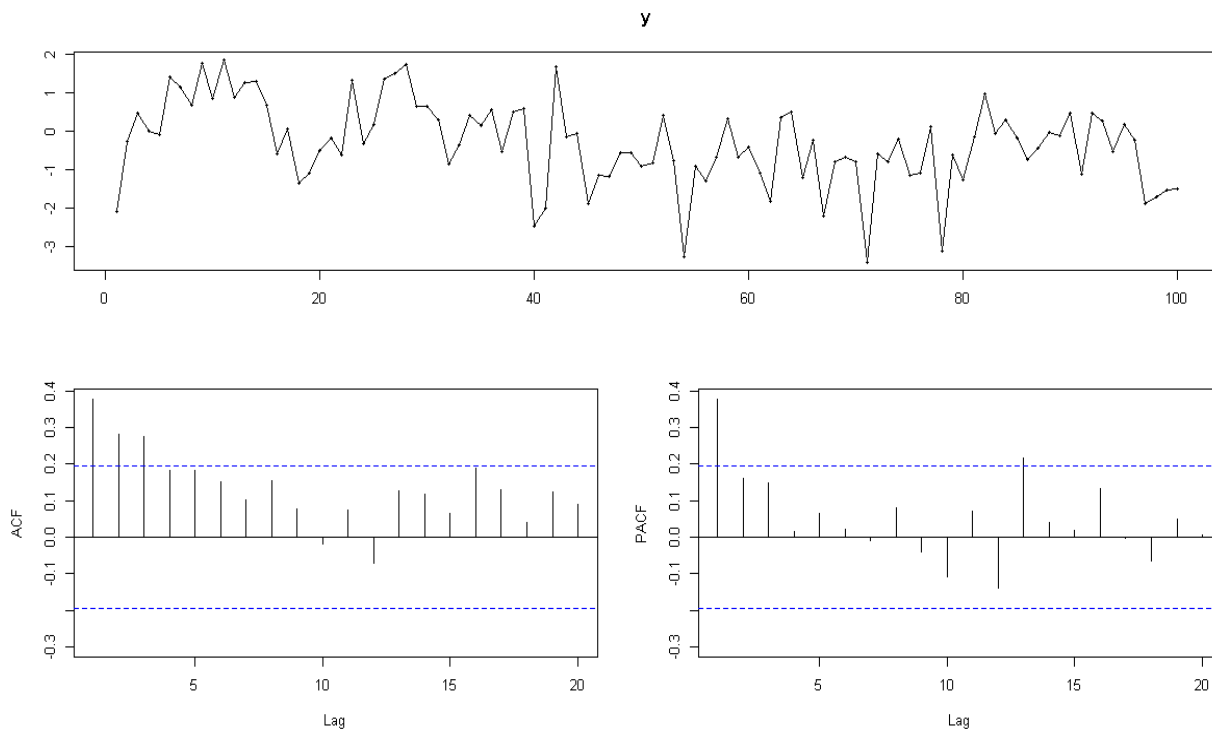


Рис. 18. График временного ряда с графиками автокорреляционной и частной автокорреляционной функций

Посмотрим еще несколько примеров симуляции AR, MA, ARMA, ARIMA процессов.

Процесс MA(1) $y_t = \varepsilon_t - 0.8\varepsilon_{t-1}$

```
arima.sim(n=100, list(ma=-0.8))
```

Процесс ARMA(1,1) $y_t = 0.5y_{t-1} + \varepsilon_t - 0.8\varepsilon_{t-1}$

```
arima.sim(n=100, list(ma=-0.8, ar=0.5))
```

Процесс ARMA(2,2) $y_t = 0.9y_{t-1} - 0.5y_{t-2} + \varepsilon_t - 0.2\varepsilon_{t-1} + 0.3\varepsilon_{t-2}$

```
arima.sim(n = 100, list(ar = c(0.9, -0.5), ma = c(-0.2, 0.3))
```

Процесс случайного блуждания $y_t = y_{t-1} + \varepsilon_t$

```
arima.sim(n=100, list(order=c(0,1,0)))
```

Белый шум $y_t = \varepsilon_t$:

```
arima.sim(n=100, list(order=c(0,0,0)))
```

Попробуем подобрать различные ARIMA-модели под реальные данные. Для примера выберем временной ряд, содержащий 98 еже-

годных наблюдений за уровнями воды в озере Гурон с 1875 по 1972 гг. (рис. 19).

help(LakeHuron)

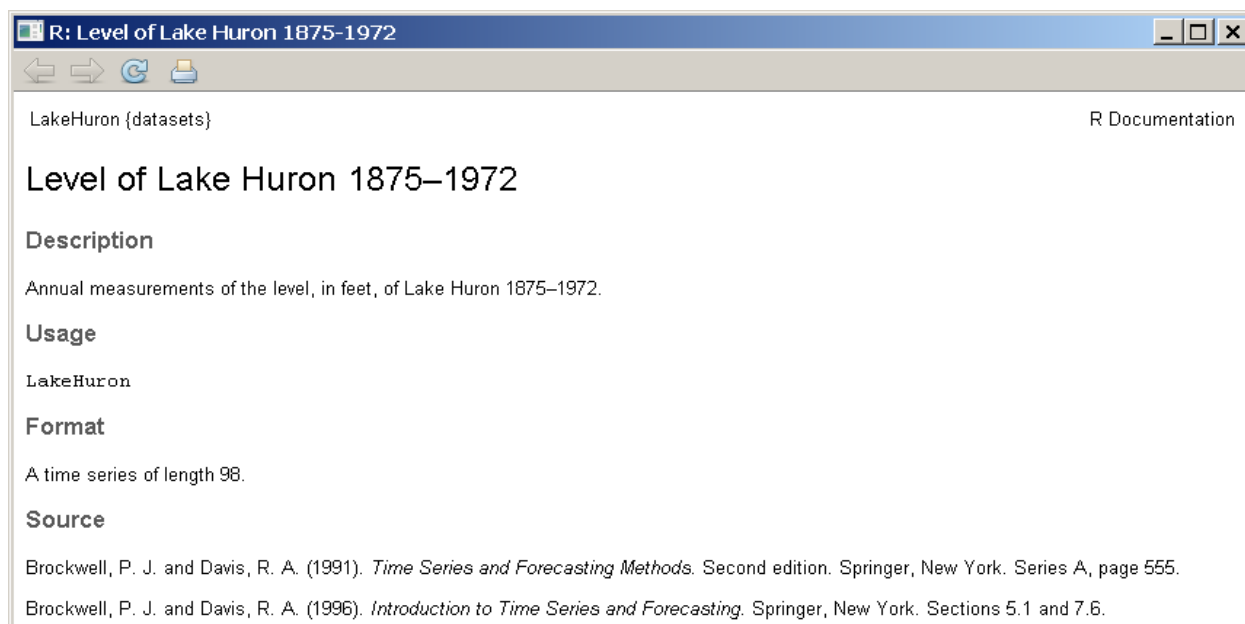


Рис. 19. Справка по набору данных LakeHuron

Оценим ARMA(2,1) модель с помощью функции Arima из пакета forecast:

```
mod<- Arima(y, order=c(2,0,1))
```

Посмотрим на результат оценивания:

```
summary(mod)
Series: y
ARIMA(2,0,1) with non-zero mean
Coefficients:
          ar1      ar2      ma1  intercept
0.7829 -0.0342  0.2857   579.0533
s.e.   0.3262   0.2845   0.3144     0.3467
sigma^2 estimated as 0.4749:  log likelihood=-103.24
AIC=216.48  AICc=217.13  BIC=229.4
Training set error measures:
              ME      RMSE      MAE      MPE      MAPE      MASE
Training set -0.008638553 0.6891058 0.5492008 -0.001635858 0.09486097 0.9378958
              ACF1
Training set 0.002260998
```

Получили модель $y_t = 579.0533 + 0.7829y_{t-1} - 0.0342y_{t-2} + \varepsilon_t + 0.2857\varepsilon_{t-1}$.

В строке s.e. перечислены стандартные ошибки коэффициентов. Оценка дисперсии $\hat{\sigma}^2 = 0.4749$. Строкой ниже перечислены значения критерия Акаике (AIC=216.48), скорректированного критерия Акаике (AICc=217.13) и критерия Шварца (BIC=229.4).

Можно отдельно вывести значение критерия Акаике командой:

```
AIC(mod)
```

Командой `fitted(mod)` можно получить модельные значения временного ряда. Построим реальные и модельные значения на одном графике (рис. 20):

```
matplot(cbind(y, fitted(mod)), type='l')
```

Функция `matplot` строит значения столбцов матрицы, поэтому ее аргументом нужно задать столбцы. Команда `cbind` соединяет в матрицу столбец `y` и столбец значений, рассчитанных по модели. `type='l'` указывает, что тип графика — линия.

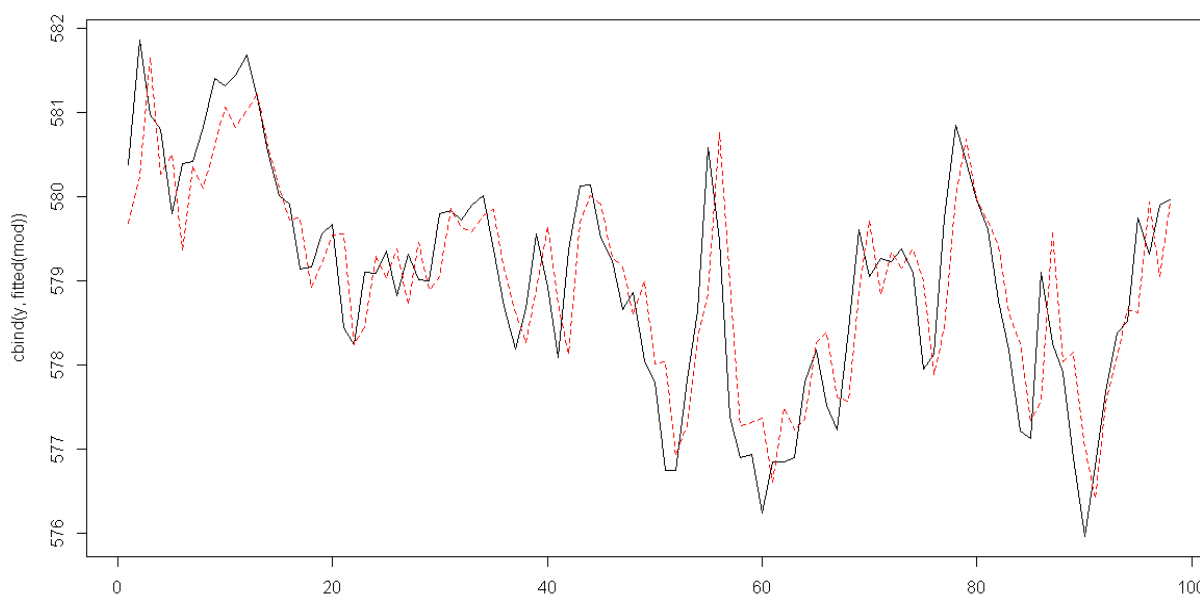


Рис. 20. График модельных и реальных значений временного ряда

Построим прогноз на 5 шагов вперед:

```
prognosz<- forecast(mod, h=5)
```

Выведем полученные значения. Результат включает 80% и 95% доверительный интервал для прогноза:

```
prognosz
      Point Forecast      Lo 80      Hi 80      Lo 95      Hi 95
1973      579.7407 578.8576 580.6238 578.3901 581.0913
1974      579.5605 578.2680 580.8530 577.5838 581.5372
1975      579.4269 577.9528 580.9009 577.1725 581.6813
1976      579.3284 577.7645 580.8924 576.9366 581.7203
1977      579.2559 577.6453 580.8666 576.7927 581.7192
```

Строим график прогноза (рис. 21):

`plot(prognoz)`

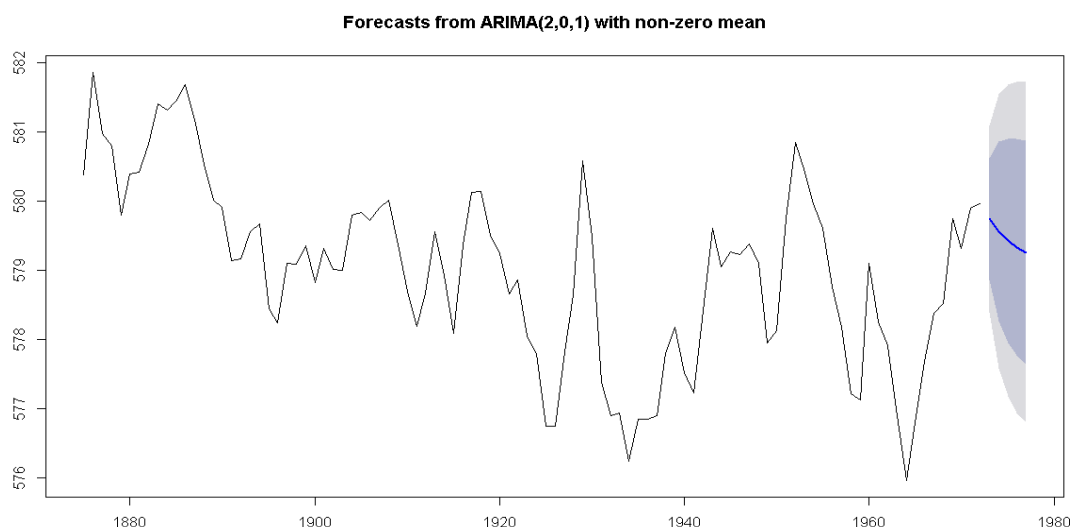


Рис. 21. График прогноза по модели ARIMA

R также может автоматически подбирать ARIMA-модель по штрафному критерию (минимальное значение критерия Акаике):

```
mod_a<- auto.arima(y)
summary(mod_a)
Series: y
ARIMA(0,1,4) with drift
Coefficients:
      ma1      ma2      ma3      ma4      drift
0.0584 -0.3158 -0.3035 -0.2349 -0.0205
s.e.  0.1031  0.1058  0.1100  0.1299  0.0167
sigma^2 estimated as 0.4806:  log likelihood=-101.09
AIC=214.18  AICc=215.12  BIC=229.63
Training set error measures:
              ME          RMSE          MAE          MPE          MAPE
MASE
Training set  0.002898915  0.6886758  0.5441565  0.0003830464  0.093983
0.9292814
              ACF1
Training set  0.01081339
```

«ARIMA(0,1,4) with drift» означает, что была подобрана ARIMA(0,1,4) с линейным трендом (drift), но без константы (отсутствует параметр intercept).

Таким образом, получим модель

$$y_t = y_{t-1} + \varepsilon_t + 0.0584\varepsilon_{t-1} - 0.3158\varepsilon_{t-2} - 0.3035\varepsilon_{t-3} - 0.2349\varepsilon_{t-4} - 0.0205t.$$

Загрузка данных из различных источников в R

Для корректного перевода дат на английский язык в русскоязычной Windows, необходимо сначала выполнить следующую команду:

```
Sys.setlocale("LC_TIME", "C")
```

Пакет `quantmod` позволяет загружать данные из нескольких источников, а именно:

- Yahoo! Finance (OHLC data) — <http://finance.yahoo.com/>;
- Federal Reserve Bank of St. Louis FRED® (11,000 экономических временных рядов) — <http://research.stlouisfed.org/fred2/>;
- Google Finance (OHLC data) — <http://finance.google.com/>;
- Oanda, The Currency Site (FX and Metals) — <http://www.oanda.com/>.

Для загрузки данных используется одна и та же функция `getSymbols`, например:

```
#данные о стоимости акций Гугл с finance.google.com за период
с 1 января 2015г. по 1 декабря 2015г.
getSymbols("GOOG", src="google", from="2015-01-01", to="2015-12-01")
```

GOOG — это краткое название акций компании на бирже (тикер). Теперь эти данные загружены в переменную с таким же названием GOOG.

Если не указывать начальную или конечную дату, то будут загружены все доступные данные. Можно посмотреть шесть первых значений и шесть последних значений командами `head` и `tail` соответственно.

```
head(GOOG)
```

	GOOG.Open	GOOG.High	GOOG.Low	GOOG.Close	GOOG.Volume	GOOG.Adjusted
2015-01-02	529.0124	531.2724	524.1023	524.8124	1447600	524.8124
2015-01-05	523.2624	524.3324	513.0623	513.8723	2059800	513.8723
2015-01-06	515.0024	516.1773	501.0523	501.9623	2899900	501.9623
2015-01-07	507.0023	507.2463	499.6522	501.1023	2065100	501.1023
2015-01-08	497.9922	503.4823	491.0022	502.6823	3353600	502.6823
2015-01-09	504.7623	504.9223	494.7922	496.1723	2069400	496.1723

```
tail(GOOG)
```

	GOOG.Open	GOOG.High	GOOG.Low	GOOG.Close	GOOG.Volume	GOOG.Adjusted
2015-11-23	757.45	762.708	751.82	755.98	1414500	755.98
2015-11-24	752.00	755.279	737.63	748.28	2333100	748.28
2015-11-25	748.14	752.000	746.06	748.15	1122100	748.15
2015-11-27	748.46	753.410	747.49	750.26	838500	750.26
2015-11-30	748.81	754.930	741.27	742.60	2035300	742.60
2015-12-01	747.11	768.950	746.70	767.04	2129900	767.04

Посмотрим на график (рис. 22):

```
barChart(GOOG, theme = "white")
```



Рис. 22. График стоимости акций Гугл GOOG

Также можно посмотреть на цену закрытия с помощью функции `tsdisplay` (рис. 23):

```
tsdisplay(GOOG$GOOG.Close)
```

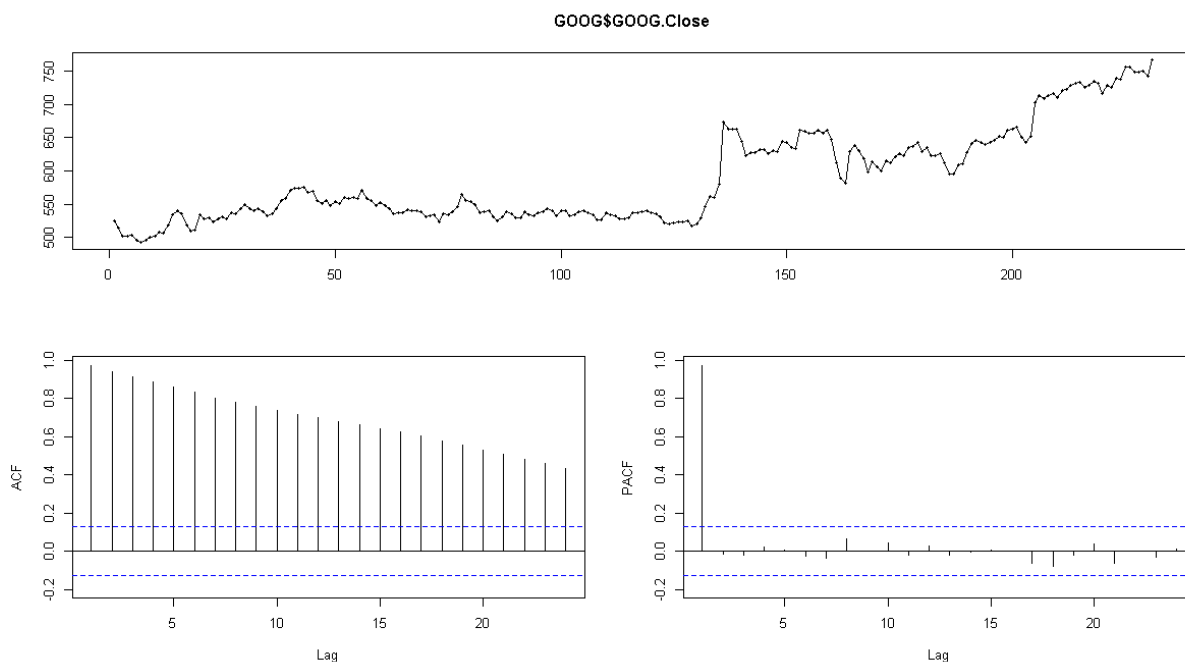


Рис. 23. График цены закрытия акций Гугл GOOG

Приведем еще несколько примеров:

```
#данные о стоимости акций Yahoo с finance.google.com
getSymbols("YHOO",src="google")
#данные о стоимости акций Apple с finance.yahoo.com
getSymbols("AAPL",src="yahoo", from="2015-01-01", to="2015-12-01")
```

Японские свечи (рис. 24):

```
candleChart(AAPL, multi.col=TRUE, theme="white")
```

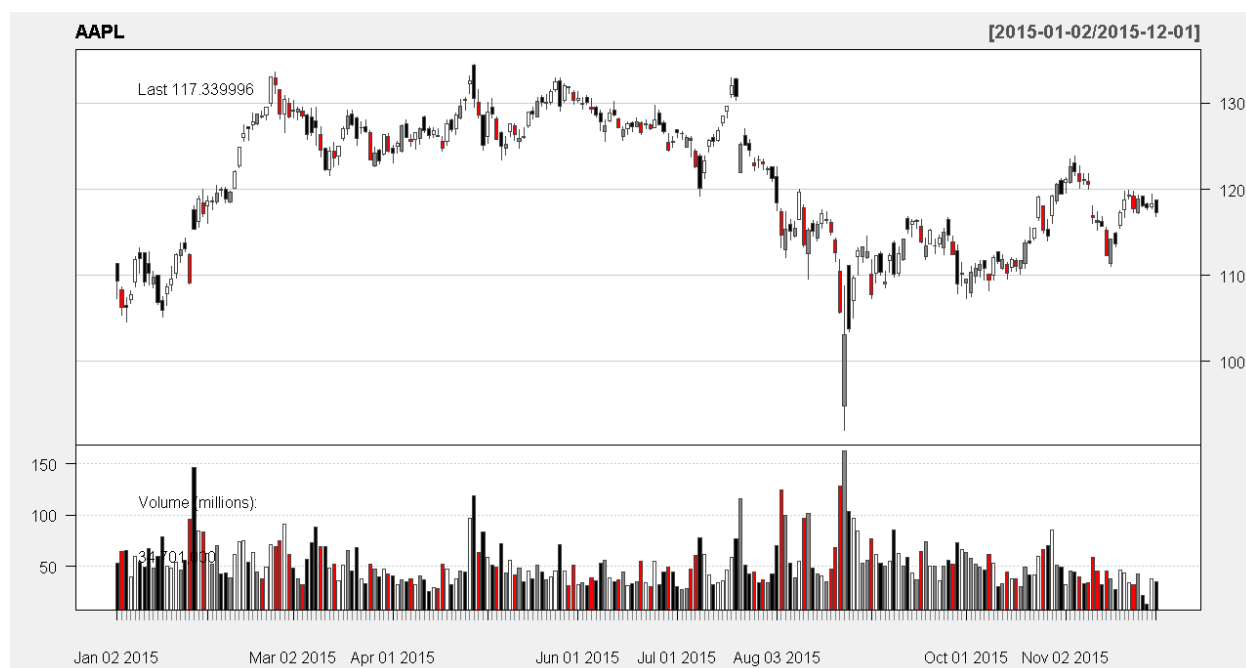


Рис. 24. График стоимости акций Apple AAPL

Для более детальной информации можно обратиться на сайт разработчиков пакета [7].

Контрольные вопросы

1. В чем отличие автокорреляционной и частной автокорреляционной функций?
2. Что такое случайное блуждание и белый шум?
3. Как можно сравнить модели временного ряда по точности?

ЛАБОРАТОРНАЯ РАБОТА 4. ГЕНЕТИЧЕСКИЕ АЛГОРИТМЫ

Цель работы

Научиться применять генетические алгоритмы (ГА) [5] для поиска оценок параметров нелинейных моделей.

Задание

1. Выполнить генерацию значений x и y .
2. Построить график зависимости y от x .
3. Задать функцию зависимости y от x в R и соответствующую ей функцию МНК.
4. Определить минимальные и максимальные значения параметров.
5. Запустить поиск решения с помощью ГА. Размер популяции, предельное число итераций и условие останова алгоритма заданы в варианте.
6. Построить график значений *fitness*-функции и график с исходными данными и полученной моделью.
7. Повторить пункты 5 и 6 еще 2 раза и сравнить полученные результаты.
8. Выбрать наилучший из трех результатов по значению коэффициента детерминации.

Указания к выполнению работы

Генетические алгоритмы

Рассмотрим решение задачи нелинейного МНК с помощью генетического алгоритма на встроенном наборе данных *trees*.

Для работы подключим следующие пакеты [10]:

```
library("GA") # генетические алгоритмы
library("dplyr") # работа с наборами данных
library("ggplot2") # графики
library("spuRs") # содержит набор данных trees
```

Получим описание набора данных по деревьям *trees* из пакета *spuRs* командой (рис. 25):

```
help("trees", package = "spuRs")
```

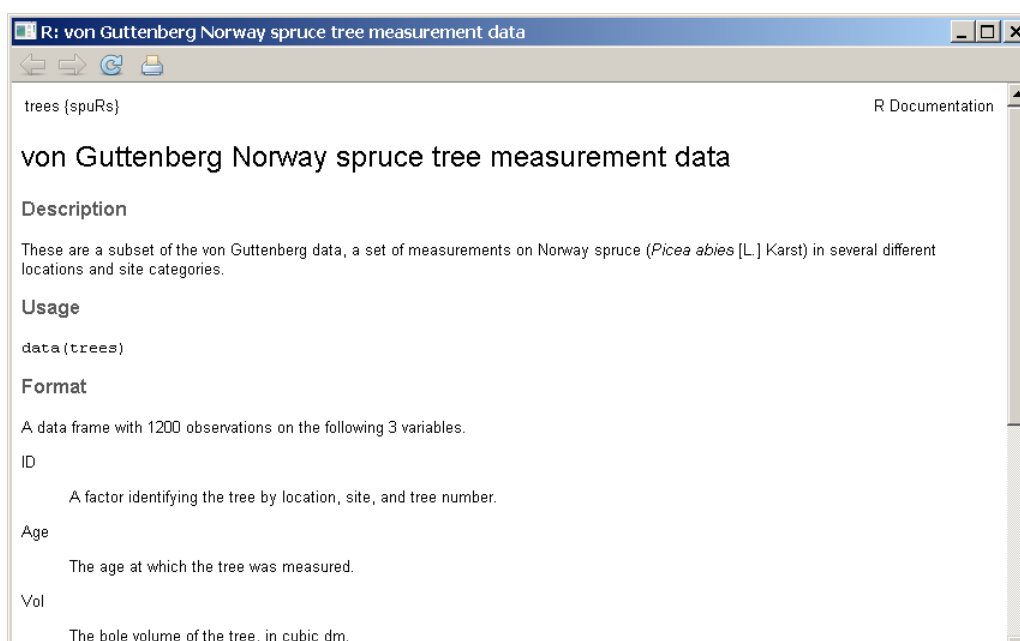


Рис. 25. Справка по набору данных `trees` из пакета `spuRs`

В этом наборе данных содержится 1200 наблюдений и три переменных (`ID` дерева, который включает номер местоположения, площадки и номер дерева; возраст дерева; объем в куб. дм). Будем рассматривать зависимость объема дерева (переменная `Vol`) от возраста (переменная `Age`).

Активируем набор данных командой:

```
data("trees", package = "spuRs")
```

Посмотрим на этот набор данных:

```
> glimpse(trees)
Observations: 1200
Variables:
$ ID   (fctr) 1.1.1, 1.1.1, 1.1.1, 1.1.1, 1.1.1, 1.1.1, 1.1.1, 1.1.1,
1.1.1, 1.1.1, 1....
$ Age  (dbl)  9.67, 19.67, 29.67, 39.67, 49.67, 59.67, 69.67, 79.67,
89.67, 99.67, 109....
$ Vol  (dbl)  5.0, 38.0, 123.0, 263.0, 400.0, 555.0, 688.0, 820.0, 928.0,
1023.0, 1104....
```

Выберем для дальнейшей работы только данные для деревьев с определенным местоположением, например, `ID = 1.3.11`¹. Поместим в переменную `tree` выбранную часть исходной выборки:

```
tree<- trees[trees$ID == "1.3.11", 2:3]
```

Командой `trees[]` можно выбрать нужные столбцы из набора данных. Таким образом, предыдущая команда выбирает из набора

¹ Подробнее см. в Zeide, B. Analysis of growth equations. Forest Science. 1993. 39 (3). Pp. 549-616.

данных только второй и третий столбцы, но так, чтобы в первом столбце ID был при этом равен 1.3.11.

Посмотрим теперь на tree:

```
> glimpse(tree)
Observations: 12
Variables:
$ Age (dbl) 2.44, 12.44, 22.44, 32.44, 42.44, 52.44, 62.44, 72.44,
82.44, 92.44, 102....
$ Vol (dbl) 2.2, 20.0, 93.0, 262.0, 476.0, 705.0, 967.0, 1203.0,
1409.0, 1659.0, 1898...
```

Получен набор данных из 12 наблюдений с двумя переменными.

Попробуем описать зависимость объема дерева (y) от его возраста (x) с помощью логистической функции Ричардса:

$$y = a(1 - e^{-bx})^c.$$

Затем нужно будет применить метод наименьших квадратов, причем параметры входят в модель нелинейно:

$$a, b, c = \arg \min_{a, b, c} \sum_{i=1}^{12} \left(Vol_i - a(1 - e^{-b \cdot Age_i})^c \right)^2.$$

Минимизировать сумму квадратов ошибок будем численно — с помощью генетического алгоритма. Генетический алгоритм реализован в R функцией `ga` из пакета `GA`.

Функция `ga` максимизирует функцию `fitness`, которую следует задать. Поскольку стоит задача минимизации, то воспользуемся следующим фактом:

$$\arg \max_{\Theta} (\Theta) = \arg \min_{\Theta} (-\Theta).$$

Для этого необходимо задать в R функцию Ричардса — она будет использована для функции `fitness`, указанной в аргументах функции `ga`. Пусть вектор параметров функции обозначен `theta`. Тогда $a = \text{theta}[1]$, $b = \text{theta}[2]$, $c = \text{theta}[3]$. Задаем функцию `richards`:

```
richards<- function(x, theta)
  theta[1] * (1 - exp(-theta[2] * x))^theta[3]
```

В первой строке указан аргумент функции x и вектор параметров `theta`. Через пробел в следующей строке описывается сама функция Ричардса, порядок параметров a, b, c в векторе `theta` должен сохраняться.

Задаем функцию `fit` суммы квадратов ошибок и после пробела поставим перед функцией суммы квадратов ошибок знак минус:

```
fit<- function(theta, x, y) -sum((y - richards(x, theta))^2)
```

В функции `ga` задаются следующие параметры:

`fit` — нужная нам функция для аргумента `fitness`, которую максимизирует генетический алгоритм;

`type` выберем `real-valued`, поскольку возраст деревьев и объем являются действительными числами;

`x` и `y` — это соответственно возраст дерева (`tree$Age`) и объем дерева (`tree$Vol`) — аргументы функции `fitness`;

`min` — это вектор минимальных значений параметров `a`, `b`, `c` функции Ричардса;

`max` — вектор максимальных значений параметров `a`, `b`, `c`;

`crossover` — функция в R, выполняющая кроссовер, т. е. функция, которая образует потомков, объединив часть генетической информации от родителей;

`popsize` — размер популяции;

`maxiter` — максимальное число итераций, после которого работа генетического алгоритма прекращается;

`run` — число последовательных поколений без какого-либо улучшения в значении `fitness`-функции перед остановом алгоритма и т. д.

Запишем результат выполнения функции `ga` в переменную `myGA`, задав значения описанным аргументам:

```
myGA<- ga(type = "real-valued", fitness = fit, x = tree$Age,
  y = tree$Vol, min = c(3000, 0, 2), max = c(4000, 1, 4),
  popSize = 500, crossover = gareal_blxCrossover, maxiter = 5000,
  run = 200, names = c("a", "b", "c"))
```

Функция при работе будет выводить в консоль значения `fit` на каждой итерации алгоритма:

```
Iter = 1 | Mean = -74580573 | Best = -6766.369
Iter = 2 | Mean = -64840448 | Best = -6766.369
Iter = 3 | Mean = -56793683 | Best = -6766.369
Iter = 4 | Mean = -48988112 | Best = -6766.369
Iter = 5 | Mean = -38083470 | Best = -6766.369
Iter = 6 | Mean = -31671411 | Best = -4661.229
Iter = 7 | Mean = -21836845 | Best = -4661.229
...
```

Посмотрим на краткие результаты подбора функции Ричардса на исходные данные:

```
>summary(myGA)
+-----+
|           Genetic Algorithm           |
+-----+
GA settings:
Type           = real-valued
Population size = 500
Number of generations = 5000
Elitism         = 25
Crossover probability = 0.8
Mutation probability = 0.1
Search domain
a b c
Min 3000 0 2
Max 4000 1 4
GA results:
Iterations           = 792
Fitness function value = -2774.113
Solution             =
               a           b           c
[1,] 3589.788 0.01546055 2.786145
```

Таким образом, получено следующее уравнение регрессии:

$$\hat{y} = 3589.788(1 - e^{-0.015x})^{2.786}.$$

Командой `plot(myGA)` можно посмотреть, как изменялись значения `fitness`-функции на протяжении всех 792 итераций алгоритма (рис. 26).

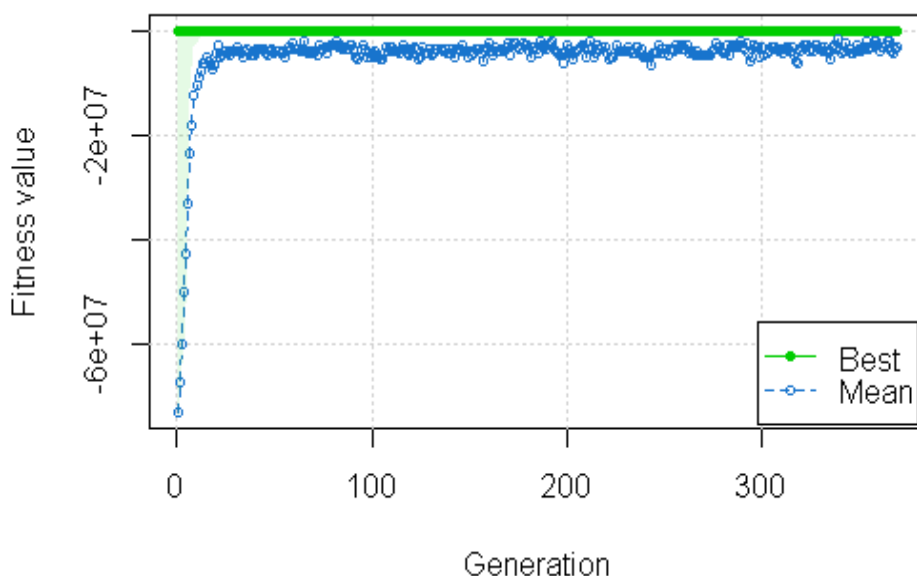


Рис. 26. Изменение значений `fitness`-функции в зависимости от числа итераций

Модельные значения y можно получить командой:

```
richards(tree$Age, myGA@solution)
```

Построим на графике исходную выборку и результат подбора функции Ричардса (рис. 27):

```
ggplot() +  
  geom_point(aes(x=tree$Age, y=tree$Vol)) +  
  geom_line(aes(x=tree$Age, y=richards(tree$Age, myGA@solution)))
```

В данной команде указано:

`ggplot()` активирует построение графика с помощью функций пакета `ggplot2`;

`geom_point(aes(x=tree$Age, y=tree$Vol))` — строит точками зависимость объема дерева от возраста по исходной выборке;

`geom_line(aes(x=tree$Age, y=richards(tree$Age, myGA@solution)))` — строит линией модельные значения объема деревьев в зависимости от возраста, рассчитанные по функции Ричардса с параметрами, найденными генетическим алгоритмом.

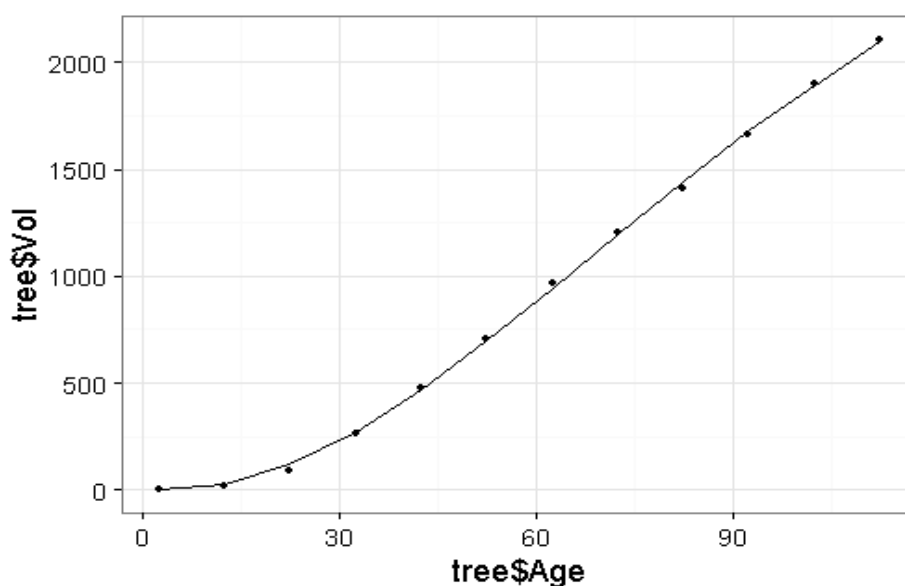


Рис. 27. График исходных данных и полученной функции Ричардса

Для оценки точности модели рассчитаем ее коэффициент детерминации:

```
RSS <- sum((tree$Vol-richards(tree$Age, myGA@solution))^2)  
TSS <- sum((tree$Vol-mean(tree$Vol))^2)  
R2 <- 1-RSS/TSS  
> R2  
[1] 0.9995553
```

Построим по полученной модели прогноз объема дерева для возраста 150 и 200 лет:

```
Age_forecast <- c(150,200)
```

```
richards(Age_forecast, myGA@solution)
[1] 2691.205 3156.228
```

Для построения графика прогноза в виде гладкой линии потребуется большое число точек, которые удобнее задать в виде последовательности от 115 до 200 с шагом 5:

```
Age_forecast <- seq(115, 200, by = 5)
```

```
ggplot() +
  geom_point(aes(x=tree$Age, y=tree$Vol)) +
  geom_line(aes(x=tree$Age, y=richards(tree$Age, myGA@solution))) +
  geom_line(aes(x=Age_forecast, y=richards(Age_forecast, myGA@solution),
    color = "maroon")) +
  theme_bw(base_size = 18)
```

Результат показан на рисунке 28.

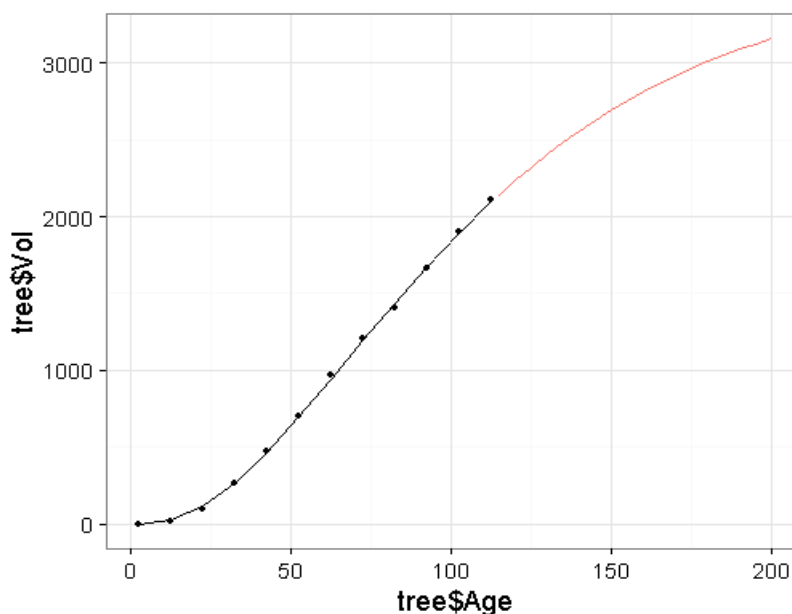


Рис. 28. График исходных данных и полученной функции Ричардса с прогнозом до 200 лет

Генерация данных

Выполним генерацию значений x с нормальным законом распределения, математическим ожиданием 45 и среднеквадратическим отклонением (СКО) 7. Объем выборки зададим равным 100:

```
x <- rnorm(n = 100, mean = 45, sd = 7)
```

Для генерации y необходимо задать формулу зависимости, например, линейную, и наложить на нее случайную помеху ε :

```
a <- 20
```

```
b <- 5
```

```
y <- a + b*x
```

Генерацию ϵ также выполним по нормальному закону распределения, с нулевым математическим ожиданием и единичной дисперсией. Однако, поскольку генераторы случайных чисел неидеальны, необходимо дополнительно выполнить нормировку и центрирование значений:

```
er <- rnorm(n = 100, mean = 0, sd = 1)
er <- (er - mean(er)) / sd(er)
```

После этого можно наложить помеху на формулу зависимости с заданным СКО, равным 2:

```
Ser <- 5
y <- y + er*Ser
```

В результате получим зависимость, показанную на рисунке 29.

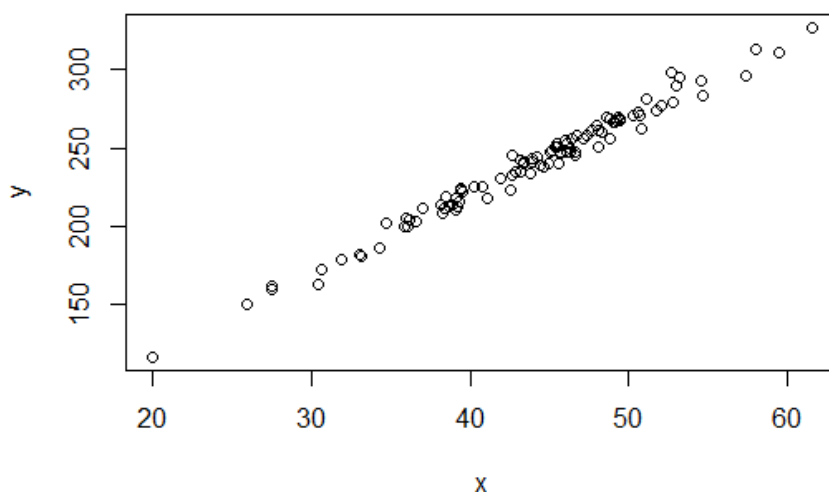


Рис. 29. Сгенерированная зависимость y от x

Контрольные вопросы

1. Максимизирует или минимизирует генетический алгоритм в R `fitness`-функцию?
2. Как задать в R функцию $y = a(x - b)^3 + c$?
3. Каким образом в R можно сгенерировать выборку со стандартным нормальным законом распределения?

ЛИТЕРАТУРА

1. Айвазян С. А., Мхитарян В. С. Прикладная статистика и основы эконометрики. — М.: ЮНИТИ, 1998. — 650 с.
2. Анализ данных в R [Электронный ресурс] // Stepic.org. — Режим доступа: <https://stepic.org/course/Анализ-данных-в-R-129>.
3. Зорин А. В., Федоткин М. А. Введение в прикладной статистический анализ в пакете R: учебно-методическое пособие. — Нижний Новгород: ННГУ, 2010. — 50 с.
4. Мастицкий С. Э., Шитиков В. К. Статистический анализ и визуализация данных с помощью R. — М.: ДМК Пресс, 2015. — 496 с.
5. Семенычев В. К., Коробецкая А. А., Кожухова В. Н. Предложения эконометрического инструментария моделирования и прогнозирования эволюционных процессов: монография. — Самара: САГМУ, 2015. — 384 с.
6. Эконометрика [Электронный ресурс] // Coursera. — Режим доступа: <https://www.coursera.org/course/econometrics>.
7. Code School — Try R [Электронный ресурс]. — Режим доступа: <http://tryr.codeschool.com/>.
8. Coghlan A. A Little Book of R for Time Series [Электронный ресурс]. — 2015. — Режим доступа: <https://media.readthedocs.org/pdf/a-little-book-of-r-for-time-series/latest/a-little-book-of-r-for-time-series.pdf>.
9. ggplot2 Help [Электронный ресурс]. — Режим доступа: <http://docs.ggplot2.org/current/index.html>.
10. Package 'GA' [Электронный ресурс] // The Comprehensive R Archive Network. — Режим доступа: <https://cran.r-project.org/web/packages/GA/GA.pdf>.
11. Peng R. D. R Programming for Data Science [Электронный ресурс]. — Режим доступа: <https://leanpub.com/rprogramming/>.
12. quantmod: Quantitative Financial Modelling & Trading Framework for R [Электронный ресурс]. — Режим доступа: <http://www.quantmod.com/>.

ВАРИАНТЫ ЗАДАНИЙ

Лабораторная работа 1

№ варианта	1.	2.	3.	4.	5.	6.	7.	8.	9.	10.
Набор данных	CO2		ChickWeight		Orange		airquality		faithful	
Имя переменной (вектора)	conc	uptake	weight	Time	age	circumference	Wind	Temp	eruptions	waiting

Лабораторная работа 2

№ варианта	1.	2.	3.	4.	5.
Набор данных	airquality	state.x77	Cars93*	Cars93*	Cars93*
у	Ozone	Life Exp***	Price	Min.Price	Max.Price
Факторы и их значения для прогноза	Solar.R = 350 Wind = 8,3 Temp = 80	Illiteracy = 1,0 Murder = 12 Income = 4000	Horsepower = 200 RPM = 5200 Passengers = 4	Horsepower = 210 RPM = 5500 Passengers = 4	Horsepower = 220 RPM = 6000 Passengers = 6
№ варианта	6.	7.	8.	9.	10.
Набор данных	stackloss	longley	longley	LifeCycleSavings	Anscombe**
у	stack.loss	GNP	Employed	sr	education
Факторы и их значения для прогноза	Air.Flow = 55 Water.Temp = 20 Acid.Conc = 89	Unemployed = 221 Armed.Forces = 180 Population = 125	GNP.deflator = 102 Armed.Forces = 170 Population = 110	pop15 = 35,5 pop75 = 1,5 dpi = 2500 ddpi = 2,15	income = 3200 young = 347,8 urban = 425

* необходимо подключить библиотеку MASS

** необходимо подключить библиотеку car

*** поскольку в названии переменной нельзя использовать символ пробела, данную переменную необходимо переименовать командой `colnames`, например, в `Life_Exp`, либо обращаться к ней по номеру столбца `[, 4]`

```
d <- state.x77
```

```
colnames(d)[4] <- "Life_Exp"
```

```
#или
```

```
d$Life_Exp <- d[,4]
```

Лабораторная работа 3

№ варианта	1.	2.	3.	4.	5.	6.	7.	8.	9.	10.
Компания	Yandex	Twitter	Facebook	ВКонтакте	Amazon	EBay	Microsoft	Сбербанк	Вымпелком	МТС
Тикер	YNDX	TWTR	FB	VK	AMZN	EBAY	MSFT	SBNC	VIP	MBT
Глубина прогноза	10	15	20	25	30	25	20	15	10	20

Лабораторная работа 4

№ варианта	1.	2.	3.	4.	5.	6.	7.	8.	9.	10.
Среднее x	7	30	100	20	50	10	25	40	60	33
Ст. отклон. x	2	3	15	4	6	1	4	7	10	8
Объем выборки	50	100	150	70	50	30	60	90	110	80
Формула y	$y = a + b(x - c)^2 + \varepsilon$			$y = a + \frac{b}{x - c} + \varepsilon$		$y = a + e^{bx+c} + \varepsilon$		$y = \frac{a}{1 + e^{-b(x-c)}} + \varepsilon$		
a	10	100	-5	10	150	20	1000	50	80	20
b	-2	5	-0.1	3	-10	0.8	-0.25	0.25	0.3	0.22
c	5	32	90	10	30	0.5	5	35	52	30
Ст. отклон. помехи	4	28	14	0.05	0.1	3000	0.25	3	6	1.5

ДЛЯ ЗАМЕТОК

Учебное издание

Кожухова Варвара Николаевна

канд. экон. наук, старший преподаватель кафедры «Высшая математика» Димитровградского инженерно-технологического института – филиала ФГАОУ ВПО «Национальный исследовательский ядерный университет «МИФИ»

Коробецкая Анастасия Александровна

канд. экон. наук, старший преподаватель кафедры
«Общенаучные дисциплины и право» МБОУ ВО «САГМУ».

Семенычев Валерий Константинович

д-р техн. наук, д-р экон. наук, профессор, ректор МБОУ ВО «САГМУ»

Свободная программная среда R

Практикум

для студентов, обучающихся по направлению
38.04.01 «Экономика» (уровень подготовки магистратуры)

Подписано в печать 20.01.2016. Бумага офсетная. Печать оперативная.
Объем 3 п.л. Формат 60х90^{1/16}. Тираж 300 экз.

Муниципальное бюджетное образовательное учреждение
высшего образования
«Самарская академия государственного и муниципального управления»
443084, г. Самара, ул. Стара-Загора, 96
<http://sagmu.ru>

Отпечатано с готового оригинал-макета в типографии ООО «Прайм»
443029, г. Самара, ул. Михаила Сорокина, д. 15
<http://prime163.ru>; e-mail: prime.163@mail.ru