Содержание

Введение 13

Кто и зачем собирает данные 13
Почему R? 14
Как устроена эта книга 15
Обратная связь 15

ЧАСТЬ I. Программирование на R **16**

Глава 1. Знакомство с R 17

Установка 17
Работа в среде RGui 19
Справка 24

Глава 2. Скаляры, векторы и матрицы 26

Арифметические операции и присваивание 26
Имена 27
Простые типы данных 28
Числа 28
Символьный тип 30
Логический тип 32
Векторы 33
Векторизация и логическая индексация 38
Матрицы и массивы 41
Резюме 43

Глава 3. Списки и таблицы 44

Списки 44
Таблицы 47
Функции, применяемые к составным данным 52
apply 52
lapply 53
sapply 54
do.call 55
Резюме 55

Глава 4. Управление процессом вычислений 56

Циклы 56
Цикл со счётчиком 56
Цикл с предусловием 59
Условные операторы 60
Резюме 61

Глава 5. Базовая графика 62

Функции низкого и высокого уровней 62
Глобальные и локальные параметры графиков 67
Легенда 69
Комбинации графиков 69
Графики функций 71
Экспорт в файлы 72
Резюме и ссылки 72

Глава 6. Функции 74

Создание функций 74
Локальные и глобальные переменные. Области видимости . . . 76
Диагностические сообщения 78
Функции в качестве аргументов 78
Функциональное программирование 80
Резюме 81

Глава 7. Факторы и даты 82

Категориальные данные 82
Дата и время 85
Резюме 88

Глава 8. Пакеты 89

Установка и загрузка 89
Выбор пакета 91
Справка и её разновидности 91
Как самому создать пакет R? 93
Пакет magrittr: конвейер операций 94

Глава 9. Ввод и вывод данных. Работа с файлами 96

Рабочий каталог пользователя 96
Запись данных в стандартное устройство вывода 96
Запись в текстовые файлы 97
Таблицы 97
Строки 99
Матрицы 99
Чтение из текстовых файлов 99
Элементы данных: scan 99
Строки: readLines 101
Таблицы 102
Работа с данными в бинарном формате 103
Управление файлами и каталогами 104

Взаимодействие с базами данных . . . 105
DBI + RSQLite . . . 105
sqldf . . . 105
Резюме . . . 106

Ссылки к Части I . . . 107

ЧАСТЬ II. Сбор данных . . . 108

Глава 10. Открытые данные . . . 109

Что это такое? . . . 109
Данные Всемирного банка . . . 110
Где взять данные? . . . 115
Резюме . . . 116

Глава 11. Протокол HTTP . . . 117

Основные понятия . . . 117
Запрос . . . 118
Ответ . . . 119
Коды состояния . . . 120
Передача параметров . . . 121
HTTP в R . . . 122
Пакет httr . . . 122
Пакет RCurl . . . 124
Кириллица и кодирование URL . . . 125
Пример: геокодирование с помощью Google Maps Geocoding API 126
Пример: доступ к API портала открытых данных РФ . . . 128
Ссылки . . . 131

Глава 12. Импорт данных . . . 132

Чтение файлов . . . 132
Скачивание . . . 133
Excel . . . 133
JSON . . . 134
Пример: какой из JSON-пакетов самый популярный? . . . 135
Google Spreadsheets . . . 139
Архивы . . . 139
Завершающий штрих: проверка типа данных . . . 141
Ссылки . . . 141

Глава 13. Веб-скрапинг . . . 142

Используйте структуру данных . . . 142
Элементы HTML и CSS . . . 145

div и span 145
Классы и идентификаторы 146
Путь к элементу 148
XPath 148
CSS 151
Как найти путь к элементу при помощи браузера 152
Проверка и упрощение пути. Консоль разработчика 155
Резюме 157
Лирическое отступление: построение графов 158
Ссылки 160
Поиск в Интернет 160
HTML и CSS: 161
XPath 161

Глава 14. Пакет rvest 162

Пакеты для веб-скрапинга 162
Получение и обработка HTML-документа 163
Поиск элемента 165
Разбор элемента 167
Пример: получаем ссылку и скачиваем файл 168
Таблицы 169
Пример: извлечение таблицы из Википедии 169
Пример: разбор страницы сериала «Светлячок» 170
Пример: извлечение данных об инвестиционных фондах 172
Работа с формами. Сессии 174
Пример: аутентификация на форуме 176
Функции навигации 177
Работа с кодировками 178
Заключительные замечания и ссылки 178

Глава 15. RSelenium: управляем браузером 180

Пример: перевод с помощью Yandex.Translate 182
Пример: динамически генерируемая ссылка на файл 183
Selenium и браузеры 186
Резюме и ссылки 186

Глава 16. PhantomJS и обработка динамических веб-страниц . 188

Динамические страницы: описание проблемы 188
Установка 189
Запуск 189
Пример: рендеринг веб-страницы 191
Сохранение веб-страницы в файл 191

Резюме и ссылки 194

Глава 17. Facebook 195

Протокол авторизации OAuth 2.0 195
Получение маркера доступа пользователя API Graph 196
Доступ к данным с помощью rvest и jsonlite 199
Пакет Rfacebook и создание приложения 201

Глава 18. Сбор информации с помощью API ВКонтакте 207

Создание приложения 207
Регистрация приложения 207
Получение кода доступа 209
Получение данных 210
Реализация в R 211
Построение графа связей 213
Получение другой информации из сети 215
Поиск пользователя 216
Ограничения 217

Глава 19. Использование Twitter API 218

Получение доступа к Twitter API 218
Подключение к Twitter из R 218
Поиск и сохранение его результатов в базе данных 220
Фильтрация результатов поиска 221
Построение облака слов 221
Данные для анализа 223
Лексический корпус и терм-документная матрица 223
Ключевые слова и их частоты 223
Облако слов 224
Ограничения Search API 225
Streaming API 225
Ссылки 226

Глава 20. Регулярные выражения 227

Символы и метасимволы 227
Квантификаторы 229
Положение образца внутри строки 230
Операторы 231
«Жадность» и «лень» квантификаторов 232
Классы символов 234
Заключительные замечания 235
Ссылки 236

Глава 21. Создание карт на основе собранных данных 237
Интерактивные карты в leaflet 237
Переходим к созданию карты 241
Извлечение адресов и названий магазинов 242
Геокодирование 245
Отображение на карте 246
Работа с шейп-файлами 247
Ссылки 249
Ссылки к Части II 251
Приложение А. Среда разработки RStudio 252
Создание скрипта 253
Автодополнение имён объектов 254
Выполнение 254
Рабочее пространство 255
История команд 256
Сохранение файлов 258
Кодировки файлов 258
Управление файлами в рабочем каталоге 259
Управление пакетами 259
Поиск и замена 260
Автоматическое создание функций 261
Комментирование 262
Переход к определению функции 262
Ссылки 263
Приложение Б. Языки поисковых запросов Google и Яндекс . . 264
Почему важно уметь пользоваться ЯПЗ 265
Предотвращение перегрузок сервиса 265
Приложение В. Введение в HTML и CSS 266
Веб-страница 266
Гиперссылки 268
Шрифт 269
Цвет 270
Стиль 270
Выравнивание 272
Рисунки 272
Списки 273
Маркированные 273
Нумерованные 273

Вложенные 274
Таблицы 274
Ссылки 275
Приложение Г. Регулярные выражения 276
Предметный указатель 278

Введение

Всё, что регистрирует человек и созданные им машины может считаться *данными*. Фиксируя новое и переводя архивы в цифровую форму, мы с каждым днём производим всё больше данных. Часть из них собрана в специальных хранилищах. Например, UN Comtrade содержит официальную статистику по международной торговле. Использовать такие данные довольно легко, достаточно лишь получить доступ к их хранилищу.

Но гораздо чаще случается так, что данные разбросаны по всему Internet на многочисленных страницах онлайновых магазинов, заметках в социальных сетях, логах серверов и т. п. Прежде чем начать работать с такими данными, их необходимо собрать и сохранить в пригодном для анализа виде. Решению этих вопросов и посвящена данная книга.

Кто и зачем собирает данные

Круг специалистов, работа которых так или иначе связана со сбором данных, весьма обширен. Судите сами.

Сбор данных является предварительным этапом в *data mining* – «интеллектуальном анализе данных» или же «добыче данных». Data mining можно трактовать весьма широко – как анализ данных вообще, и более узко, как обнаружение скрытых закономерностей в (больших объемах) данных.

Весьма близко к data mining находится *машинное обучение* (*machine learning*). Обе дисциплины пользуются одними и теми же методами, так что чёткой границы между ними провести нельзя. Несколько упрощая можно сказать, что data mining больше интересуется получением закономерностей, а машинное обучение – их использованием. Пользуясь собранными данными машинное обучение решает задачи классификации (например, фильтрации спама или выработки рекомендаций на основе мнений пользователей со сходными запросами), прогнозирования, кластеризации (например, сегментации рынка), выявления аномальных наблюдений и т. п.

Сбор информации о конкурентах, например, мониторинг их ценовых предложений является одной из задач *бизнес-аналитики*. Сходные задачи, но несколько в другом контексте решает *open source intelligence* – разведка по открытым источникам данных, которая отвечает за поиск, сбор и анализ информации из общедоступных источников. Ещё одна близкая область деятельности – *business intelligence*. Несмотря на наличие «intelligence» (*англ.* ‘разведка’) в названии, она предполагает не столько слежение за конкурентами, сколько за бизнес-процессами в собственной организации.

В отличие от предыдущих дисциплин, где результатами анализа данных пользовались лишь отдельные организации, новое направление в журнали-

стике – *журналистика данных* (data-driven journalism) проводит своих исследования в хранилищах данных для информирования публики.

Одним из крупнейших поставщиков общедоступной информации являются социальные сети. В них люди размещают свои анкетные данные, делятся новостями, личными фотографиями, вкусами (лайкая что-нибудь или вступая в какую-либо группу), кругом своих знакомств. Причём всё это делается по доброй воле, подчас не задумываясь о возможных последствиях. Поэтому данные из социальных сетей, добываемые с помощью *social media mining* активно используются для проведения социологических и маркетинговых исследований.

Собранные сведения, имеющие географическую привязку, могут пригодиться в геоинформатике. Если раньше основными источниками данных для *геоинформационных систем* (ГИС) являлись наземная съёмка и результаты дистанционного зондирования Земли, то теперь к ним присоединилась информация, поставляемая многочисленными «людьми-датчиками», отправляющими со своих смартфонов фотографии в Instagram, оставляющих заметки в Facebook, твиты в Twitter и т.п.

К специалистам указанных выше профессий нужно добавить ещё специалистов будущих – студентов, добывающих информацию для своих курсовых и дипломных работ. Всем им, а также просто любопытствующим, эта книга может оказаться полезной.

Почему R?

Заниматься сбором данных можно и на чистом C. В конце концов, большинство библиотек, использующихся при сборе данных, написано именно на этом языке. В качестве примера назовём библиотеки libcurl и libxml2. Но...

Если от языка программирования вам хочется большего «дружелюбия», то можно использовать Python. В нём существует множество пакетов, предназначенных как для сбора, так и для анализа данных. Поэтому, рассматривая ту или иную задачу, мы будем ещё не раз упоминать о Python – как об альтернативном инструменте для её решения.

Но использовать мы будем R. Если Python – это всё-таки язык общего назначения, снабжённый нужными библиотеками, благодаря чему он способен трансформироваться в инструмент для анализа данных, то пакет R в буквальном смысле слова создан статистиками и для статистиков. Поэтому задачи сбора данных в этом языке реализуются наиболее прямолинейно, что позволяет быстрее добиваться результатов, не отвлекаясь на тонкости программирования.

Но и после того как данные собраны, пакет R сопровождает пользователя на протяжении всего цикла их анализа – от предварительной обработки

данных до получения окончательных результатов и представления их на графиках – вплоть до оформления текстов статей полиграфического качества. Кстати, эти строки также набраны в R.

Как устроена эта книга

Основной материал книги разделён на две части. В первой части (к ней относятся главы 1–9) дано краткое введение в R – описание среды разработки, языка и основных пакетов-расширений.

Вторая часть (главы 10–21) посвящена непосредственно сбору данных: работе с открытыми данными (глава 10), извлечению данных из веб-страниц (главы 13–16) и из социальных сетей (главы 17–19). Главы 11, 12 и 20 рассматривают необходимые технические вопросы: протокол HTTP, функции импорта данных различных форматов и регулярные выражения. Завершается рассказ в главе 21 – созданием карт на основе собранных данных.

В конце каждой части приведены ссылки на литературу и веб-ресурсы.

Кроме этого, в приложениях содержится: описание среды разработки RStudio (Приложение А), команды поисковых сервисов Google и Яндекс (Приложение Б), введение в язык разметки HTML (Приложение В) и сводка регулярных выражений (Приложение Г).

Обратная связь

Ваши вопросы и конструктивную критику по содержанию книги присылайте по электронному адресу: *dkhramov@mail.ru*.

Новости, связанные со сбором данных, файлы примеров, список замеченных ошибок, а также ответы на вопросы читателей можно найти на сайте книги: *http://dkhramov.dp.ua/Comp.DataGathering*

Часть I

ПРОГРАММИРОВАНИЕ НА R

Глава 1

Знакомство с R

В этой главе мы установим R и научимся пользоваться средой разработки RGui. По пути решим простую, но весьма распространённую задачу – построим график функции.

Установка

Рассмотрим способ установки R, который подойдёт для любого типа операционных систем, поддерживаемых пакетом: Linux, Mac или Windows. В случае Windows этот путь – единственный, для Linux и Mac проще воспользоваться менеджером пакетов соответствующей системы.

Чтобы установить R, зайдём на его официальный сайт и выберем интересующую версию программы. Сам R, документация к нему и дополнительные пакеты распространяются через сеть ftp- и web-серверов, называемую CRAN (Comprehensive R Archive Network). Поэтому следующим шагом будет выбор одного из более чем шести десятков зеркал CRAN. После этого, указав тип операционной системы, скачиваем дистрибутив R.

Запустим инсталляционный файл и будем следовать указаниям программы-установщика. Единственный момент, который потребует вашего внимания — выбор разрядности операционной системы (рис. 1.1).

После установки, запустим программу. В Linux и Mac, по умолчанию, R работает в консоли. Кроме того, вместе с пакетом поставляется графический интерфейс. В Linux он основан на связке Tcl/Tk и запускается командой: `R --gui=Tk`. В Mac встроенный графический интерфейс для R называется R.app.

Версия R для Windows поставляется с графической оболочкой RGui (рис. A.5), которую мы и рассмотрим в дальнейшем.

Заметим, что работа со всеми указанными выше графическими оболочками выглядит примерно одинаково.

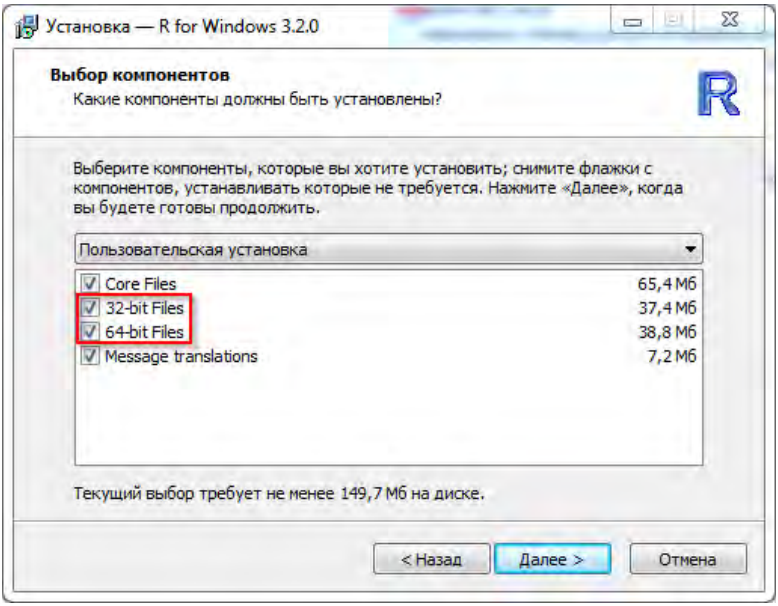

Рис. 1.1 ❖ Выбор компонентов в ходе установки R под Windows

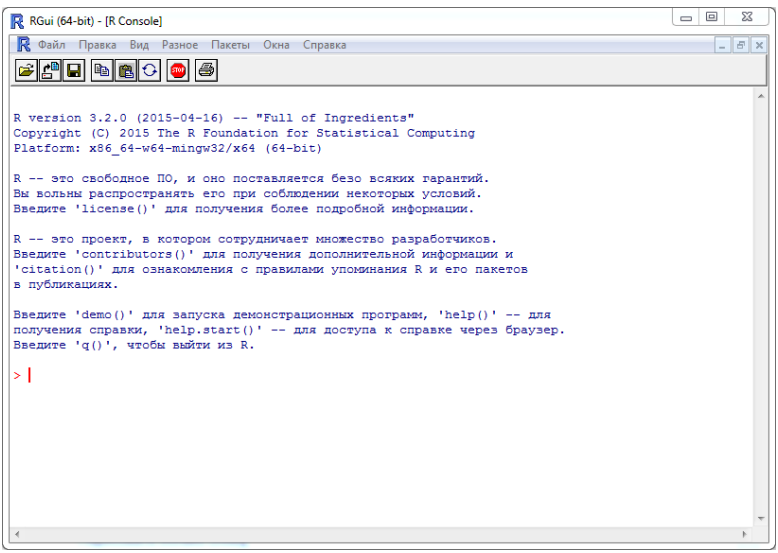

Рис. 1.2 ❖ Консоль R в графической оболочке RGui (Windows)

Работа в среде RGui

RGui – это стандартная графическая оболочка R под Windows, являющаяся простейшей средой разработки. Она быстро загружается и достаточно удобна в использовании. В RGui есть три вида окон:

- консоль;
- редактор скриптов;
- окно графического устройства.

Команды R вводятся в консоли (рис. А.5) после приглашения пользователя (значка `'>'`) и отправляются на выполнение нажатием *Enter*.

Управление консолью:

- дополнение команды – *Tab*;
- перемещение по истории команд – клавиши со стрелками;
- прекращение выполнения команды – *Esc*;
- переключение в другое окно – *Ctrl+Tab*;
- очистка консоли – *Ctrl+L*;

Для создания собственных программ (скриптов)[1] удобнее использовать не консоль, а редактор (рис. 1.3).

Открыть его можно в меню Файл/Новый скрипт. Первое окно открывается с помощью меню, последующие – так же или комбинацией клавиш *Ctrl+N*.

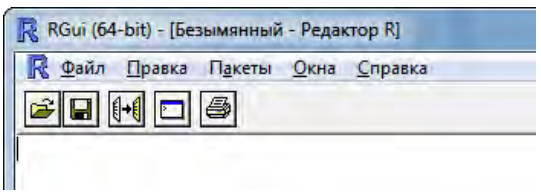

Рис. 1.3 ❖ Интерфейс редактора кода в RGui

Обратите внимание на то, как изменилась панель инструментов по сравнению с консолью (рис. А.5).

В качестве примера построим график синусоиды:

```
x <- seq(-pi,pi,.1)
y <- sin(x)
plot(x,y)
```

В первой строке формируется одномерный массив (вектор) x-координат, значения которых изменяются от $-\pi$ до π с шагом 0.1. Этот массив сохраняется в переменной `x`. Комбинация символов `<-` обозначает операцию присваивания.

[1] Мы используем термины «скрипт» и «программа» как синонимы.

Во второй строке создаётся вектор y, состоящий из синусов элементов вектора x.

Наконец, в третьей строке функция `plot` строит требуемый график.

В редакторе можно набирать команды и выполнять их как по одной, так и целыми блоками, с помощью комбинации *Ctrl+R* (на панели инструментов также есть соответствующая кнопка). Например, можно выделить весь скрипт – *Ctrl+A* и отправить его на выполнение *Ctrl+R* (рис. 1.4).

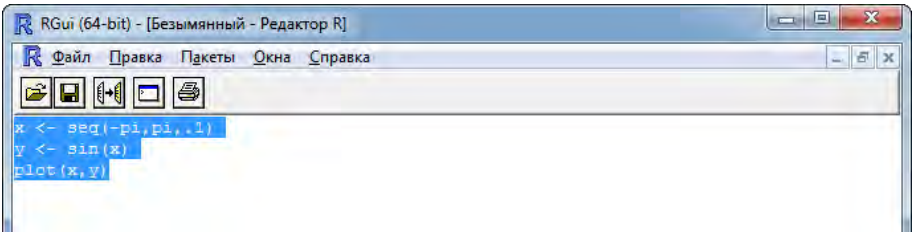

Рис. 1.4 ❖ Выделенную часть кода можно отправить на выполнение комбинацией клавиш *Ctrl+R*

В результате получим графическое окно с построенным в нём графиком синусоиды (рис. 1.5). По терминологии R окна, в которых строятся графики, и файлы, в которых эти графики сохраняются, вместе называются *графическими устройствами*.

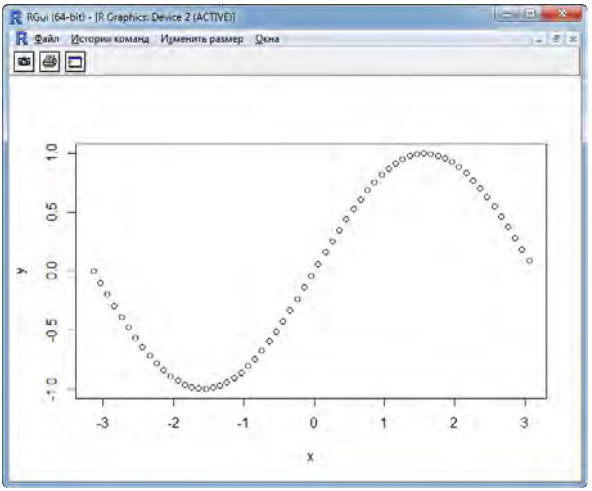

Рис. 1.5 ❖ Графическое устройство в RGui

Вернуться к консоли можно нажатием кнопки на панели инструментов (рис. 1.6) или при помощи *Ctrl+Tab*.

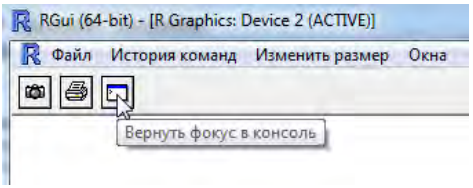

Рис. 1.6 ❖ Кнопка Вернуть фокус в консоль на панели инструментов графического устройства

Управление окнами при помощи меню Окна показано на рис. 1.7.

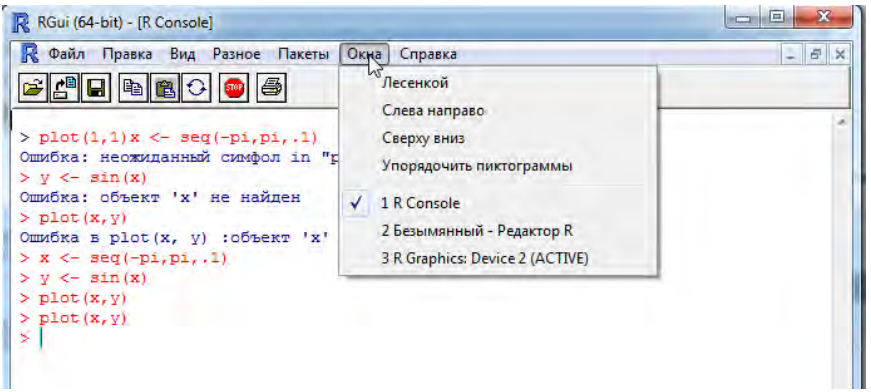

Рис. 1.7 ❖ Управление окнами с помощью меню Окна

Сохранить скрипт можно при помощи команд Сохранить или Сохранить как... меню Файл редактора или соответствующей кнопки на его панели управления.

Заметим, что скрипты R, объединённые общей темой, удобно держать в отдельном каталоге.

Выйти из R можно, вызвав в консоли `q()` или воспользовавшись меню Файл/Выйти.

При выходе из RGui, среда предложит сохранить рабочее пространство (рис. 1.8).

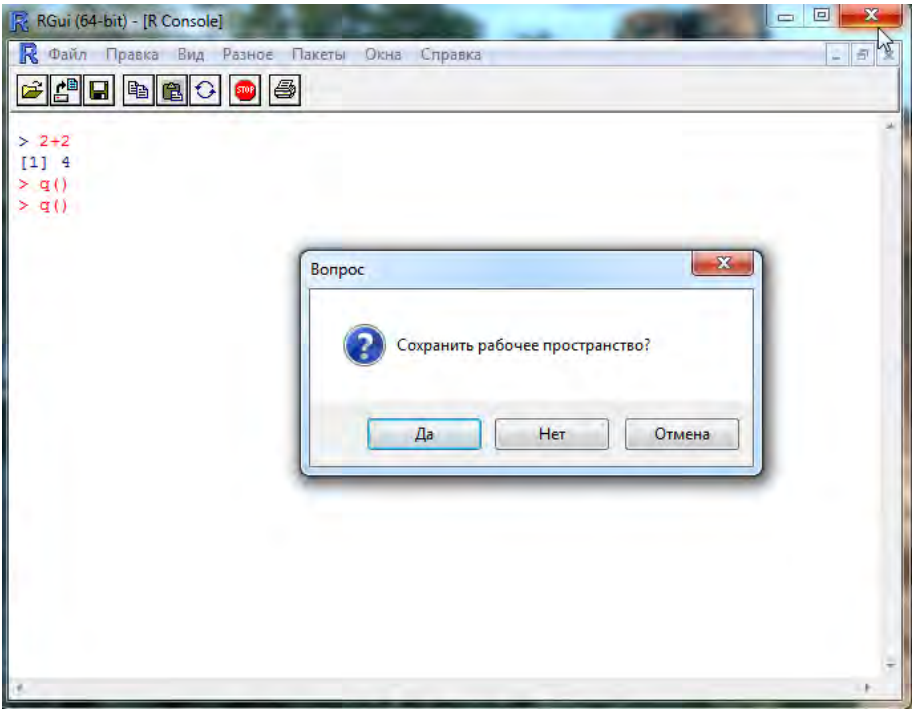

Рис. 1.8 ❖ Сохранение рабочего пространства по выходу из RGui

Рабочее пространство (workspace) – это область оперативной памяти, в которой хранятся все созданные пользователем объекты (векторы, матрицы, таблицы, списки, функции и т. п.). Сохранение рабочего пространства в файле и его последующая загрузка в новом сеансе работы с R (Файл/Загрузить рабочее пространство...) позволяют продолжить работу с того места, где она была прервана. При этом сохраняются значения всех переменных, вычисленные на прошлом сеансе работы.

Просмотреть список объектов в рабочем пространстве можно при помощи функции `ls`. Вызовем её в консоли

```
> ls()
```

и получим в результате

```
## [1] "x" "y"
```

Действительно, в памяти сейчас хранятся векторы координат синусоиды.

Два символа решётки (##) указывают на то, что далее идёт результат выполнения программы (в консоли они не отображаются, см. рис. 1.9). Смысл единицы ([1]) станет ясен в следующей главе.

Другие команды управления рабочим пространством:

```
# Сохранить рабочее пространство в файл .RData в текущем рабочем каталоге
save.image()
# Сохранить заданные объекты в файле
save(object_list, file="myfile.RData")
# Загрузить образ рабочего пространства
load("myfile.RData")
# Очистить рабочее пространство
rm(list=ls())
```

Символ # открывает строку комментария.

Сохранить или загрузить образ рабочего пространства можно при помощи меню **Файл**. Следует иметь в виду, что это меню, как и любое другое в RGui, просто запускает на выполнение соответствующие функции R.

Например, команда меню **Файл/Сохранить рабочее пространство...** вызывает функцию `save.image` и результат этого вызова можно увидеть в консоли (рис. 1.9).

```
> x <- seq(-pi, pi, .1)
> y <- sin(x)
> plot(x,y)
> ls()
[1] "x" "y"
> save.image("C:\\Users\\Admin\\Documents\\myimage")
>
```

Рис. 1.9 ❖ Команда меню Файл/Сохранить рабочее пространство... представляет собой вызов функции `save.image`

Клавиши со стрелками вверх и вниз позволяют перемещаться по списку выполненных ранее команд – *истории команд*. Историю команд также можно сохранить в файл на диске.

```
# Вывод истории команд
history()                 # выводит список 25 последних выполненных команд
history(max.show=Inf)     # выводит все выполненные в сессии команды
# Сохранить историю команд
savehistory(file="myfile") # по умолчанию, ".Rhistory"
# Загрузить историю команд
loadhistory(file="myfile") # по умолчанию, ".Rhistory"
```

Помимо RGui, для R существуют другие, более продвинутые среды разработки, например, R Commander или RStudio. Также, как и сам R они являются кроссплатформенными и распространяются свободно. Заметим, что хотя возможностей у этих сред гораздо больше, они дополняют возможности RGui, но не перечёркивают их. Так что навыки работы с RGui пригодятся и при работе в более совершенной среде.

Справка

Верхний пункт меню Справка (в консоли – это Консоль, в редакторе скриптов, соответственно, Редактор) даёт короткую подсказку по работе с соответствующим окном. Справка для редактора показана на рис. 1.10.

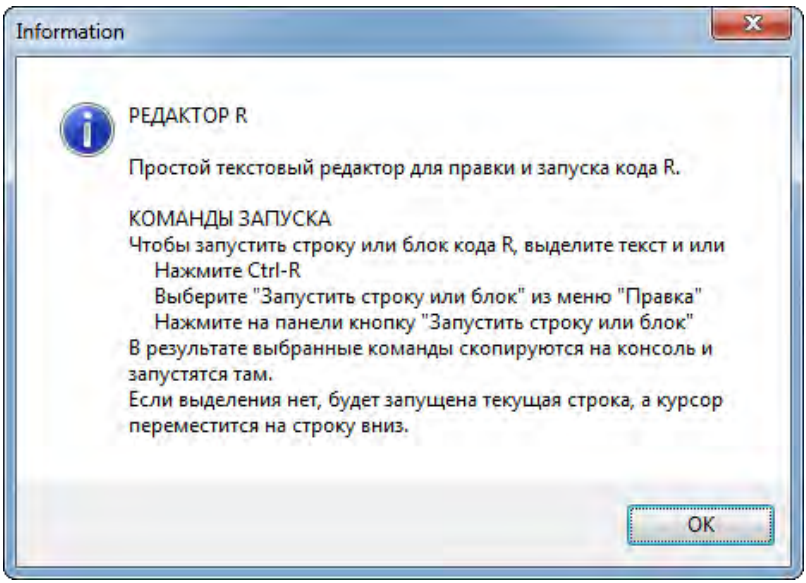

Рис. 1.10 ❖ Справка по работе в редакторе скриптов RGui

Справку по функции с именем `fun` можно получить, набрав в консоли `?fun`. Например, для получения справки по функции создания последовательностей `seq`, наберём:

```
?seq
# или
help(seq)
```

В ответ на выполнение этой команды откроется браузер со справочной информацией (рис. 1.11).

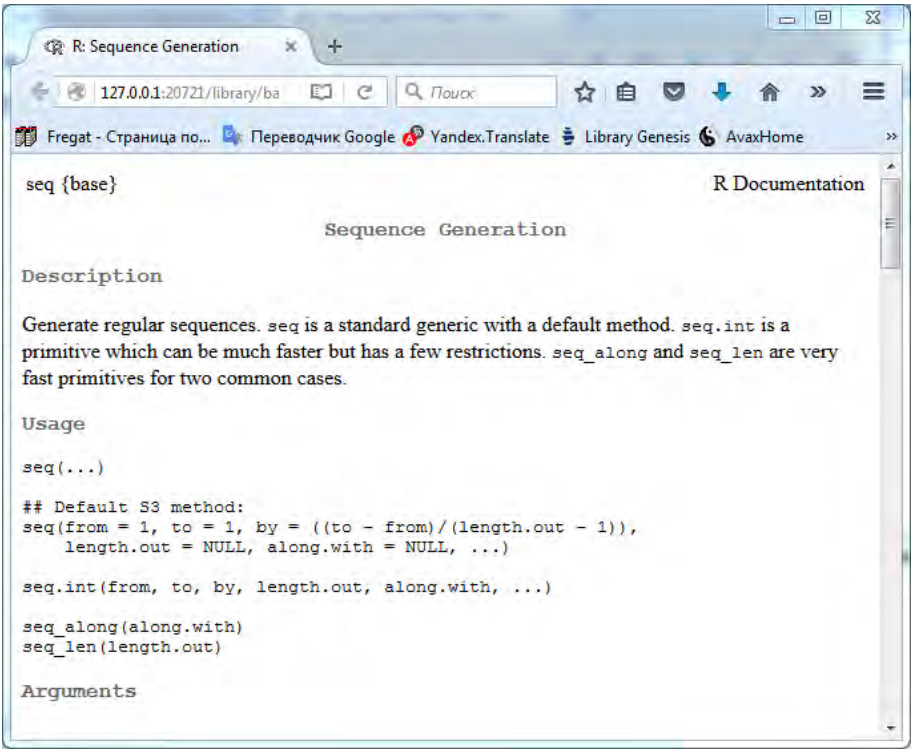

Рис. 1.11 ❖ Справка по функции seq