Владимир Брюков

КАК ПРЕДСКАЗАТЬ КУРС ДОЛЛАРА

Поиск доходной стратегии с языком R

12+

Владимир Брюков

Как предсказать курс доллара. Поиск доходной стратегии с языком R

«ЛитРес: Самиздат»

2018

Брюков В. Г.

Как предсказать курс доллара. Поиск доходной стратегии с языком R / В. Г. Брюков — «ЛитРес: Самиздат», 2018

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. Эффективные 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 распространяется 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.

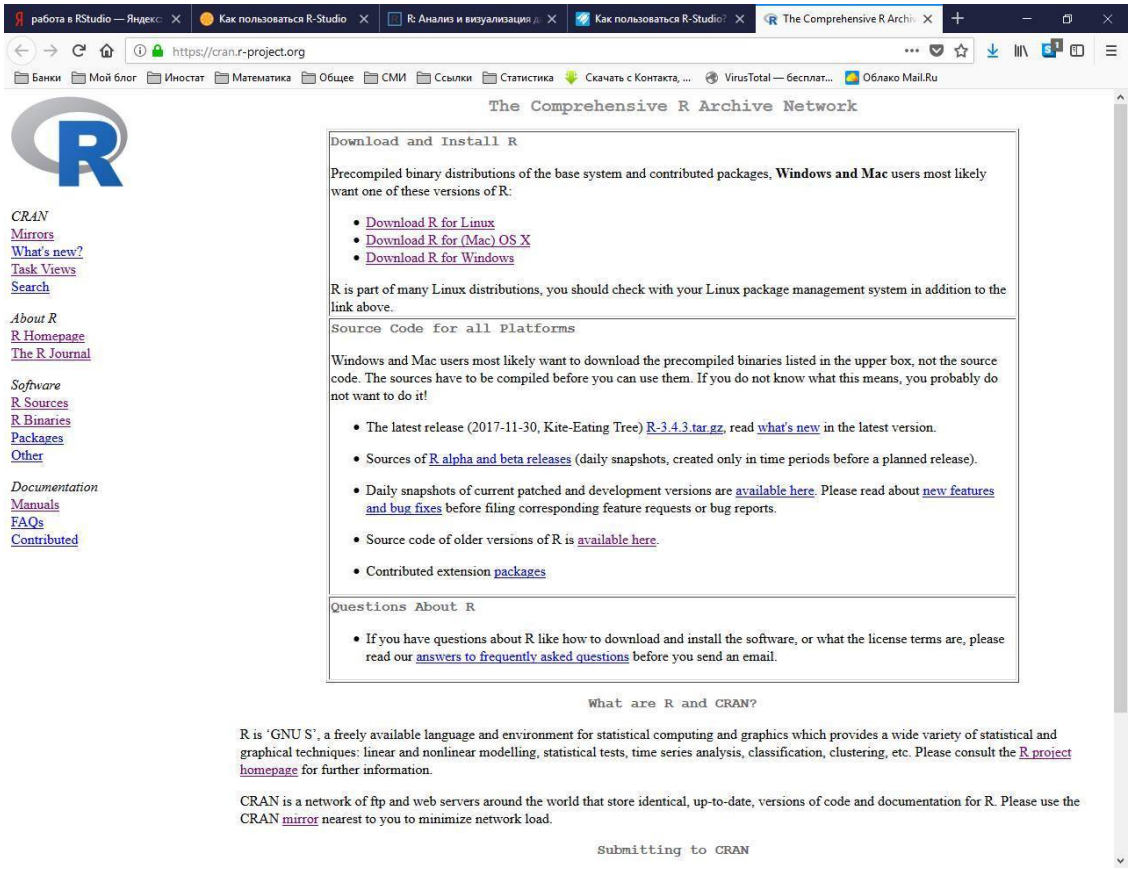


Рис. 1

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 компьютер 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 разработки – 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 редактируется 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).

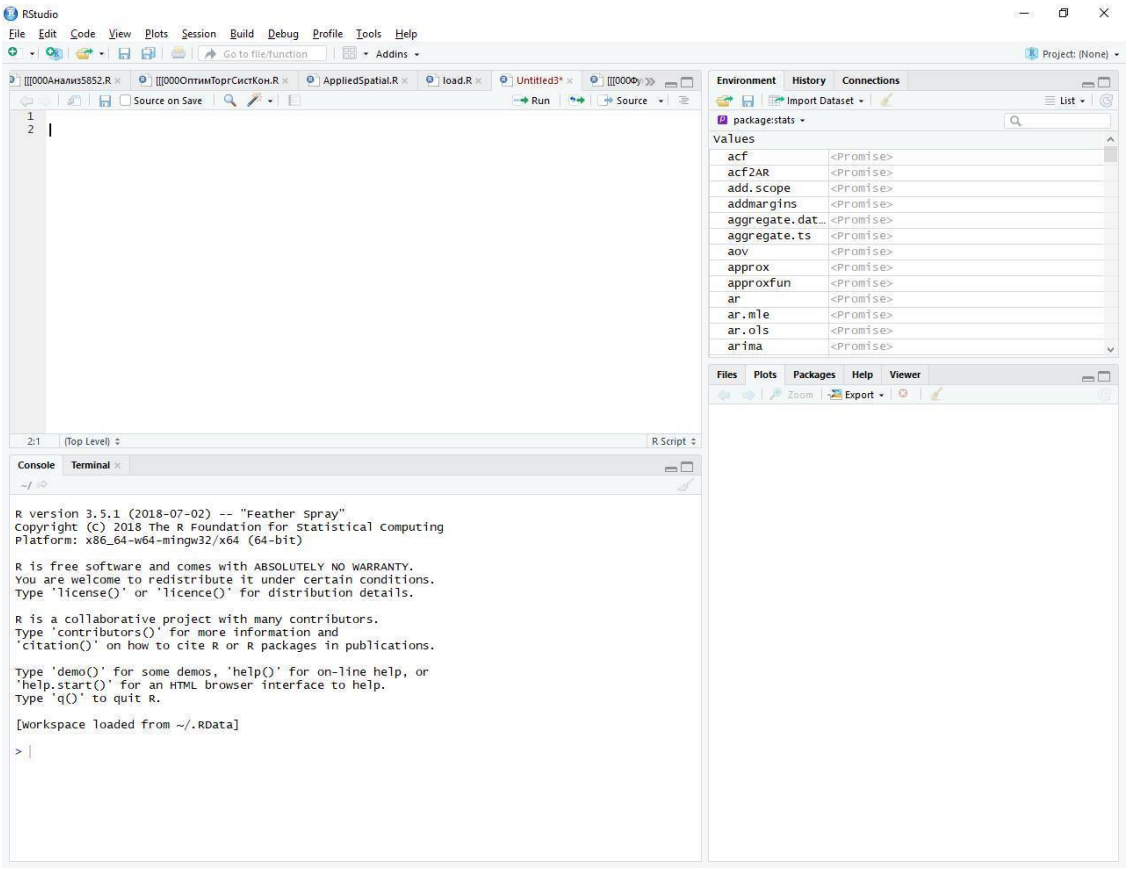


Рис. 2

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 данных, 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.А<– 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.А= объединить(1,5,7)
> вектор.А
[1] 1 5 7
> # задать R вопрос является ли вектор.А вектором можно так:
> is.vector(вектор.А)
# 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, соответственно, 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 алгебраические 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).

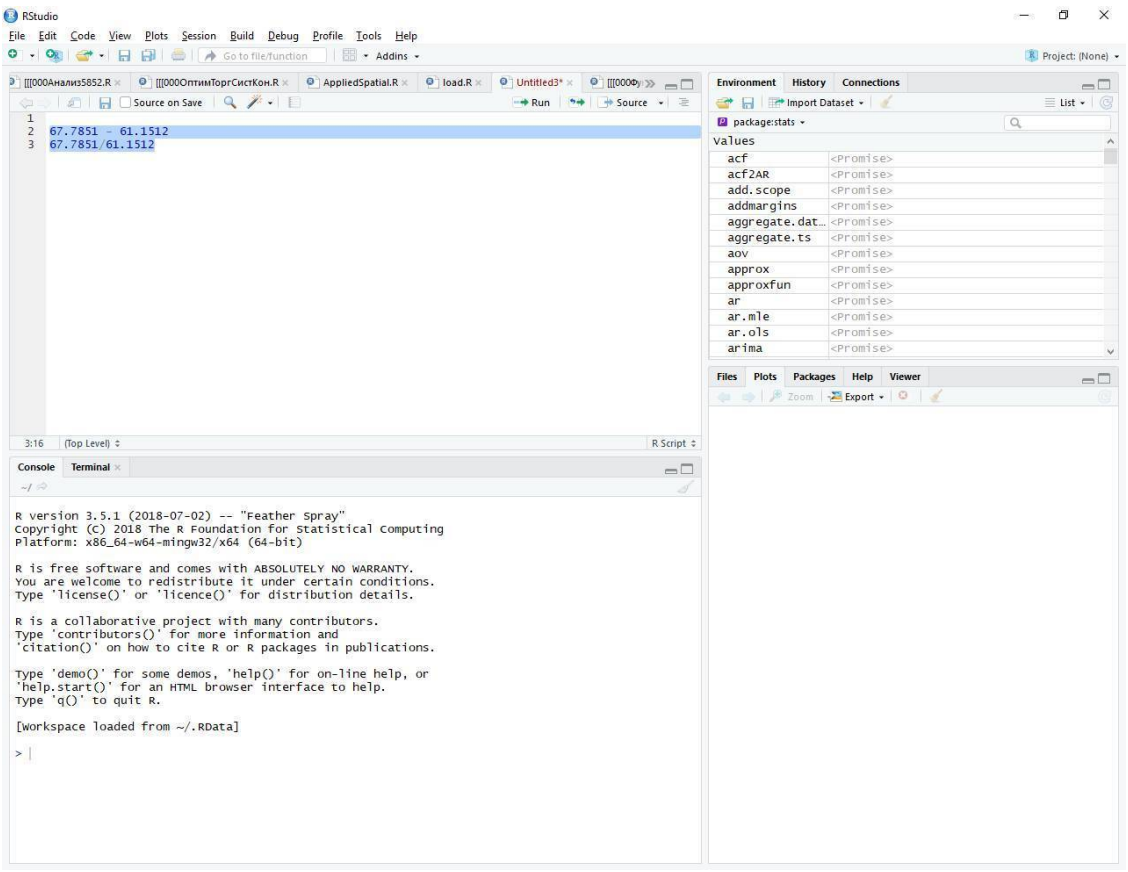


Рис. 3

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 умножить 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.

Хочу также порекомендовать нашим читателям прочитать следующие замечательные книги на русском языке: Роберт И. Кабаков «R в действии. Анализ и визуализация данных в программе R (пер. с англ. Полины А. Волковой)», С. Э. Мастицкий и В. К. Шитиков «Статистический анализ и визуализация данных с помощью R», а также коллективный труд – «Наглядная статистика. Используем R!» А. Б. Шипунова, Е. М. Балдина, П.А. Волковой, А. И. Коробейникова, С. А. Назаровой, С. В. Петрова и В. Г. Суфиянова.

Поскольку наша книга посвящена прогнозированию на валютном рынке, поэтому программированию на языке R мы будем учиться на конкретных примерах. При этом наиболее трудные моменты автор постарается объяснить, как можно проще и доступнее даже для самых неподготовленных читателей. Если же у Вас останутся какие-то вопросы, то ответы на них можно получить в перечисленной выше литературе по R. Но прежде чем перейти к главной теме нашей книги остановлюсь на наиболее распространенных ошибках, которую допускают новички, работающие на языке R.

В том случае, если программа выдает информацию о допущенной ошибке, то вполне возможно, что начинающий программист допустил одну из следующих ошибок:

1. Забыл поставить в написанном им программном коде кавычки там, где они обязательны. Так, в команде install. packages(“zoo”) название устанавливаемого пакета нужно обязательно ставить в кавычках.

2. Написал функцию прописными буквами, в то время как ее нужно было написать строчными. Например, команды ?lm, либо help (lm) будут поняты R, а вот при вводе команд ? Lm, Help (lm) появиться сообщение об ошибке. Аналогичная ошибка появится, если вместо команды по загрузке пакета library(MASS) будет ошибочно введена команда library(mass), в которой название пакета дано строчными буквами.

3. Забыл поставить скобки при обращении к функции: к примеру, нужно набирать help(), а не help даже если аргументы у функции в данном случае отсутствуют.

4. Использовал в команде по установки рабочей директории setwd('C: \Users …’) при указании пути к файлу в операционной системе Windows используется обратный слэш \ , в то время как в R в этом случае нужно поставить: либо два обратных слэша \\, либо один прямой /.

Таким образом эту команду надо вводить либо как setwd('C: \\Users …'), либо как setwd('C: / Users …').

5. Ввел функцию из пакета, который еще не загружен. Например, функция read.zoo() относится к пакету zoo. Если Вы этот пакет, еще не загрузили, а функцию read.zoo() уже ввели, то появится сообщение об ошибке. Правда не все пакеты нужно загружать, поскольку часть пакетов, считающихся базовыми, устанавливаются одновременно с установкой R.

Задание 1. Овладеваем азами работы с языком R

После того, как читатель познакомился с некоторыми азами работы с R, теперь давайте попробуем применить их на практике. С этой целью нужно выполнить следующие задания:

1. Построить вектор А с помощью функции, которую если бы R понимал по-русски, можно было бы ввести так (какой командой в R надо заменить русское слово «объединить»):

вектор.А= объединить(2,6,8)

Какую функцию в R используют, чтобы выполнить команду «объединить»? Определите класс или тип объекта вектор.А в R

2. Построить вектор Б с помощью функции, которую если бы R понимал по-русски, можно было бы ввести так:

вектор.Б<-последовательность (0,-8, -2)

Какую функцию в R используют, чтобы выполнить команду «последовательность»? Определите класс или тип объекта вектор.Б в R.

3. Построить вектор.В из последовательностей цифр от -2.5 до 3.5 с помощью функции, обозначаемой двоеточием.

Определите класс или тип объекта вектор.В в R.

Переведите данные из вектор.В в текстовые.

Проверьте являются ли данные из вектор.В текстовыми.

Переведите данные из вектор.В в количественные (действительные числа)

Проверьте, являются ли данные из вектор.В количественными.

4. Создайте текстовый вектор Г из следующих элементов: Первый класс, Второй класс, Третий класс, Четвертый класс, Пятый класс. С помощью какой функции можно создать текстовый вектор?

Определите класс или тип объекта вектор.Г в R

5. Создайте логический вектор Д с помощью функции, которую если бы R понимал по-русски, можно было бы ввести так:

вектор.Д<– объединить (ЛОЖЬ, ИСТИНА, ЛОЖЬ, ЛОЖЬ, ЛОЖЬ, ИСТИНА)

Определите класс или тип объекта вектор.Д в R

Проверьте являются ли данные из вектор.Д логическими

6. Извлеките первый по счету элемент из вектора А, второй – из вектора Б, третий – из вектора В, четвертый – из вектора Г и пятый – из вектора Д.

7. Замените первый по счету элемент из вектора А цифрой 1, второй – из вектора Б цифрой 2, третий – из вектора В цифрой 3, четвертый – из вектора Г текстом 'элемент4' и пятый – из вектора Д логическим значением ИСТИНА. Можно ли в векторы Г и Д вставить числовые значения?

8. Удалите первый по счету элемент из вектора А, второй – из вектора Б, третий – из вектора В, четвертый – из вектора Г и пятый – из вектора Д.

9. Преобразуйте вектор.В, в котором содержатся действительные числа в вектор целых чисел. Задайте R вопрос содержит ли вектор.В данные в формате целых чисел?

10. Создайте матрицу, состоящую из четырех строк и четырех столбцов, из следующей последовательности цифр seq(2, 32, 2). Обозначьте эту матрицу Матрица4.

11. Извлеките элемент Матрица4, находящийся в третьей строке ее второго столбца.

12. Замените элемент Матрица4, находящийся в третьей строке ее второго столбца, на NaN (англ. 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; сложение бесконечности с бесконечностью противоположного знака; вычисления квадратного

корня отрицательного числа, логарифмирования отрицательного числа, а также в результате всех математических операций с NaN в качестве одного из операндов.

13. Замените второй столбец Матрицы4 последовательностью цифр 10, 12, 14, 16

14. Удалите четвертую строку Матрица4.

15. Создайте массив данных Мой.Массив1 из последовательности чисел 3,6 … 90, состоящий из двух матриц с тремя строками и пятью столбцами. Определите размер Мой.Массив1.

16. Создайте таблицу из данных Мой.Массив1. Задайте R вопрос является ли Таблица1 таблицей? Проверьте структуру данных Мой.Массив1. Проверьте структуру данных Мой.Список1. Назовите

тип данных.

17. Создайте список данных Мой.Список1 из векторов: вектор.А, вектор.Б, вектор.В, вектор.Г, вектор.Д. Проверьте структуру данных Мой.Список1.

Ответы на задание 1 – см. в конце книги.

Глава 2. Анализ фундаментальных факторов, влияющих на курс валюты

Прежде чем приступить к поиску эффективной торговой стратегии займемся анализом фундаментальных факторов, которые, в конечном счете, и определяют стоимость торгуемых валют. С этой целью построим по каждой интересующей нас валюте уравнение регрессии. В него включим те факторы, к которым, по нашей оценке, интересующая нас валюта наиболее чувствительна. Решив уравнение регрессии, найдем расчетное значение курса валюты. В том случае, если расчетное значение ее курса выше фактического, то в этом случае делается вывод, что данная валюта недооценена. Или наоборот, если расчетное значение ниже текущего курса валюты, то тогда делается вывод о том, что она переоценена.

Далее загрузим ежедневные данные по курсам 12 валютам, а также по узкому индексу доллара США (к корзине из шести валют), по ценам на нефть и золоту – за период с 30 июня 1992 года и до 1 апреля 2018 года. Информация по динамике узкого индекса доллара США взята на сайте американской ФРС, по ценам на нефть марки Брент – на сайте Управления по энергетической информации США (EIA), по ценам на золото – на сайте Лондонской биржи драгоценных металлов. Данные по курсам доллара США и евро к рублю взяты на сайте Банка России в разделе «Динамика официального курса заданной валюты».

В то время как остальные курсы по не рублевым валютным парам представляют собой кросс-курсов, рассчитанные на основе данных российского Центробанка. Поскольку евро появился в обороте лишь с 1 января 1999 года, то до этой даты нами использовались данные ЦБ РФ по котировкам ЭКЮ – практически полного аналога (с точки зрения курсовой стоимости) современной единой европейской валюты, которую при переходе к евро в еврозоне обменяли в соотношении 1 к 1.

Как известно, официальные курсы валют к рублю Банк России устанавливает на следующий день исходя из рыночных котировок по итогам утренних торгов на Московской межбанковской валютной биржи. Для определения официального валютного курса ЦБ РФ на конкретный день будем использовать средневзвешенное значение курса доллара, сложившееся по итогам Единой торговой сессии на 11 час. 30 мин. С учетом этого даты по курсам валют сдвинуты нами на один день назад – ко дню торгов, по итогам которых они установлены, так как в этом случае официальные валютные курсы будут лучше коррелировать с фактическими итогами биржевых торгов.

Кроме того, с целью обеспечения сопоставимости данных при расчете всех курсов валют была сделана корректировка на деноминацию рубля. Дело в том, что с 1992 года и по 31 декабря 1997 года Банк России дает курсы валют в неденоминированных рублях, в то время как после деноминации с 1 января 1998 года у российской валюты исчезли три нуля.

Все необходимые для работы с этой книгой ежедневные данные (часть из них Вам в дальнейшем потребуется для самостоятельной работы) по курсам 12 валют, а также аналогичные данные по узкому индексу доллара США (к корзине из шести валют), по ценам на нефть и золоту за период с 30 июня 1992 года и до 28 апреля 2018 года читатель может скачать, открыв публикацию «Как предсказать курс доллара. Поиск доходной стратегии с языком R» по следующему адресу: https://bryukov.blogspot.com/.

В конце этой небольшой публикации есть ссылка, по которой можно скачать файл 'Данные.csv'. Хочу обратить Ваше внимание на тот факт, что в скопированном файле дни торгов в столбце Дата определяются в цифровом формате, исходя из стартовой даты – 1 января 1970 года, то есть с момента начала так называемой эры UNIX (англ. Unix Epoch). В то время как в Excel стартовой датой, от которой ведется отсчет, является 1 января 1900 года.

В том случае, если из-за технического сбоя ссылка с файлом 'Данные.csv' вдруг не сработает, то Вы можете загрузить эти данные, скопировав размещенные на моем блоге четыре материала: «Данные к книге «Как предсказать курс доллара. Поиск доходной стратегии с языком R», ч 1, 2. 3 и 4. Соответственно, ссылки по этим четырем материалам следующие: https://bryukov.blogspot.com/2018/08/r-1.html, https://bryukov.blogspot.com/2018/08/r-2.html, https://bryukov.blogspot.com/2018/08/r-3.html и https://bryukov.blogspot.com/2018/08/r-4.html.

Сначала скопируйте «Данные к книге «Как предсказать курс доллара. Поиск доходной стратегии с языком R» из части 1. После чего разместите эти данные в файле Excel, выделите мышкой столбец с этими данными (четыре столбца не разделены), затем последовательно щелкните по опциям ДАННЫЕ/ТЕКСТ ПО СТОЛБЦАМ. А потом в появившемся окне, которое называется МАСТЕР РАСПРЕДЕЛЕНИЯ ТЕКСТА ПО СТОЛБЦАМ, укажите формат данных – С РАЗДЕЛИТЕЛЯМИ, в котором выберите опцию С ПРОБЕЛАМИ. В результате у Вас в файле Excel получится четыре отдельных столбца с данными.

Далее последовательно скопируйте 2, 3 и 4 части с данными к книге «Как предсказать курс доллара. Поиск доходной стратегии с языком R» и аналогичным образом поделите их на столбцы с помощью МАСТЕРА РАСПРЕДЕЛЕНИЯ ТЕКСТА ПО СТОЛБЦАМ. А затем все 16 столбцов с данными сохраните в экселевском файле в формате csv в рабочей директории. Назовите этот файл 'Данные.csv'. Используйте при этом следующую опцию при сохранении файла в формате csv – «CSV (разделители – запятые)».

Итак, сохранив в своей рабочей директории упомянутые выше данные, приступим к работе. При этом любую сессию в R нужно начинать таким образом:

```
> rm(list=ls(all.names=T))
# команда rm – сокращение от англ. слова remove
# этой командой удаляем все объекты, оставшиеся в памяти после прошлой сессии
# если нужно удалить только часть объектов, то тогда используем команду rm(x, y,z)
# в скобках rm() указываем названия удаляемых переменных x, y и z и т.д.
> ls()
# эту команду вводим, чтобы проверить наличие не удаленных из памяти объектов
# выводит список всех объектов, находящихся в рабочей среде программы;
character(0)
# ответ character(0) говорит о том, что переменных, оставшихся от прошлой сессии, нет
> getwd()
# функция getwd() позволяет узнать, какая директория в данный момент является рабочей
[1] "C:/Users/Vladimir/Documents"
# в ответе содержится название рабочей директории по умолчанию
# если мы хотим изменить рабочую директорию, то вводим команду:
> setwd('C:/Users/Vladimir/Documents/Cloud Mail.Ru/1 ANALITIKA/000 R/000 Книга прогноз доллара с R')
# устанавливаем текущую рабочую директорию, где находятся файлы данных
# не забудьте поменять название рабочей директории, так как у каждого читателя она своя
# в рабочую директорию по умолчанию сохраняются полученные результаты
# для импорта файла не из рабочей директории пишем к нему полный путь
# в этом случае заключайте в кавычки названия файлов и директорий
> install.packages('zoo')
# если бы R понимал по-русски, то эту команду можно было бы ввести так:
# установить.пакеты('zoo')
```

```
> install.packages(‘fBasics’)
# загружаем на диск компьютера пакеты (библиотеки) zoo и fBasics
# не забудьте при загрузке на диск компьютера заковычить название пакетов
> library(zoo)
# если бы R понимал по-русски, то эту команду можно было бы ввести так:
# библиотека(zoo)
> library(fBasics)
# загружаем в память компьютера библиотеки zoo и fBasics
# эти пакеты нужны нам для текущей работы
# перед каждой сессией нужные пакеты надо загружать в память компьютера
# далее читаем файл с загруженными для работы данными:
> Мои.данные<-read.zoo('Данные.csv', sep = ";", header=TRUE, FUN=as.Date)
# если бы R понимал по-русски, то эту команду можно было бы ввести так:
# Мои.данные<-чтение.zoo('Данные. csv', знак раздела = ";", заголовок=ИСТИНА, Функция=as.Date)
# FUN =as.Date по-русски можно было перевести: функция=как.Даты
# as.Date переводит загружаемые данные в нужный формат календарных дат
# в результате файл Excel в формате csv с загружен из рабочей директории
> options("scipen"=100, "digits"=8)
# устанавливаем количество сокращаемых после запятой знаков=8
# избавляемся от экспоненциального формата представления цифр
# options по-русски означет опции, а digits– цифры
# "scipen" – порядок использования научной нотации в R
> head(Мои.данные)
# по умолчанию загружает первые 6 строк файла Мои.данные
# head в переводе на русский означает голова, заголовок, верхняя часть
# если бы R понимал по-русски, то эту команду можно было бы ввести так:
# > начало(Мои.данные)
# смотрим первые 6 строк с загруженными данными – см. рис. 4
```

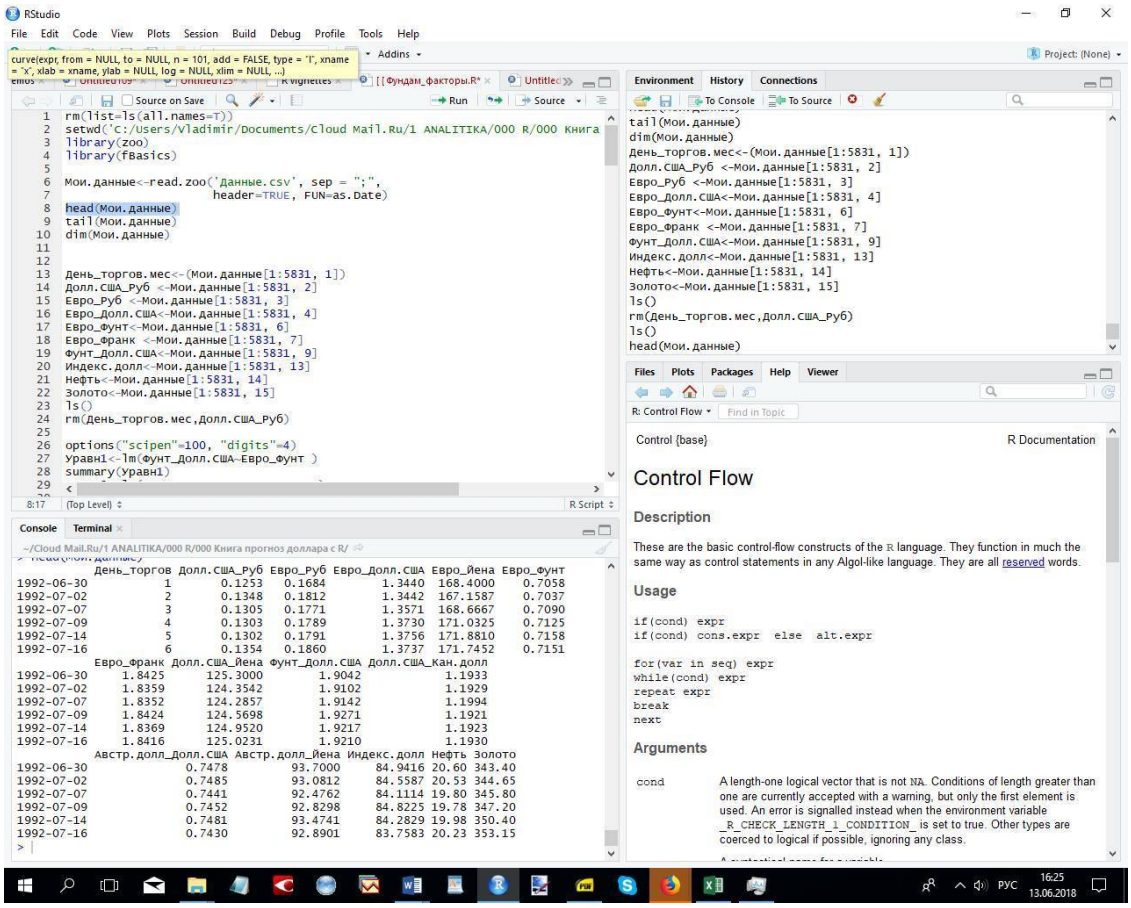


Рис. 4

>tail(Мои.данные)

tail в переводе на русский означает хвост, задняя часть

Смотрим последние 6 строк с загруженными данными– см. рис. 5

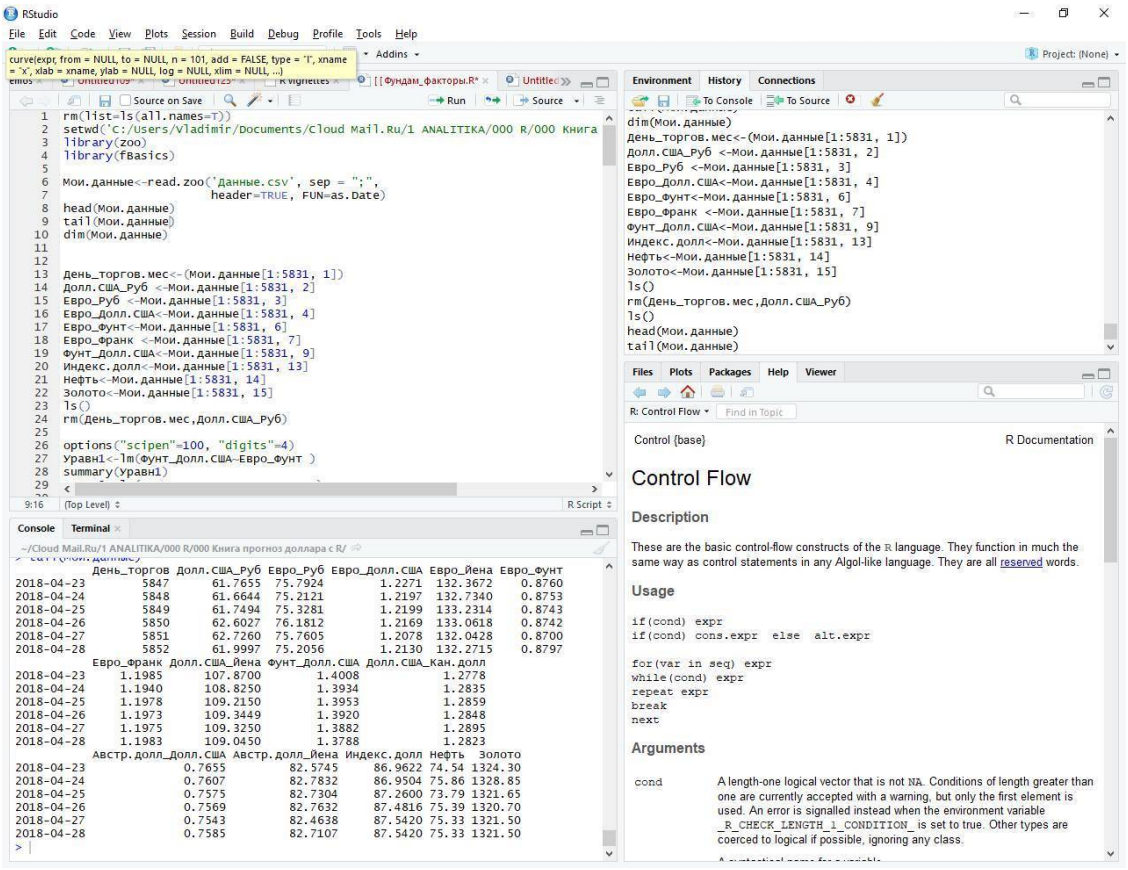


Рис. 5

```
> dim(Мои.данные)
[1] 5852 15
# смотрим количество строк и колонок в загруженном файле.
# всего загружено 5852 строки и 15 колонок с 15 переменными.
> День_торгов.мес<-(Мои.данные[1:5831, 1])
# переменные будем обозначать на кириллице, хотя, как правило, используют латиницу
# в квадратных скобках укажем сначала номера загружаемых строк, а затем номер столбца
# символ 1:5831 означает номера строк сверху вниз по порядку 1,2 … 5830, 5831.
# строки 5832: 5852 с данными за апрель 2018 года пока не будем использовать
# неиспользованные данные будут нам нужны для последующего тестирования
> Долл.США_Руб <-Мои.данные[1:5831, 2]
# Долл.США_Руб <-Мои.данные [1:5831 наблюдений, 2-й столбец загруженного файла]
> Евро_Руб <-Мои.данные[1:5831, 3]
> Евро_Долл.США<-Мои.данные[1:5831, 4]
> Нефть<-Мои.данные[1:5831, 14]
> Золото<-Мои.данные[1:5831, 15]
# присваиваем названия тем загруженным данным, которые сейчас будем использовать
```

Далее построим уравнение регрессии, включив в него в качестве зависимой переменной Долл.США_Руб, а в качестве независимых переменных – фундаментальные факторы, которые, по нашему мнению, влияют на его курс – Евро_Долл.США, Евро_Руб, Нефть и Золото.

Как известно, задачей регрессионного анализа является расчет формулы, описывающей связь между зависимой переменной Y (ее называют также результативным признаком) и неза-

висимыми (их называют также факторными) переменными X1, X2, … , Xn. При этом формула связи результативного признака Y с факторами X1, X2, … , Xn, либо с одним фактором X, получила название уравнения регрессии.

В качестве метода аппроксимации (приближения) в уравнении регрессии используется метод наименьших квадратов, который минимизирует сумму квадратов отклонений фактических значений Y от его предсказываемых значений, рассчитанных по определенной математической формуле. При этом линию, которая лучше всего подойдет к этим данным, выбирают так, чтобы сумма квадратов значений вертикальных отклонений зависимой переменной (фактического курса доллара) от линии, рассчитанной по уравнению регрессии (предсказанный курс доллара), была минимальной.

Подробнее о том, как решить уравнение регрессии, можно прочитать в моей книге «Как предсказать курс доллара. Эффективные методы прогнозирования с использованием Excel и EViews» -см. главу 2. «Метод наименьших квадратов и решение уравнения регрессии в Excel», а также в другой моей книге «Как предсказать курс доллара. Расчеты в Excel для снижения риска проигрыша» – см. главу 2. «Как в Excel решить однофакторное уравнение регрессии для линейного тренда».

Решение уравнения регрессии с одной зависимой переменной Долл.США_Руб и четырьмя независимыми переменными, обозначающими воздействие на эту валюту четырех фундаментальных факторов, в R примет следующий вид:

```
> Уравн1<-lm(Долл.США_Руб~Евро_Долл.США+Евро_Руб+Нефть+Золото)
# решаем уравнение регрессии
#сокращение lm означает уравнение, решаемое методом наименьших квадратов
# символ ~ отделяет зависимую переменную Долл.США_Руб от независимых переменных
# между зависимыми переменными ставится знак +.
> summary(Уравн1)
# summary означет краткий вывод, итоги решения уравнения
```

В результате получаем табл. 1 с выводом данных по итогам решения этого уравнения регрессии.

Табл. 1. Вывод данных по итогам решения уравнения регрессии

```
Call:
lm(formula = Долл.США_Руб ~ Евро_Долл.США + Евро_Руб + Нефть + Золото)

Residuals:
Min 1Q Median 3Q Max
–3.745 -0.580 -0.061 0.385 6.123

Coefficients:
Estimate Std. Error t value Pr(>|t|)
(Intercept) 26.6701534 0.1319762 202.08 <0.0000000000000002 ***
Евро_Долл.США -22.5209932 0.1194950 -188.47 <0.0000000000000002 ***
Евро_Руб 0.8717064 0.0010245 850.84 <0.0000000000000002 ***
Нефть -0.0168154 0.0008252 -20.38 <0.0000000000000002 ***
Золото 0.0001949 0.0000611 3.19 0.0014 **
–
Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Residual standard error: 1 on 5826 degrees of freedom
Multiple R-squared: 0.996, Adjusted R-squared: 0.996
F-statistic: 3.83e+05 on 4 and 5826 DF, p-value: <0.0000000000000002
Источник: расчеты автора

Проверим решение уравнения 1 на статистическую значимость. Судя по табл. 1, F-statistic у данного уравнения регрессии имеет p -value: <0.0000000000000002, то есть уровень его надежности более 99% = ((1-0.0000000000000002)*100). Таким образом можно сделать вывод, что в целом мы получили статистически значимое уравнение регрессии. Также близки к нулю p -value и у константы этого уравнения регрессии (Intercept) и у коэффициентов переменных Евро_Долл.США, Евро_Руб, Нефть и Золото, следовательно, из этого можно сделать вывод, что они также статистически значимы с 99% уровнем надежности.

Случайный (стохастический) процесс X называют стационарным в сильном смысле, если совместное распределение вероятностей всех его лаговых переменных X1, X2, …, Xn, точно такое же, как и для переменных X1+t, X2+t, …, Xn+t. Заметим, что стационарных процессов в сильном смысле на финансовых рынках практически нет, а вот стационарные процессы в более широком слабом смысле активно используются для прогнозирования динамики финансовых активов. Под стационарным процессом в слабом смысле понимается случайный процесс, в котором его среднее значение и дисперсия (разброс, отклонения от средней) не зависят от рассматриваемого времени, а автоковариация (корреляционная связь) зависит только от длины лага между лаговыми переменными.

Далее проверим все переменные (временные ряды), включенные в уравнение 1 на стационарность. С этой целью будем использовать расширенный тест Дикки-Фуллера. Как известно, временной ряд считается стационарным в слабом смысле, если построенное на основе его уравнение первого порядка имеет коэффициент регрессии $\rho < 1$. Например, уравнение Xt= 0.975*Xt-1 +C, – свидетельствует о том, что временной ряд Xt с константой C и коэффициентом $\rho = 0.975 < 1$ является стационарным в слабом смысле. Поэтому в этом временном ряде иногда может возникать сильная волатильность, но в случае стационарности в широком смысле она постепенно затухает. Соответственно, если бы коэффициент ρ был бы больше 1, то тогда этот временной ряд считался бы нестационарным, а, следовательно, волатильность в этом ряде не имела бы тенденции с течением времени не затухать.

Подробнее о стационарности временных рядов можно прочитать в моей книги «Как предсказать курс доллара. Эффективные методы прогнозирования с использованием Excel Задание и EViews» – см. главу 1 «Понятие о стационарном и нестационарном временном ряде, выявление нестационарности ряда графическим способом».

Проверка авторегрессионного процесса на стационарность проводится следующим образом. Согласно нулевой гипотезе, предполагается, что если $\rho=1$, то временной ряд считается нестационарным, а в случае ее опровержения принимается альтернативная гипотеза, утверждающая, что $\rho < 1$, а, следовательно, ряд стационарный. В ходе решения обычного уравнения регрессии рассчитывается t– статистика для коэффициента регрессии ρ, совпадающая с расчетными значениями статистики Дикки-Фуллера, которая потом сравнивается с критическими значениями статистики Дикки-Фуллера (обычно в книгах они даются в специальных таблицах, но в R мы их получаем в готовом виде).

Поскольку проверка гипотезы проводится по одностороннему критерию, то в этом случае, если расчетное значение t-статистики для коэффициента регрессии ρ будет ниже критического значения статистики расширенного теста Дикки-Фуллера (с поправкой на число наблюдений), то в этом случае нулевая гипотеза о том, что $\rho = 1$ отклоняется и принимается альтернативная гипотеза о том, что $\rho < 1$, а, следовательно, тестируемый временной ряд можно считать стационарным. Для того, чтобы провести расширенный тест Дикки-Фуллера загружаем

для текущей работы пакет urca. Если его еще нет на Вашем компьютере, то воспользуйтесь командой install.packages('urca'), а затем введите следующий код:

```
> library(urca)
> Долл.США_Руб.адф <– ur.df(Долл.США_Руб, type = "drift")
# проводим расширенный тест Дикки-Фуллера
# опция теста type = "drift" означает константу
> summary(Долл.США_Руб.адф)
# вывод итогов теста
```

В результате получаем табл. 2 с выводом данных по итогам выполнения теста расширенного теста Дикки-Фуллера.

Табл. 2. Вывод данных по итогам выполнения теста расширенного теста Дикки-Фуллера

```
###############################################
# Augmented Dickey-Fuller Test Unit Root Test #
###############################################

Test regression drift

Call:
lm(formula = z.diff ~ z.lag.1 + 1 + z.diff.lag)

Residuals:
Min 1Q Median 3Q Max
–8.165 -0.052 -0.014 0.035 6.641

Coefficients:
Estimate Std. Error t value Pr(>|t|)
(Intercept) 0.019965 0.011019 1.81 0.07 .
z.lag.1 -0.000343 0.000327 -1.05 0.29
z.diff.lag -0.002071 0.013103 -0.16 0.87
–
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.405 on 5826 degrees of freedom
Multiple R-squared: 0.000194,      Adjusted R-squared: -0.000149
F-statistic: 0.566 on 2 and 5826 DF, p-value: 0.568
Value of test-statistic is: -1.05 2.267
Critical values for test statistics:
1pct 5pct 10pct
tau2 -3.43 -2.86 -2.57
phi1 6.43 4.59 3.78
```

Источник: расчеты автора

Из табл. 2 следует, что значение тестовой статистики (Value of test-statistic) = -1.05 (рядом стоящая цифра 2.267 оценивает значимость включенной в тест константы), то есть выше критического значения tau2=-2.57 для 10% уровня значимости (или, что тоже самое для 90% уровня надежности = 100% – 10% уровень значимости). Таким образом нулевая гипотеза о наличии единичного корня не может быть отклонена, а потому временной ряд Долл.США_Руб нельзя считать стационарным. Аналогичным образом проверим на стационарность и другие

временные ряды по всем факторам, включенным в уравнение регрессии 1. С этой целью введем следующие восемь строк кода:

```
> Евро_Долл.США.адф <– ur.df(Евро_Долл.США, type = "drift")
> summary(Евро_Долл.США.адф)

> Евро_Руб.адф <– ur.df(Евро_Руб, type = "drift")
> summary(Евро_Руб.адф)

> Нефть.адф <– ur.df(Нефть, type = "drift")
> summary(Нефть.адф)

> Золото.адф <– ur.df(Золото, type = "drift")
> summary(Золото.адф)
```

По итогам всех этих четырех тестов с помощью функции summary() нам удалось выяснить по каждому временному ряду, включенному в уравнение регрессии 1, что все они нестационарные. Делается это аналогичным образом, как и при анализе результатов, полученных нами с помощью функции summary(Долл.США_Руб.адф). Итоги этого анализа в целях экономии места в данном случае не приводятся.

Теперь проверим на стационарность остатки, полученные после решения уравнения регрессии 1. Заметим, что под остатками в данном случае имеется в виду разница между фактическими значениями Долл.США_Руб и их расчетными значениями, найденными по уравнению регрессии 1. Остатки в нашем коде обозначим как Уравн1$residuals, а в функцию ur.df() введем type = "none", то есть расширенный тест Дикки-Фуллера будет проводиться, исходя из того, что в остатках нет ни константы, ни тренда. (В последнем случае в функции ur.df() нужно было бы поставить опцию type = "trend").

```
>Долл.США_Руб.ост_адф <– ur.df(Уравн1$residuals, type = "none")
>summary(Долл.США_Руб.ост_адф)
```

По итогам тестирования команда summary(Долл.США_Руб.ост_адф) выдаст нам следующие итоги – см. табл. 3.

Табл. 3. Вывод данных по итогам выполнения расширенного теста Дикки-Фуллера по остаткам, полученным после решения уравнения 1

```
###############################################
# Augmented Dickey-Fuller Test Unit Root Test #
###############################################

Test regression none

Call:
lm(formula = z.diff ~ z.lag.1 – 1 + z.diff.lag)

Residuals:
Min 1Q Median 3Q Max
–1.2278 -0.0297 -0.0006 0.0274 1.2447

Coefficients:
```

```
Estimate Std. Error t value Pr(>|t|)
z.lag.1 -0.00751 0.00149 -5.03 0.0000005 ***
z.diff.lag 0.02879 0.01308 2.20 0.028 *
–
Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.114 on 5827 degrees of freedom
Multiple R-squared: 0.00497,      Adjusted R-squared: 0.00463
F-statistic: 14.6 on 2 and 5827 DF, p-value: 0.000000492

Value of test-statistic is: -5.031

Critical values for test statistics:
1pct 5pct 10pct
tau1 -2.58 -1.95 -1.62
```

Источник: расчеты автора

Как видим, значение тестовой статистики (Value of test-statistic) = -5.031, то есть ниже tau2=-2.58 – критического значения (Critical values for test statistics) для 1% уровня значимости (или, что тоже самое, для 99% уровня надежности). Следовательно, нулевая гипотеза о наличии единичного корня отклоняется с 99% уровнем надежности, а, потому остатки, полученные по итогам решения уравнения регрессии 1, носят стационарный характер. Как мы уже выяснили, эти остатки представляют собой линейную комбинацию нестационарных временных рядов, состоящих из четырех факторов (Евро_Долл.США+Евро_Руб+Нефть+Золото) с зависимой переменной Долл.США_Руб. При этом наличие стационарных остатков – при нестационарности временных рядов факторов, включенных в это уравнение регрессии – свидетельствует о наличии коинтеграции между всеми этими переменными, включенными в уравнение 1. Коинтеграция временных рядов значительно упрощает процесс их анализа, а также свидетельствует о совпадении их динамики в течение весьма длительного времени.

Таким образом для того, чтобы выявить коинтеграцию между временными рядами необходимо сделать следующее: во-первых, построить и решить статистически значимое уравнение регрессии; во-вторых, проверить временные ряды переменных, включенные в уравнение регрессии на стационарность с помощью расширенного теста Дикки-Фуллера; в-третьих, в случае, если эти временные ряды не стационарны, провести тестирование остатков уравнения регрессии на стационарность; в-четвертых, если расширенный тест Дикки-Фуллера покажет, что эти остатки стационарны, то можно сделать вывод о наличии коинтеграции между временными рядами, включенными в уравнение регрессии.

Посмотрим, какими были остатки, полученные в результате решения уравнения регрессии 1, в течение 20 торговых дней марта 2018 года:

```
>tail(Уравн1$residuals, 20)
# по умолчанию команда tail дает только 6 последних наблюдений
# поэтому цифра 20 специально указана в команде tail
```

Табл. 4. Вывод остатков, полученных после решения уравнения 1.

2018-03-01	2018-03-02	2018-03-05	2018-03-06	2018-03-07	2018-03-12	2018-03-13
-1.969	-2.129	-2.214	-2.312	-2.551	-2.307	-2.318
2018-03-14	2018-03-15	2018-03-16	2018-03-19	2018-03-20	2018-03-21	2018-03-22
-2.507	-2.455	-2.332	-2.202	-2.406	-2.160	-2.361
2018-03-23	2018-03-26	2018-03-27	2018-03-28	2018-03-29	2018-03-30	
-2.284	-2.409	-2.653	-2.496	-2.259	-2.263	

Источник: расчеты автора

Как известно, в статистической науке остатками называют разницу между фактическим и расчетным значением переменной, найденным после решения уравнения регрессии. Судя по табл. 4, все остатки за 20 дней торгов марта 2018 года были со знаком минус, то есть отрицательные. Таким образом можно сделать вывод, что курс доллара к рублю с точки зрения воздействия на него фундаментальных факторов оказался недооцененным. Причем, в период с 1 марта по 30 марта 2018 года эта недооценка американского доллара выросла с -1.969 руб. до -2.263 руб.

По итогам тестирования мы также выяснили, что остатки, полученные по итогам решения уравнения 1, являются стационарными, то есть имеют тенденцию колебаться вокруг их среднего значения, равного нулю. В связи с этим, имеет смысл найти среднюю полуамплитуду их колебаний вокруг 0, то есть найти среднестатистическое время, в течение которого остатки, опустившись до своего локального минимума, поднимаются до своего очередного локального максимума. Найдем эту полуамплитуду таким образом:

```
коэф.возврата<– summary(Долл.США_Руб.ост_адф)@testreg$coefficients[1,1]
# коэффициент @testreg$coefficients[1,1] берем из summary(Долл.США_Руб.ост_адф)
полупериод.средней <-(-log(2)/коэф.возврата)
# log(2) – натуральный логарифм от 2
полупериод.средней
# [1] 92.33331
```

Таким образом, средняя полуамплитуда колебаний остатков по курсу доллара к рублю составляет 92.33 торговых дня. По мнению Ernest P. Chan, автора книги «Algorithmic Trading. Winning Strategies and Their Rationale», pp 46-47 (Эрнест П. Чан. «Алгоритмическая торговля. Стратегии выигрыша и их обоснование», стр. 46-47), знание полуамплитуды колебания позволяет инвестору оценить период возврата его вложений в валюту. На наш взгляд, на практике сделать это очень непросто, поскольку фактические полуамплитуды колебаний значительно отличаются от их среднего значения. Чтобы это наглядно показать читателю, построим график колебаний остатков за период с 30 июня 1992 г. по 30 марта 2018 г. С этой целью введем следующий код:

```
plot(Уравн1$residuals[1:5831], main='Полупериоды колебания остатков', xlab='Годы', type='l')
# строим график остатков
# plot означает график
abline(h=0, lwd=4)
```

```
# приставка ab означает от, a line – линия
# строим линию =0 с толщиной lwd=4
# lwd сокращение от line width, что означает ширину линии
```

В результате получаем рис. 6 с графиком колебаний остатков вокруг нуля за период с 30 июня 1992 г. по 30 марта 2018 г. Толстая прямая линия на этом графике представляет собой линию перехода остатков от отрицательных к положительным значениям. Судя по этому графику, различные циклы колебаний остатков вокруг нуля довольно существенно отличаются друг от друга, а также и от их среднего значения.

Поэтому можно сделать вывод, что знание средней за весь период наблюдений полуамплитуды колебаний остатков не дает инвестору возможности точно оценить, какими будут остатки в момент его выхода на рынок. Если же трейдер все-таки решит приобрести недооцененный актив, то он должен понимать, что срок получения прибыли по такого рода сделкам точно неизвестен. При этом с точки зрения риск-менеджмента лучше ориентироваться на максимальные сроки длительности полуамплитуды колебаний остатков.

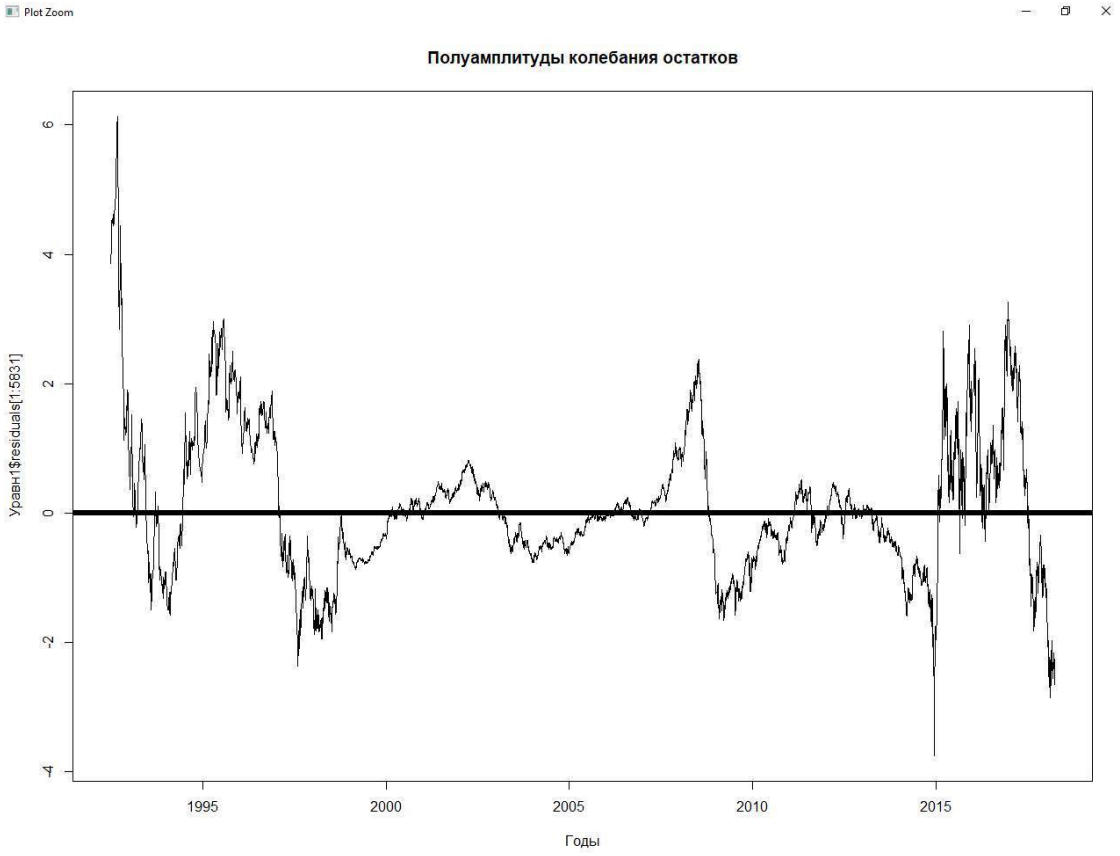


Рис. 6

После того как мы создали вышеуказанный код его нужно будет сохранить. Щелкнув в верху RStudio опции File и Save as (Ctrl+S), мы по умолчанию сохраняем созданный скрипт в текущей рабочей директории. Если скрипт сохраняется в первый раз, то появится окно, в котором будет предложено ввести имя сохраняемого файла. При этом файл для дальнейшей работы в R сохраняется с расширением *.R, хотя в некоторых случаях Вам могут потребоваться и другие форматы.

Задание 2. Проанализировать влияние фундаментальных факторов на курс евро к рублю

После того, как читатель познакомился с анализом фундаментальных факторов, влияющих на курс американского доллара к рублю, пора применить эти знания на практике. В данном случае читателю нужно будет внести небольшие изменения в код, анализирующий фундаментальные факторы, влияющие на курс доллара США к рублю. В результате он получит код, анализирующий влияние фундаментальных факторов на курс евро к рублю.

С этой целью ниже дается код, с помощью которого проводился анализ фундаментальных факторов, влияющих на курс доллара США к рублю. Здесь и ниже во всех заданиях автор использует в коде символ решетки с тремя восклицательными знаками #!!!, который ставится ниже строки кода. Символ #!!! является для читателя указанием на то, что данная строка с кодом должны быть им во время решения задания изменены. При необходимости после символа #!!! даются иногда и пояснения о необходимости изменить расположенную выше строку кода.

```
rm(list=ls(all.names=T))
getwd()

setwd('C:/Users/Vladimir/Documents/Cloud Mail.Ru/1 ANALITIKA/000 R/000 Книга прогноз доллара с R')
ls()
library(zoo)
library(fBasics)
library(urca)
Мои.данные<-read.zoo('Данные.csv', sep = ";", header=TRUE, FUN=as.Date)
head(Мои.данные)
tail(Мои.данные)
dim(Мои.данные)

День_торгов.мес<-(Мои.данные[1:5831, 1])
Долл.США_Руб <-Мои.данные[1:5831, 2]
Евро_Руб <-Мои.данные[1:5831, 3]
Евро_Долл.США<-Мои.данные[1:5831, 4]
Нефть<-Мои.данные[1:5831, 14]
Золото<-Мои.данные[1:5831, 15]

options("scipen"=100, "digits"=4)

Уравн1<-lm(Долл.США_Руб~Евро_Долл.США+Евро_Руб+Нефть+Золото)
summary(Уравн1)
#!!! какую зависимую переменную нужно поменять в Уравн1
#!!! какую независимую переменную нужно поменять в Уравн1
#!!! обратите внимание на значимость Pr(>|t|) коэффициентов в Уравн1
Долл.США_Руб.адф <– ur.df(Долл.США_Руб, type = "drift")
summary(Долл.США_Руб.адф)
Евро_Долл.США.адф <– ur.df(Евро_Долл.США, type = "drift")
```

```
summary(Евро_Долл.США.адф)
Евро_Руб.адф <– ur.df(Евро_Руб, type = "drift")
summary(Евро_Руб.адф)

Золото.адф <– ur.df(Золото, type = "drift")
summary(Золото.адф)

Нефть.адф <– ur.df(Нефть, type = "drift")
summary(Нефть.адф)

Долл.США_Руб.ост_адф <– ur.df(Уравн1$residuals, type = "none")
#!!! тестируем остатки, полученные по итогам уравнения регрессия
#!!! с какой зависимой переменной решено это уравнение
summary(Долл.США_Руб.ост_адф)
#!!!
tail(Уравн1$residuals, 20)
#!!!
коэф.возврата<– summary(Долл.США_Руб.ост_адф)@testreg$coefficients[1,1]
#!!! коэф.возврата для какой зависимой переменную нужно протестировать
полупериод.средней <-(-log(2)/коэф.возврата)
полупериод.средней
# 92.33 торговых дней полупериод.средней

plot(Уравн1$residuals[1:5831], type='l')
#!!!
abline(h=0, lwd=4)
plot(Уравн1$residuals[1:5831], main='Полупериоды колебания остатков', xlab='Годы',
type='l')
#!!!
abline(h=0, lwd=4)
```

Ответы на задание 2 – см. в конце книги.

Глава 3. Каким должно быть кредитное плечо, чтобы не проторговаться

Как мы уже говорили, любой выход на валютный рынок сопряжен с большим риском потерь. Поэтому для того, чтобы минимизировать риски трейдеру перед выходом на рынок необходимо провести определенные расчеты. Прежде чем заняться расчетами по минимизации рисков сначала введем следующий код:

```
> rm(list=ls(all.names=T))
# удаляем все объекты, оставшиеся после прошлой сессии.
> setwd('C:/Users/Vladimir/Documents/Cloud Mail.Ru/1 ANALITIKA/000 R/000 Книга прогноз доллара с R')
# устанавливаем рабочую директорию.
> library(zoo)
# загружаем в память компьютера библиотеку zoo

> Мои.данные<-read.zoo('Данные.csv', sep = ";", header=TRUE, FUN=as.Date)
# снова загружаем наши данные из файла Данные.csv
> head(Мои.данные)
# смотрим первые 6 строк с загруженными данными
> tail(Мои.данные)
# Смотрим последние 6 строк с загруженными данными
> dim(Мои.данные)
# Смотрим, сколько наблюдений (строк) и переменных (колонок) в файле Данные.csv
[1] 5852 15
> options("scipen"=100, "digits"=4) )
# устанавливаем количество сокращаемых после запятой знаков
# избавляемся от экспоненциального формата представления цифр
> Курс <-Мои.данные[1:5831 ,2]
# загружаем данные по курсу доллара США к рублю с 30 июня 1992 г. по 30 марта 2018 г.
> Курс0<-Курс[5831, ]
# обозначаем как Курс0 – курс доллара США к рублю на 30 марта 2018 г.
> Курс0
2018-03-30
57.2649
# смотрим курс доллара США к рублю на 30 марта 2018 г.
> Лот<-1000
# устанавливаем размер торгуемого микролота=1000 долларам США
> Цена_Лота<-Лот*Курс0
# находим цену лота в рублях на 30 марта 2018 г.
> Цена_Лота
2018-03-30
57264.9
# цена лота в рублях на дату покупки лота по курсу закрытия
```

Теперь предположим, что в данном случае мы планируем торговать в течение всего апреля 2018 года, то есть в течение 21 торгового дня. При этом в зависимости от характера торговых сигналов (о них мы подробнее поговорим немного позже) трейдер будет открывать или закрывать длинную (короткую) позицию по ценам закрытия на конец торгового дня, либо воздерживаться от участия в торгах. С целью риск-менеджмента составим интервальный прогноз, в рамках которого курс доллара к рублю на момент закрытия торгов будет находится с 1 апреля по 28 апреля 2018 года.

При этом будем исходить из того, что с 50% вероятностью курс прогнозируемой валюты может за месяц торгов либо упасть, либо вырасти. Заметим, что в биржевой торговле вероятность удачно закрыть позицию, чаще всего, колеблется в диапазоне 45,0%-55,0% и лишь очень хорошему трейдеру при удачном стечении обстоятельств на рынке удается повысить ее до 60,0-65,0% и чуть более. Поэтому хочу еще раз повторить, что любой выход на валютный рынок сопряжен с серьезным риском потерь, который нельзя совсем устранить, а можно лишь минимизировать.

Для прогнозирования нам необходимо найти так называемые квантили, а, точнее сказать процентили, по изменениям курса интересующей нас валюты. Квантилем в математической статистике называют такое значение, которое заданная случайная величина (в данном случае это разница между текущим и предыдущими курсами доллара с лагом в один, два, три … 250 торговых дней) не окажется выше (либо ниже) определенного уровня с заранее заданной вероятностью. Если вероятность задана в процентах, то квантиль называется процентилем. Подробнее о процентилях можно прочитать в моей книге см. главу 1 «Как предсказать курс доллара. Расчеты в Excel для снижения риска проигрыша».

Для того, чтобы найти процентили, сначала на основе данных по курсу закрытия курса доллара к рублю на конец 30 марта 2018 г. построим прогноз на следующий торговый день на момент закрытия с 0.1 % уровнем риска (или 99.9% уровнем надежности). Причем, 0.1% риска свидетельствует о том, что существует лишь один шанс из 1000, что курс валюты выйдет за рамки интервального прогноза. (Заметим, что эта оценка риска основана на прошлых данных, но после сильных колебаний на рынке, она может измениться). Для наших расчетов воспользуемся функцией quantile, в которой для верхнего интервала прогноза установим уровень надежности =0.999, а для нижнего интервала прогноза – на уровне 0.001, что по сути означает уровень надежности =1-0.001=0.999 или 99.9%. При этом команда diff(Курс, 1) означает разницу между курсами валют с лагом в один день.

Для получения интервального прогноза с лагом в один торговый день с 0.1% уровнем риска введем следующий день:

```
> НижПрогноз <– Курс0+quantile(diff(Курс, 1), 0.001)
> НижПрогноз
2018-03-30
54.458959
> ВерхПрогноз<-Курс0+quantile(diff(Курс, 1), 0.999)
> ВерхПрогноз
2018-03-30
60.808652
```

Таким образом на основе этого прогноза, построенного по данным на 30 марта 2018 г., можно с 99.9% уровнем надежности утверждать, что курс доллара к рублю по итогам торгов в следующий торговый день, то есть 2 апреля 2018 г., должен был находиться в диапазоне от 54.4590 руб. и до 60.8087 руб. Если же нужно построить прогноз с лагом в 2,3,4 … 250 торговых дней, то одна из команд в этом коде приобретет вид diff(Курс, 2,3,4 … 250). А в случае использования операторов циклов вместо этих конкретных цифр ставят букву n.

Далее построим аналогичные интервальные прогнозы на период от одного, двух … и до 250 торговых дней (в календарном году около 250 торговых дней). Для прогнозов будем использовать данные по курсам доллара к рублю за период с 30 июня 1992 г. по 30 марта 2018 г. С этой целью введем код, в котором используется оператор циклов for(n in 1:250), «пробегающий» значения от 1 до 250. При этом в R используются фигурные скобки { }, чтобы создать новую функцию, в которой «упаковано» сразу несколько операторов. В некоторых ситуациях фигурные скобки можно опускать, тем не менее начинающим программистам рекомендуется всегда их ставить.

Прогноз по курсу валюты с заданным количеством дней торгов и 0.1% уровнем риска:

```
Вывод.данных<-data.frame()
# создаем таблицу для вывода данных с 0 колонок и 0 строк
for(n in 1:250) {
# оператор циклов for
# вектор 1:250 означает последовательность 1, 2 … 250.
НижПрогноз<-round(Курс0+quantile(diff(Курс, n), 0.001),digits=4)
# нижний интервал прогноза, digits=4 – округлить на 4 цифры
# функция round означает округлить
ВерхПрогноз<-round(Курс0+quantile(diff(Курс, n), 0.999),digits=4)
# верхний интервал прогноза
Вывод.данных<-c(Вывод.данных, data.frame(n, НижПрогноз, ВерхПрогноз))
# заполняем таблицу данных
show(data.frame(n, НижПрогноз, ВерхПрогноз, row.names='Дней торгов'))
# команда показать вывод данных
>Табл.05 <– data.frame(matrix(unlist(Вывод.данных0), nrow=250, byrow=T), stringsAsFactors=FALSE)
# преобразуем лист с выводом данных в табл.05
>ДнейТоргов <-Табл.05[, 1]
# присваиваем колонке 1 из табл. 05 имя первой переменной
НижПрогноз <-Табл.05[, 2]
# присваиваем колонке 2 из табл. 05 имя второй переменной
ВерхПрогноз<-Табл.05[, 3]
# присваиваем колонке 3 из табл. 05 имя третьей переменной
Табл.5 <-data.frame(ДнейТоргов, НижПрогноз, ВерхПрогноз)
# создаем таблицу с тремя обозначенными переменными для последующей работы
Табл.5
# выводим таблицу с тремя обозначенными переменными – см. табл. 5
```

В результате получим следующие интервальные прогнозы (в целях экономии места, здесь они даются не полностью) – см. табл. 5.

>Табл. 5. Нижние и верхние интервальные прогнозы с лагом от 1 до 250 дней

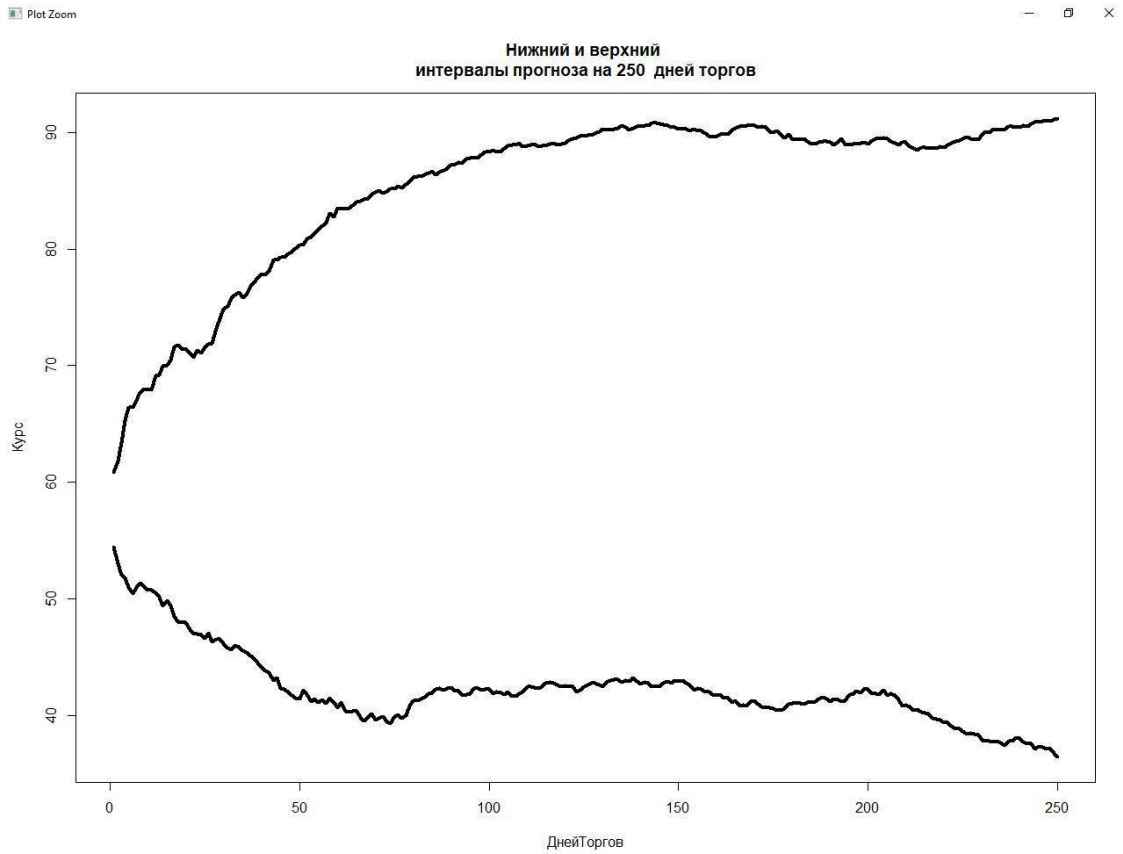
	ДнейТоргов	НижПрогноз	ВерхПрогноз
1	1	54.459	60.8087
2	2	53.026	61.7846
3	3	52.0593	63.4151
4	4	51.8053	65.2721
5	5	50.9782	66.4082
6	6	50.4451	66.4435
7	7	51.0002	66.9953
8	8	51.3413	67.6751
9	9	51.0357	67.9941
10	10	50.752	67.9521
11	11	50.7146	67.9568
12	12	50.4898	69.1079
13	13	50.1687	69.182
14	14	49.3582	69.9191
15	15	49.8801	70.0094
16	16	49.3731	70.4895
17	17	48.4774	71.5747
18	18	47.9649	71.7313
19	19	47.9371	71.4089
20	20	47.9913	71.487
21	21	47.3384	71.1137
22	22	46.9892	70.7176
23	23	46.9587	71.2865
24	24	46.9299	71.0846
25	25	46.6069	71.5759
250	250	36.4774	91.1698

Источник: расчеты автора

Судя по табл. 5, своего максимума верхний интервал прогноза =91.1698 руб. достигает при прогнозе на 250 дней. Такой же лаг и у минимума нижнего интервала прогноза= 36.4774 руб. Воспользуемся соответствующими встроенными функциями, чтобы не тратить время на поиск этих важных для трейдера параметров интервальных прогнозов.

```
> max(ВерхПрогноз)
# находим максимум по верхнему интервалу прогноза
[1] 91.1698
> which.max(ВерхПрогноз)
# находим в таблице порядковый номер строки максимума по верхнему интервалу прогноза
[1] 250
> min(НижПрогноз)
# находим минимум по нижнему интервалу прогноза
[1] 36.4774
> which.min(НижПрогноз)
# находим в таблице порядковый номер минимума по нижнему интервалу прогноза
[1] 250
> НижПрогноз[21]
# находим нижний интервал прогноза с лагом в 21 день
[1] 47.3384
> ВерхПрогноз[21]
# верхний интервал прогноза с лагом в 21 день
[1] 71.1137
>plot(НижПрогноз, main = 'Интервальный прогноз на 250 торг. дней',
type='l', lwd=4, ylim=c(min(НижПрогноз), max(ВерхПрогноз)))
>lines(ВерхПрогноз, lwd=4)
# делаем график интервального прогноза с лагом на 250 торговых дней
```

В результате получаем следующий график – см. рис. 7



Источник: расчеты автора

Рис. 7

На рис. 7 хорошо видно, что в целом, особенно, на первом этапе (для нижнего интервала – до лага в 75 дней торгов, а для верхнего – до 100 дней торгов) диапазон интервального прогноза по мере увеличения временного лага резко расширяется. Однако затем это расширение замедляется, а временами иногда даже немного уменьшается, хотя потом снова временами слегка подрастает. Это объясняется тем, что наши интервальные прогнозы рассчитаны на основе фактических изменений курса валюты с различным временным лагом. В то время как при расчете интервальных прогнозов, исходя из теоретического нормального распределения, таких колебаний не наблюдалось бы, но зато у них есть другой серьезный недостаток. Дело в том, что интервальные прогнозы, построенные на основе нормального распределения с 70%-90% уровнем надежности, как правило, существенно завышают вероятность волатильности валюты, на уровне 95% надежности близки к фактической волатильности, а на 99% и 99.9% уровнях надежности занижают этот риск.

Далее на основе данных за период с 30 июня 1992 г. по 30 марта 2018 г. мы сначала попытаемся разработать торговую систему, а затем ее протестировать на основе данных за весь апрель 2018 г. Поскольку в этом месяце насчитывается 21 день, в течение которых мы будем торговать, то, вполне логичным будет посмотреть, каким будет интервальный прогноз на конец этого периода. С этой целью мы введем следующий код:

```
> НижПрогноз[21]
[1] 47.3384
> ВерхПрогноз[21]
[1] 71.1137
```

Благодаря командам НижПрогноз[21] и ВерхПрогноз[21] нам удалось выяснить, что при лаге в 21 торговый день верхний и нижний интервалы прогнозов равняются, соответственно, 47.3384 руб. и 71.1137 руб. Таким образом наши расчеты, выполненные на основе имеющихся на 30 марта 2018 г. данных, с 99.9% уровнем надежности предсказывают, что на конец апреля 2018 г. курс доллара к рублю должен находиться в указанном диапазоне. В тот момент, когда пишутся эти строки, уже известно, что в течение апреля 2018 года, курс доллара к рублю находился в диапазоне от 57.285 руб. и до 64.0626 руб., то есть не выходил за рамки интервального прогноза.

Если бы мы захотели построить интервальные прогнозы на конец апреля 2018 г. с более низким 1% уровнем надежности, то нам следовало бы ввести следующий код:

```
> НижПрогноз <– Курс0+quantile(diff(Курс, 21), 0.01)
> НижПрогноз
2018-03-30
52.301041
> ВерхПрогноз<-Курс0+quantile(diff(Курс, 21), 0.99)
> ВерхПрогноз
2018-03-30
65.302888
```

В этом случае верхние и нижние интервалы прогнозы равнялись бы, соответственно, 52.3010 руб. и 65.3029 руб., но риск выхода курса доллара к рублю оказался бы выше.

После того как мы нашли диапазон возможного роста и падения курса доллара к рублю, как по итогам месячных торгов в апреле 2018 г., так и в целом с лагом от одного до 250 торговых дней, то теперь перед нами стоит новая задача, а именно: рассчитать кредитное плечо с 0.1% уровнем надежности. То есть нужно вычислить, какой должен быть у трейдера залог (объем собственных средств), используемых при покупке валюты, чтобы его позиция не была принудительно закрыта из-за нехватки у него средств в случае неблагоприятного движения рынка.

Для дальнейших расчетов будем использовать только что составленные нами верхние и нижние интервалы прогнозов (их мы обозначили как НижПрогноз и ВерхПрогноз), составленные на срок от 1,2.,3 … и до 250 дней. С этой целью введем следующий код, с помощью которого найдем долю привлеченного кредита, необходимого для покупки (или продажи) лота при открытии длинной (или короткой) позиции сроком от 1,2.,3 … и до 250 дней:

```
# Доля кредита при покупке или продаже лота
# расчет с заданным кол-вом дней торгов и 0.1% уровнем риска

> МинЦена<-НижПрогноз*Лот
# создаем вектор с минимальными ценами лота с 0.1% уровнем риска
# вероятность резкого падения курса оценивается с 99.9% уровнем надежности
> МаксЦена<-ВерхПрогноз*Лот
# создаем вектор с максимальными ценами лота с 0.1% уровнем риска
# вероятность резкого роста курса оценивается с 99.9% уровнем надежности
>Цена_Лота250<-rep(Цена_Лота, 250)
# создаем вектор из 250 одинаковых цен лота по состоянию на 30.03. 2018 г.
>ДоляКредита<-ifelse(МинЦена/Цена_Лота250>=Цена_Лота250/МаксЦена, >Цена_Лота250/МаксЦена,МинЦена/Цена_Лота250)
# находим долю кредита для торговли на срок от 1,2.,3 … и до 250 дней
# ifelse -логическое выражение условия
```

```
# символ >= означает больше, либо равно
# если (МинЦена/Цена_Лота250>=Цена_Лота250/МаксЦена)
# то тогда ДоляКредита= Цена_Лота250/МаксЦена
# в противном случае ДоляКредита= Цена_Лота250/МаксЦена
# подробнее о выражении ifelse в R можно узнать по запросу ?ifelse

>ДоляКредита<-round((ДоляКредита-0.005), 2)
```

Конец ознакомительного фрагмента.