Владимир Брюков

КАК ПРЕДСКАЗАТЬ КУРС ДОЛЛАРА

Поиск доходной стратегии с языком R

12+

Владимир Георгиевич Брюков

Как предсказать курс доллара. Поиск доходной стратегии с языком R

http://www.litres.ru/pages/biblio_book/?art=37401488
SelfPub; 2018

Аннотация

Валютный трейдер с первых же минут работы на рынке решает две сложнейшие задачи. Во-первых, перед ним стоит задача заработать крупные суммы, оправдывающие серьезные риски, связанные с торговлей на валютном рынке. А во-вторых, он ни на секунду не должен забывать о потенциально возможных больших потерях вложенных средств, и, следовательно, должен торговать, соблюдая все правила риск-менеджмента. Именно этим двум важнейшим вопросам и посвящена наша книга. При этом особый акцент в ней будет сделан на подробном изложении алгоритма поиска наиболее оптимальной – с точки зрения соотношения доходности и риска – торговой стратегии.

Содержание

Предисловие

Валютный трейдер с первых же минут работы на рынке решает две сложнейшие задачи. Во-первых, перед ним стоит задача заработать крупные суммы, оправдывающие серьезные риски, связанные с торговлей на валютном рынке. А во-вторых, он ни на секунду не должен забывать о потенциально возможных больших потерях вложенных средств, и, следовательно, должен торговать, соблюдая все правила риск-менеджмента, позволяющие минимизировать этот риск. При этом каждый трейдер должен понимать, что из-за значительного воздействия случайного фактора на динамику валютного курса риск потери средств в ходе торгов невозможно полностью устранить, а можно лишь свести к определенному разумному минимуму.

Именно этим двум важнейшим вопросам и посвящена наша книга «Как предсказать курс доллара. Поиск доходной стратегии с языком R». При этом особый акцент в ней будет сделан на подробном изложении алгоритма поиска наиболее оптимальной – с точки зрения соотношения доходности и риска – торговой стратегии. Хочу напомнить моим читателям, что в предыдущих двух своих книгах – «Как предсказать курс доллара. Расчеты в Excel для снижения риска проигрыша» и «Как предсказать курс доллара. Эффек-

тивные методы прогнозирования с использованием Excel и EViews» (первое и второе дополненное издания) – акцент в большей степени сделан на обучении трейдера расчетам, позволяющим свести валютные риски к разумному минимуму. Все желающие могут приобрести эти книги в интернет-магазинах компании ЛИТРЕС и ее партнеров.

Ну а в этой книге мы будем учиться делать прогнозы и искать наиболее доходную стратегию с помощью такого мощного инструмента, как язык программирования R. Тем, кто еще не занимался программированием, либо не знаком с языком R, советую внимательно проштудировать вводную часть этой книги, в которой рассказывается об установке R и основных азах работы с этим языком. Все остальные могут сразу перейти к последующим материалам, посвященным основной теме нашей книги.

Об авторе

Брюков Владимир Георгиевич, независимый финансовый аналитик, с 2003 года занимается банковской журналистикой.

С 2005 года особое место в его публикациях занимают статистические методы анализа валютных и финансовых рынков. Теме валютного прогнозирования, в первую очередь, прогнозу по курсу доллара США, посвящены многие его статьи, опубликованные в журналах «Валютный спекулянт», «Инвестиционный банкинг» и в ряде других изданий. В этих публикациях обобщаются результаты проведенного автором исследования валютного рынка, предлагаются оптимальные методы прогнозирования по курсам валют с учетом последних достижения современной статистической науки. В 2011 году издательство КНОРУС и Центр Исследований Платежных Систем и Расчетов опубликовали первою его книгу, посвященную валютному прогнозированию – см. Брюков В. Г. «Как предсказать курс доллара. Эффективные методы прогнозирования с использованием Excel и EViews».

С 2009 по 2015 год В. Г. Брюков на портале Банкир.Ру ежемесячно публиковал прогнозы на будущий месяц по курсам пятнадцати ведущих мировых валют. Насколько точными при этом были прогнозы, наши читатели могут убедиться сами, посетив на этом сайте рубрику «Валютный рынок». Сотрудничество с этим известным порталом, а также большой интерес, проявленный читателями к книге «Как

предсказать курс доллара. Эффективные методы прогнозирования с использованием Excel и EViews», стали для автора важным стимулом, способствовавшим написанию новой книге по валютному прогнозированию.

В 2017 году у автора вышла в электронном виде еще одна книга по этой теме: «Как предсказать курс доллара. Расчеты в Excel для снижения риска проигрыша», тогда же вышло и второе дополненное издание «Как предсказать курс доллара. Эффективные методы прогнозирования с использованием Excel и EViews», опубликованное также в электронном виде. Все желающие могут приобрести эти книги (а также ряд других, не связанных с проблемами валютного прогнозирования) в интернет-магазинах компании ЛИТРЕС и ее партнеров. Так, что книга «Как предсказать курс доллара. Поиск доходной стратегии с языком R» – уже третья из этой серии, причем, полное представление о ряде важных аспектах валютного прогнозирования можно получить, только ознакомившись с содержанием всех этих трех произведений.

Глава 1. Основы работы с языком R

Созданный в 1993 году язык программирования R сегодня получил во всем мире очень широкое распространение, в том числе и для анализа рисков, связанных со сделками на различных финансовых рынках, включая и торговлю на валютном рынке. Это объясняется, главным образом, тремя основными причинами.

Во-первых, потому что R – это чрезвычайно эффективный язык программирования, с помощью которого можно выполнять практически все способы статистического и графического анализа данных.

Во-вторых, в отличие от разного рода платных статистических программ, язык R невероятно гибок, что позволяет создавать пакеты прикладных программ (приложений) для самых различных сфер деятельности, в том числе и для прогнозирования валютного и прочих финансовых рынков. Причем, количество этих приложений бурно растет. Так, на конец августа 2018 года в глобальном репозитарии (хранилище) CRAN (The Comprehensive R Archive Network – Полный сетевой архив R) находилось 12940 доступных для загрузки пользователями прикладных программ.

В-третьих, у языка R свободный код, то есть распростра-

няется он бесплатно. И это дает ему весьма серьезное конкурентное преимущество перед зачастую очень дорогим платным программным обеспечением. В 2010 году язык R за особые заслуги перед сообществом программистов вошёл в список победителей конкурса журнала InfoWorld в номинации на лучшее открытое программное обеспечение для разработки приложений.

Прежде чем начать пользоваться языком R для расчетов нам нужно сначала загрузить его последнюю версию. Все необходимые установочные файлы и всю необходимую информацию можно найти на одном из дублирующих (зеркальных) сайтов CRAN. С этой целью нужно пройтись по ссылке – https://cran.r-project.org/, где пользователи операционных систем Linux, Mac OS X и Windows должны выбрать свои опции: Download R for Linux, Download R for (Mac) OS X и Download R for Windows – см. рис. 1.

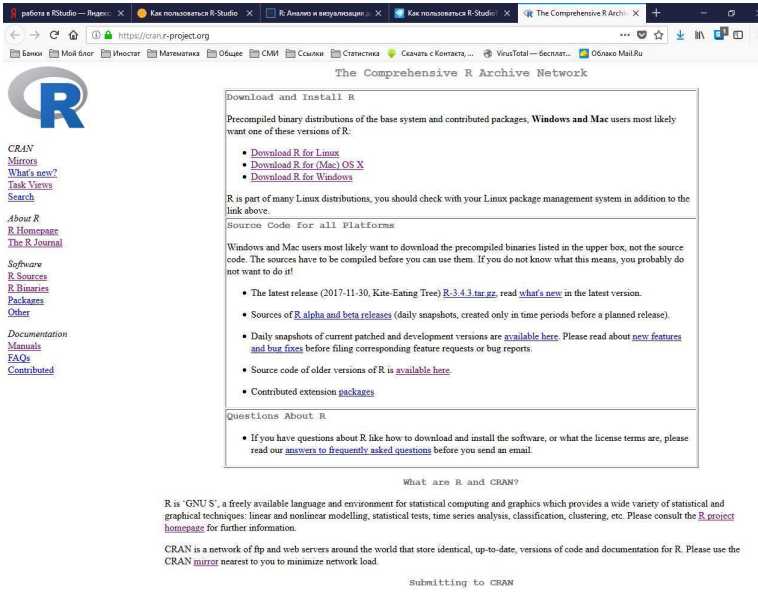

Рис. 1

Заметим, что зачастую у пользователей Linux установочный файл с R уже входит в базовый дистрибутив, а потому его не нужно скачивать. Если же его нет, то пользователи, как Linux, так и Mac OS X при установке R должны сначала выбрать уже упомянутые выше опции, соответственно, Download R for Linux или Download R for (Mac) OS X, а затем внимательно следовать подробным инструкциям, которые даются на портале https://cran.r-project.org/. Причем, установленный алгоритм действий нужно строго соблюдать,

поскольку некомпетентные действия могут причинить вред операционной системе пользователя.

В свою очередь, пользователи Windows при установке R должны воспользоваться следующими опциями: сначала пройтись по ссылке Download R for Windows, затем щелкнуть по ссылке base и, наконец, по ссылке – Download R 3.5.1 for Windows (62 megabytes, 32/64 bit), то есть загрузить последнюю версию языка (в тот момент, когда пишутся эти строки, последней версией была R 3.5.1).

Всю необходимую информацию по установке R можно найти по следующей ссылке – Installation and other instructions (установка и другие инструкции), которая, к большому сожалению для русскоязычных пользователей, дается на английском языке. Впрочем, даже неопытный пользователь Windows легко справится с инсталляцией R, если он при этом будет использовать уже установленные по умолчанию опции и щелкать опцию Next до завершения этого процесса.

При этом 32-битовую версию R, как правило, рекомендуют устанавливать в том случае, если пользователь работает на 32-битовой операционной системе Windows, а обе версии – на 64-битовой. Заметим, что 32-битовая версия иногда работает быстрее 64-битовой, но последняя необходима пользователю в том случае, если ему требуется больший объем оперативной памяти, чем 32-битовая версия R в состоянии справиться. Если у Вас относительно современный компью-

тер с оперативной памятью 4 Гб или более – смело ставьте 64-битную версию. Если оперативной памяти у Вашего компа менее 4 Гб и ее Вы не планируете расширять – ставьте 32-бита.

Для того чтобы узнать, 32 или 64-битовая версия Windows стоит на Вашем компьютере, нужно: во-первых, набрать на клавиатуре сочетание клавиш Windows+E, после чего откроется меню параметров, в том числе и такой ее параметр как «Этот компьютер» (в последних версиях Windows), либо «Мой компьютер» (в более ранних версиях Windows); во-вторых, нужно щелкнуть правой кнопкой мышки по параметру «Этот компьютер» или «Мой компьютер», после чего появится окно (рис. 1), в котором о типе системе сказано следующее: «64-разрядная операционная система процессора x64».

Язык R имеет встроенную стандартную среду разработки – RGui (Graphic User Interface – графический пользовательский интерфейс), используемую для ввода и редактирования программного кода. Эта среда разработки имеет вид командной строки в окне, называемом консолью. Командная строка работает по принципу «вопрос-ответ».

Помимо встроенной среды разработки RGui для языка R создано еще ряд других, зачастую, более совершенных ее аналогов, одни из которых находятся в свободном доступе, а другие платные. В этой книге мы будем использовать для наших расчетов интегрированную бесплатную среду разработ-

ки – RStudio, имеющую более удобный интерфейс, что существенно упрощает работу с R. Облегчает работу пользователя также и наличие в RStudio цветовой подсветки, автоматического завершения кода и удобной навигации по скрипту. Серьезным плюсом RStudio является также и ее совместимость с основными операционными системами – Windows, Linux, Mac OS X.

RStudio – это относительно более новая среда разработки для R, появившаяся в декабре 2010 года. RStudio доступна в двух версиях: 1. RStudio Desktop, в которой программа выполняется на локальном компьютере как обычное приложение; и 2. RStudio Server, в которой предоставляется доступ через браузер к RStudio, установленной на удаленном Linux-сервере. RStudio можно загрузить по следующей ссылке – https://www.rstudio.com/products/rstudio/.

После того как мы успешно завершили установку последней версии R и инсталляцию среды разработки RStudio, можно приступать к работе. Для этого нужно два раза щелкнуть левой кнопкой мышки по иконке RStudio, которая находится в меню «Пуск» в перечне остальных программ, установленных на Вашем компьютере. Для большего удобства иконку RStudio лучше прикрепить к панели задач Вашего компьютера, что ускорит начало работы.

После запуска RStudio появляется окно (рис. 2), разделенное на четыре части: 1. Слева в верхней части этого окна находится редактор кода, в котором вводится и редактирует-

ся создаваемый программный код; 2. Слева в нижней части окна пользователь увидит консоль, выполняющую команды, введенные редактором кода; 3. Справа в верхней части окна размещена история команд; 4. Справа в нижней части окна находится своего рода справочная, с помощью которой можно получить помощь Help, доступ к списку загруженных программных пакетов, графиков и файлов, находящихся в текущем рабочем каталоге. В ходе работы в основном графическом окне могут появиться и другие окна – редактор скриптов, окна с графическим результатом выполнения команд (графики).

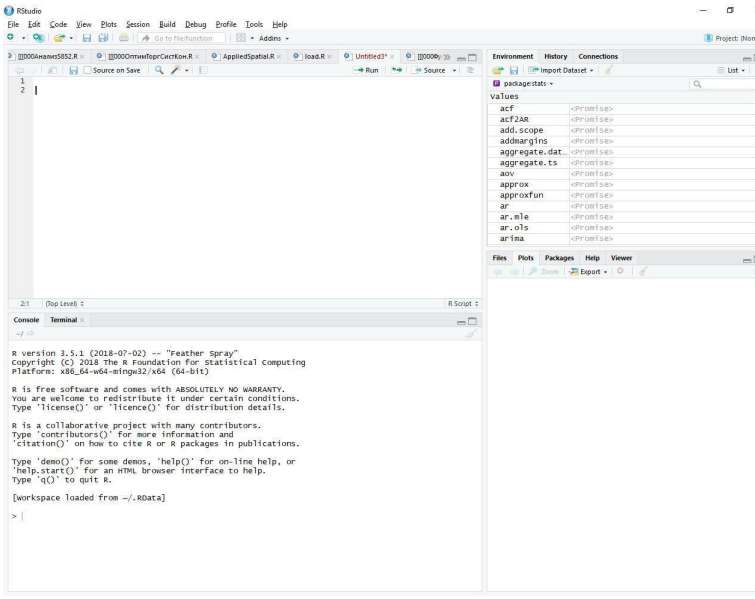

Рис. 2

После того как RStudio загрузится, в нижней части справа появившегося его окна, то есть в консоли, появится символ «>», означающий, что среда разработки в данный момент ничем не занята и ожидает ввода наших новых команд. Символом присвоения в языке R является символ «<-», но иногда используется также и обычный символ присвоения «=». Команды, размещаемые на одной строке, отделяются точкой с запятой.

Далее все пояснения к программному коду будем давать по ходу изложения, предваряя их символом #, который не является частью распознаваемого компьютером кода. Всё, что находится после этого знака в рамках одной строки, игнорируется программой. При этом все пояснения к программному коду будем давать по ходу изложения, предваряя их символом решетки #, которая не является частью распознаваемого компьютером кода.

В R можно создавать имена для различных объектов (переменных) как на латинице, так и на кириллице. Кроме того, R различает регистр, то есть одно и тоже слово, начинающееся со строчной или прописной буквы, для него не одинаково. В этой книге для более понятного изложения кода автор решил давать названия переменным на кириллице. Функции в R англоязычные, но для лучшего их запоминания автор советует начинающим программистам попытаться перевести их с английского на русский. Правда, это не всегда может получится, поскольку нередко такие функции представляют собой англоязычные сокращения, либо новые словоформы, которые не могут быть переведены с помощью словаря общеупотребительной лексики. Тем не менее тому, кто решил всерьез заняться программированием на R, нужно будет освоить язык Шекспира, по крайней мере в той степени, чтобы читать и понимать справочные материалы по этому языку.

Каждый объект в R относится к тому или иному типу дан-

ных, использование которых для программирования имеет свою специфику. R работает с самыми разными структурами данных, включая векторы, матрицы, массивы данных, таблицы и списки, которые различаются типами данных, способом создания, сложностью устройства, а также способом обозначать и извлекать из них отдельные элементы.

Векторы – это одномерные массивы данных, которые могут содержать числовые, текстовые или логические значения. Для создания векторов применяются:

1. Функция объединения c, с помощью которой объединяются элементы, перечисленные в скобках через запятую:

```
> вектор.A<– c(1,5,7)
# c – функция объединения от англ. слова concatenation – объединение, слияние
# объединяет аргументы в один вектор определенного типа
# если бы R понимал по-русски, то эту команду можно было бы ввести так:
# вектор.A= объединить(1,5,7)
> вектор.A
[1] 1 5 7
> # задать R вопрос является ли вектор.A вектором можно так:
> is.vector(вектор.A)
# если бы R понимал по-русски, то эту команду можно было бы ввести так:
```

вектор.ли(вектор.А)

[1] TRUE

ответ TRUE по-русски означает ИСТИНА, т.е. да, вектор.А является вектором

2. Функция последовательности seq, в которой первая цифра в скобках обозначает начальное значение вектора, вторая – конечное значение вектора, а третья цифра – величину интервала создаваемой последовательности:

> вектор.Б<-seq(0,4,2)

если бы R понимал по-русски, то эту команду можно было бы ввести так:

вектор.Б<-последовательность (0,4,2)

> вектор.Б

[1] 0 2 4

3. Для объединения используется также функция, обозначаемая знаком двоеточия, после которого следует первая цифра, присваиваемая начальному значению вектора, затем – вторая, которая присваивается конечному значение вектора. При этом вектор с указанной последовательностью цифр перечисляется с интервалом =1:

> вектор.В<-0.5:6

> вектор.В

[1] 0.5 1.5 2.5 3.5 4.5 5.5

> # перевести количественные данные вектора В в текстовые можно так:

> вектор.В<-as.character(вектор.В)

по-русски: вектор.В<-как.текст(вектор.В)

проверить являются ли данные вектора В текстовыми можно так:

> is.character(вектор.В)

по-русски: текст.ли(вектор.В)

[1] TRUE

перевести текстовые данные вектора В в количественные данные можно так:

> вектор.В<-as.numeric(вектор.В)

проверить, являются ли данные вектора В количественными можно так:

> is.numeric(вектор.В)

[1] TRUE

> вектор.В

[1] 0.5 1.5 2.5 3.5 4.5 5.5

Элементы в рамках одного вектора могут быть только одного типа, но различные векторы могут содержать данные различных типов. При этом все элементы вектора с текстом при объединении заключаются в кавычки:

> Текстовый.вектор.Г<– c('элемент1','элемент2', 'элемент3')

> Текстовый.вектор.Г

[1] "элемент1" "элемент2" "элемент3"

> class(вектор.Г)

[1] "character"

> Логический.вектор.Д<– c(TRUE, FALSE, TRUE,

FALSE, TRUE)

если бы R понимал по-русски, то эту команду можно было бы ввести так:

Логический.вектор.Д<– c(ИСТИНА, ЛОЖЬ, ИСТИНА, ЛОЖЬ, ИСТИНА)

> Логический.вектор.Д

[1] TRUE FALSE TRUE FALSE TRUE

> # определите тип данных вектора Д можно так:

> class(вектор.Д)

[1] "logical"

Подробнее о векторах с логическими данными можно узнать, введя команду help("&"). Эту команду легко запомнить, если знать, что слово help в переводе на русский означает помощь.

Класс или тип объекта в R можно определить с помощью функции class() так:

> class(вектор.А)

числовой вектор

[1] "numeric"

> class(Текстовый.вектор.Г)

текстовый вектор

[1] "character"

> class(Логический.вектор.Д)

логический вектор

[1] "logical"

Отдельный элемент вектора можно извлечь, обозначив

его положение (номер строки) в квадратных скобках:

> Текстовый.вектор.Г[2]

[1] "элемент2"

Отдельный элемент из вектора можно убрать, поставив в квадратных скобках перед его положением (номером строки) знак минус:

> вектор.Б

[1] 0 2 4

> вектор.Б[-1]

[1] 2 4

Отдельный элемент можно вставить в вектор, указав в квадратных скобках положение (номер строки) элемента, куда его нужно вставить и приравняв его к определенному значению:

> вектор.Б[1]<-0

> вектор.Б

[1] 0 2 4

В R основным типом данных являются данные количественного ("numeric") и текстового типа ("character"). При этом данные количественного типа ("numeric") представляются собой действительные числа, которые могут быть представлены в виде дробей. В то время как данные логического типа (“logical”), факторы (“factor”) и целые числа (“integer”) считаются дополнительными. Причем, дополнительный тип данных(“integer”) хранит количественные данные в формате целых чисел (“integer”). Преобразование из "numeric" в

“integer” можно выполнить следующим образом:

> вектор.B<-0.5:6

> вектор.B

числа в векторе представлены в виде чисел с десятичными дробями

[1] 0.5 1.5 2.5 3.5 4.5 5.5

> class(вектор.B)

[1] "numeric"

> вектор.B<-as.integer(вектор.B)

по-русски эту команду можно перевести так:

вектор.B<-как.целое(вектор.B)

вектор.B из "numeric" преобразуют в "integer"

> class(вектор.B)

[1] "integer"

> вектор.B

[1] 0 1 2 3 4 5

числа в векторе представлены в виде целых чисел без дробной части

Матрицы представляют собой двумерный массив данных, в котором каждый ее элемент имеет одинаковый тип данных. Матрицу можно создать при помощи функции matrix:

Например, матрицу из последовательности цифр 1,2 …15 из трех строк (nrow=3) можно создать следующим образом:

> Матрица1 <– matrix(1:15, nrow=3)

эта команда создает матрицу из вектора 1:15=1,2 … 15

количество строк в этой команде задается аргументом

nrow

если объект x (в этой команде он =1:15) не обладает достаточной длиной

его элементы при создании матрицы будут использованы повторно ("recycling")

если бы R понимал по-русски, то эту команду можно было бы ввести так:

Матрица1 <– матрица(1:15, число строк=3)

1:15 означает последовательность 1, 2 … 15

> Матрица1

[,1] [,2] [,3] [,4] [,5]

[1,] 1 4 7 10 13

[2,] 2 5 8 11 14

[3,] 3 6 9 12 15

Эту же матрицу, но из трех столбцов (ncol=3) можно создать следующим образом:

> Матрица2 <– matrix(1:15, ncol=3)

количество столбцов задается аргументом ncol

> Матрица2

[,1] [,2] [,3]

[1,] 1 6 11

[2,] 2 7 12

[3,] 3 8 13

[4,] 4 9 14

[5,] 5 10 15

Отдельный элемент матрицы можно извлечь, обозначив

его положение (номер строки и номер столбца) в квадратных скобках:

> Матрица2[3,2]

[1] 8

Отдельный элемент матрицы можно удалить (указав со знаком минус номер удаляемого элемента, предварительно определив его порядковый номер, считая от начала первой строки первой колонки и до конца последней строки последней колонки), но в результате она становится вектором:

> Матрица2[-8]

[1] 1 2 3 4 5 6 7 9 10 11 12 13 14 15

Отдельный элемент можно вставить в матрицу, указав в квадратных скобках положение (номер строки и номер столбца) куда его нужно вставить и приравняв его к определенному значению:

> Матрица2[3,2]<– NaN

> Матрица2

[,1] [,2] [,3]

[1,] 1 6 11

[2,] 2 7 12

[3,] 3 NaN 13

[4,] 4 9 14

[5,] 5 10 15

NaN– по-английски означает Not-a-Number-«не число». NaN получается в результате: деления 0 на 0, деления 0 на бесконечность, деления бесконечности на бесконечность,

умножения 0 на бесконечность, сложения бесконечности с бесконечностью противоположного знака, вычисления квадратного корня отрицательного числа, логарифмирования отрицательного числа, а также в результате всех математических операций с использованием NaN в качестве одного из операндов. В R бесконечность обозначается как Inf. Например, в результате деления Inf на Inf получаем NaN:

> Inf/Inf

[1] NaN

Чтобы NAN в Матрице 2 заменить на нуль нужно ввести такой код:

> Матрица2[is.na(Матрица2)]<-0

по-русски: Матрица2[является. nan (Матрица2)]<-0

> Матрица2

[,1] [,2] [,3]

[1,] 1 6 11

[2,] 2 7 12

[3,] 3 0 13

[4,] 4 9 14

[5,] 5 10 15

Отдельный столбец матрицы можно удалить, указав со знаком минус номер удаляемого столбца:

> Матрица2[, -2]

#Матрица2[, -2 столбец] – перед запятой вместо номера строки оставляют пустое место

[,1] [,2]

[1,] 1 11
[2,] 2 12
[3,] 3 13
[4,] 4 14
[5,] 5 15

Отдельный столбец можно вставить в матрицу, указав в квадратных скобках столбец, куда его нужно вставить, и приравняв его к вектору вставляемых значений:

> Матрица2[, 2]<-16:20
Матрица2[, 2 столбец] <-16:20
> Матрица2
[,1] [,2] [,3]
[1,] 1 16 11
[2,] 2 17 12
[3,] 3 18 13
[4,] 4 19 14
[5,] 5 20 15

Отдельную строку матрицы можно удалить, указав в квадратных скобках со знаком минус номер удаляемой строки:

> Матрица2[-3,]
Матрица2[-3 строка,] – после запятой вместо номера столбца оставляют пустое место
[,1] [,2] [,3]
[1,] 1 16 11
[2,] 2 17 12
[3,] 4 19 14

[4,] 5 20 15

Отдельную строку матрицы можно вставить, указав ее номер в квадратных скобках, и приравняв ее к вектору вставляемых определенных значений

> Матрица2[3,] <-c(3,8,13)

Матрица2[3 строка,] <-c(3,8,13)

> Матрица2

[,1] [,2] [,3]

[1,] 1 6 11

[2,] 2 7 12

[3,] 3 8 13

[4,] 4 9 14

[5,] 5 10 15

R также работает и с массивами данных (array), которые сходны с матрицами, но могут иметь данные с более чем двумя измерениями. Очевидно, что массивы данных – это просто расширенные матрицы. Как и в матрицах, все элементы массива должны иметь одинаковый тип данных. Массивы данных создаются при помощи функции array. Например, массив из последовательности чисел 1,2 …30, состоящий из двух матриц с тремя строками и пятью столбцами можно создать следующим образом:

> Мой.Массив<– array(1:30, dim=c(3,5,2))

аргумент dim указывает на размер массива данных

dim =c (3,5,2) создает из вектора 1:30 массив данных из

3 строк, 5 столбцов и 2 матриц.

если бы R понимал по-русски, то эту команду можно было бы ввести так:

Мой.Массив<– множество(1:30, размер=объединить(3,5,2))

```
> Мой.Массив
, , 1

[,1] [,2] [,3] [,4] [,5]
[1,] 1 4 7 10 13
[2,] 2 5 8 11 14
[3,] 3 6 9 12 15

, , 2

[,1] [,2] [,3] [,4] [,5]
[1,] 16 19 22 25 28
[2,] 17 20 23 26 29
[3,] 18 21 24 27 30
> dim(Мой.Массив)
[1] 3 5 2
```

Заметим, что в функции array в скобках сначала дается вектор 1:30, из которого создается массив данных, затем следует выражение dim=c(3,5,2), предписывающее с помощью функции объединения создать массив данных, соответ-

ственно, из трех строк, пяти столбцов и двух матриц.

Отдельный элемент массива данных можно извлечь, обозначив его положение (номер строки, номер столбца и номер матрицы) в квадратных скобках. Например, цифру, стоящую в третьей строке и третьем столбце второй матрицы можно извлечь следующим образом:

```
Мой.Массив[3,3,2]
> Мой.Массив[3,3,2]
[1] 24
```

Таблицы данных, которые в отличие от матриц могут состоять из различных типов данных, широко используются в R. Таблицы данных создаются при помощи функции data.frame(). Покажем, как это делается на конкретном примере. Сначала создадим три вектора данных, из которых один будет текстовый, а два других цифровых:

```
> Успеваемость <-c('Отличники', 'Хорошисты' , 'Троечники', 'Двоечники')
> Успеваемость
[1] "Отличники" "Хорошисты" "Троечники" "Двоечники"
> Студенты<-c(2, 5,10,2)
> Студенты
[1] 2 5 10 2
> Студентки <-c(3,7,14,1)
> Студентки
[1] 3 7 14 1
```

Теперь создаем таблицу с помощью функции data.frame,

которую назовем Моя.Таблица:

```
> Моя.Таблица <– data.frame(Успеваемость,Студенты, Студентки)
> Моя.Таблица
Успеваемость Студенты Студентки
1 Отличники 2 3
2 Хорошисты 5 7
3 Троечники 10 14
4 Двоечники 2 1
# узнаем является ли Моя.Таблица таблицей:
> is.data.frame(Моя.Таблица)
# по-русски: таблица.ли(Моя.Таблица)
[1] TRUE
# по-русски ответ: ИСТИНА, то есть этот объект является таблицей
```

Далее проверим структуру данных Моя.Таблица с помощью следующей функции:

```
> str(Моя.Таблица)
# по-русски: структура(Моя.Таблица)
'data.frame':    4 obs. of 3 variables:
$ Успеваемость: Factor w/ 4 levels "Двоечники","Отличники",..: 2 4 3 1
$ Студенты : num 2 5 10 2
$ Студентки : num 3 7 14 1
# по-русски: 'data.frame'– таблица
# 4 obs. of 3 variables – 4 наблюдения из 3 переменных
```

знак $ обозначет переменные, включенные в таблицу

Factor w/ 4 levels – фактор из 4 уровней

num – количественные данные

Отдельный элемент таблицы можно извлечь, обозначив его положение (номер строки и номер столбца) в квадратных скобках:

> Моя.Таблица[3,1]

[1] Троечники

Levels: Двоечники Отличники Троечники Хорошисты

Внизу из текстового элемента Моя.Таблица есть следующая строка: «Levels: Двоечники Отличники Троечники Хорошисты». Levels в переводе на русский язык означает Уровни. Так называемые «Уровни» (Levels) присваиваются факторам. Фактор – это векторный объект, кодирующий категориальные данные (классы), в состав которых входят как номинальные, так и порядковые данные. Номинальные данные – это качественные данные, которые отражают условные коды количественно не измеряемых категорий, которые также не подлежат ранжированию или упорядочиванию. В качестве примера номинальных данных можно привести индексы отделений связи, поскольку они служат только для их идентификации. По отношению к номинальным данным возможны только операции «равенство-неравенство».

В отличие от номинальных порядковые данные могут быть ранжированы как в порядке убывания, так и увеличения какого-либо их качества. Но в отличие от обычных

количественных данных, которые можно выразить в конкретных единицах и к которым можно применить широкий круг алгебраических операций, к порядковым данным можно применить лишь операции «равенство-неравенство», а также «больше-меньше». Порядковые данные используются в том случае, когда порядок ранжирования элементов по какому-то критерию важен, а вот количественные различия между различными рангами этих элементов не поддаются точной оценке.

Например, такие ответы респондентов на вопрос социолога, как: «согласен», «частично согласен», «нет могу сказать, согласен или не согласен», «частично не согласен», «не согласен», – можно ранжировать по степени их согласия или степени их несогласия, в то время как количественную разницу между вариантами этих ответов трудно оценить в каких-то конкретных единицах. Следовательно, эти данные являются порядковыми или ранжируемыми. Впрочем, иногда порядковым данным могут присваиваться какие-то условные порядковые числа, но и в этом случае количественная разница между различными рангами одной и той же последовательности носит весьма условный характер. Например, порядковыми данными являются пятибалльные оценки знаний учащихся, поскольку они не могут быть сгенерированы методом измерения в конкретных единицах, а получены методом достаточно субъективного оценивания.

Созданная нами переменная Успеваемость относится к

числу ранжируемых, но по умолчанию уровни фактора в R присваиваются текстовому вектору в алфавитном порядке. Поскольку переменная Успеваемость является порядковой, то такая градация в этом случае не подходит. Поэтому сначала проверим тип данных переменной Успеваемость, а затем присвоим значение различных уровней фактора в порядке возрастания успеваемости с помощью следующей команды:

```
> class(Успеваемость)
[1] "character"
# тип данных – текстовый
> Успеваемость <– factor(Успеваемость, order=TRUE, levels=c('Двоечники', 'Троечники', 'Хорошисты', 'Отличники'))
# превращает вектор Успеваемость в упорядоченный фактор
# число уровней фактора задается при помощи аргумента levels
> Успеваемость
[1] Отличники Хорошисты Троечники Двоечники
# уровни фактора в порядке их возрастания
Levels: Двоечники < Троечники < Хорошисты < Отличники
> class(Успеваемость)
[1] "ordered" "factor"
# тип данных – упорядоченный фактор
```

Список в R представляет собой упорядоченный набор объектов с различными типами данных. В результате под одним своим именем списки могут включать векторы, матрицы, таблицы и другие списки. Список можно создать при помощи функции list():

```
> Мой.Список <– list(Моя.Таблица, Успеваемость, Матрица1,Матрица2)
# по-русски: Мой.Список <– список(Моя.Таблица, Успеваемость, Матрица1,Матрица2)
> Мой.Список
[[1]]
Успеваемость Студенты Студентки
1 Отличники 2 3
2 Хорошисты 5 7
3 Троечники 10 14
4 Двоечники 2 1

[[2]]
[1] Отличники Хорошисты Троечники Двоечники
Levels: Двоечники < Троечники < Хорошисты < Отличники

[[3]]
[,1] [,2] [,3] [,4] [,5]
[1,] 1 4 7 10 13
[2,] 2 5 8 11 14
```

[3,] 3 6 9 12 15

[[4]]
[,1] [,2] [,3]
[1,] 1 6 11
[2,] 2 7 12
[3,] 3 8 13
[4,] 4 9 14
[5,] 5 10 15

Теперь проверим структуру данных Моя.Таблица с помощью следующей функции:

> str(Мой.Список)
List of 4 # список из объектов
$:'data.frame': 4 obs. of 3 variables:
..$ Успеваемость: Factor w/ 4 levels "Двоечники","Отличники",..: 2 4 3 1
..$ Студенты : num [1:4] 2 5 10 2
..$ Студентки : num [1:4] 3 7 14 1
$: Ord.factor w/ 4 levels "Двоечники"<"Троечники"<..: 4 3 2 1
$: int [1:3, 1:5] 1 2 3 4 5 6 7 8 9 10 …
$: num [1:5, 1:3] 1 2 3 4 5 16 17 8 19 20 …
характеризуется тип данных по каждой переменной $ (объекту) списка

Теперь попробуем поработать в RStudio как с обычным

калькулятором. С этой целью подсчитаем, насколько вырос курс американского доллара к рублю по итогам торгов 17 декабря 2014 года. Обратите внимание, что при работе с языком R дробная часть числа отделяется точкой, а не запятой.

Согласно данным Банка России, официальный курс доллара США, установленный на 18 декабря 2014 г., равнялся 67.7851 руб. Поскольку Центробанк каждый рабочий день по итогам утренних торгов на Московской межбанковской валютной биржи устанавливает официальные курсы валют, которые вступают в силу лишь на следующий день, то, следовательно, официальный курс доллара на 18 декабря 2014 г. по сути является его текущим курсом по итогам торгов, прошедших 17 декабря 2014 г. В свою очередь, по итогам торгов от 16 декабря 2014 г. курс доллара равнялся 61.1512 руб., а потому к моменту их закрытия 17 декабря 2014 г. американская валюта подорожала до 67.7851 руб.

Давайте с помощью R поработаем с этими цифрами. Для того, чтобы выяснить, например, насколько рублей и во сколько раз за один день подорожала американская валюта, нужно, во-первых, от второй цифры отнять первую, а, во-вторых, вторую цифру поделить на первую. Иначе говоря, нам необходимо выполнить два следующих простейших действия: 1). 67.7851 – 61.1512 и 2). 67.7851/61.1512.

С этой целью последовательно щелкнем мышкой в верхней левой части RStudio по опциям File/New File/R Script, а затем введем с клавиатуры вышеуказанные цифры и ал-

гебраические символы в редакторе кода. Затем выделим их мышкой и нажмем на клавиатуре кнопки Ctrl и Enter (Ввод). В результате в консоли (нижней левой части) RStudio появятся не только введенные нами математические выражения, но и ответы – см. рис. 3. Отправить на консоль эти выражения можно также последовательно щелкнув в верху, в левой части RStudio, по опциям Code/ Run Selected Line(s) (Код/Отправить на консоль выбранные строки).

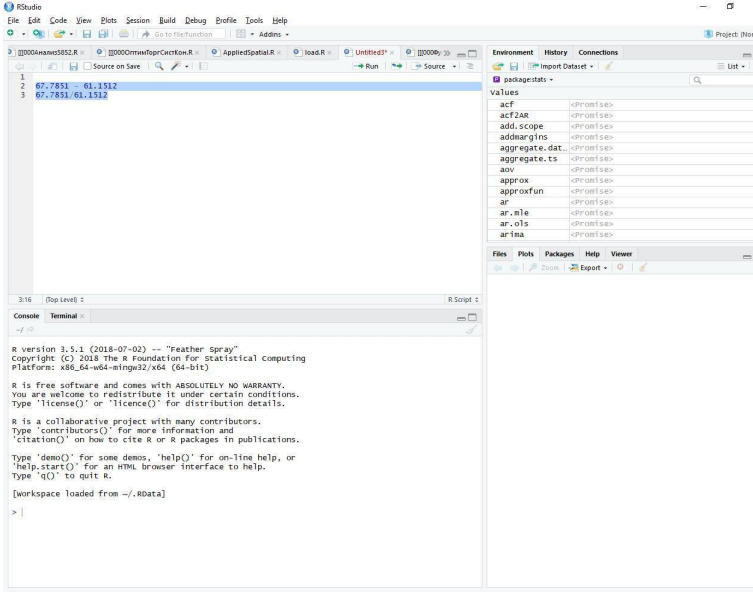

Рис. 3

В результате в консоли появятся следующие два выражения:

> 67.7851 – 61.1512
[1] 6.6339
> 67.7851/61.1512
[1] 1.108484

При этом после символа > в начале абзаца даны введенные нами в редакторе кода выражения, а после [1] даются полученные ответы: [1]6.6339 и [1] 1.108484. Таким образом по итогам утренних торгов, прошедших 17 декабря 2014 г., курс доллара вырос на 6.6339 руб. или в 1.108484 раза. Почему выводимые результаты в R начинаются с [1]? Это объясняется тем, что R по умолчанию рассматривает любые данные как массив данных. В данном случае выводимое число – это состоящий из одного элемента вектор, который и нумеруется соответствующей порядковой цифрой [1]. В том случае, когда вектор состоит из множества элементов, занимающих сразу несколько строк, тогда в начале каждой строки в квадратных скобках выдается порядковый номер (подсчет ведется от начала вектора) первого элемента каждой строки.

В R помимо уже использовавшихся нами для вычитания и деления символов « – » и « / » применяются также символы «+», «*», «^» (или «**»), соответственно, для сложения, умножения и возведения в степень. Например, если умно-

жить число 1.108484 на 100 и отнять 100, то тогда получим: 1.108484*100 -100=10. 8484%. Таким образом с помощью этой операции мы выясним, что по итогам утренних торгов, прошедших 17 декабря 2014 г., курс доллара по сравнению с торгами предыдущего дня вырос на 10. 85%.

R имеет встроенную систему помощи, которая содержит подробные разъяснения, а также ссылки на литературу и примеры для каждой функции из установленных пакетов. Правда, все эти справки даются на английском языке. Например, если пользователь хочет получить справку по функции lm, с помощью которой в R решаются уравнения регрессии, то с этой целью ему надо ввести команду ?lm, либо help(lm). В результате он получит справочный файл по этой функции. Но если пользователю необходимо получить информацию об этой функции, содержащуюся во всех имеющихся справочных файлах, то в этом случае надо ввести команду help.search (lm) или ??lm.

Команда example(lm) дается в том случае, когда пользователь хочет ознакомиться с конкретными примерами по работе с этой функцией, с ее помощью можно также ознакомиться и с примерами по другим функциям, если внутри скобок вместо lm указать их название. Команда RSiteSearch("lm") позволяет получить справочные материалы по функции lm, имеющиеся в онлайн-руководствах и в заархивированных рассылках. Команда apropos("lm", mode="function") даст список всех функций, в которых есть

название lm. Вполне естественно, что если в скобках после apropos вместо lm указать, например, foo, то тогда можно получить аналогичную информацию о foo. Список всех доступных руководств по загруженным пакетам можно получить с помощью команды vignette(). Ну а если запустить команду help.start(), то в правой нижней части RStudio появится обширная справочная литература по R. В первую очередь, из этого списка пользователю, знающему английский язык, можно посоветовать внимательно познакомиться с пособием «An Introduction to R» («Введение в язык R»). Кроме того, тем, кто владеет английским языком, весьма полезно будет также проштудировать еще и книгу Andrie de Vries, Joris Meys «R For Dummies» (Андри де Фриз, Джорис Мейс «R для чайников»), в которой весьма доступно излагаются основные азы работы с языком.

Конец ознакомительного фрагмента.