МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«САМАРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
УНИВЕРСИТЕТ ИМЕНИ АКАДЕМИКА С.П. КОРОЛЕВА»
(САМАРСКИЙ УНИВЕРСИТЕТ)

ЭКОНОМЕТРИЧЕСКОЕ МОДЕЛИРОВАНИЕ И ПРОГНОЗИРОВАНИЕ ВРЕМЕННЫХ РЯДОВ СРЕДСТВАМИ ЯЗЫКА R

Рекомендовано редакционно-издательским советом федерального государственного автономного образовательного учреждения высшего образования «Самарский национальный исследовательский университет имени академика С.П. Королева» в качестве методических указаний для обучающихся Самарского университета по основной образовательной программе высшего образования по направлению подготовки 38.03.05 Бизнес-информатика

Составители: *А.А. Коробецкая,*
В.К. Семенычев

С А М А Р А
Издательство Самарского университета
2022

УДК 338(075)+004.9(075)
ББК 65.290я7

Составители: ***А.А. Коробецкая, В.К. Семенычев***

Рецензент канд. экон. наук, доц. М. В. Ц а п е н к о

Эконометрическое моделирование и прогнозирование временных рядов средствами языка R: методические указания / составители: *А.А. Коробецкая, В.К. Семенычев.* – Самара: Издательство Самарского университета, 2022. – 32 с.

Методические указания содержат цикл лабораторных работ, направленных на изучение приемов работы в свободной среде R, в частности загрузки, анализа и моделирования временных рядов.

Предназначены для обучающихся по направлению подготовки 38.03.05 Бизнес-информатика и другим экономическим направлениям очной, очно-заочной и заочной форм обучения.

УДК 338(075)+004.9(075)
ББК 65.290я7

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

R – это универсальный язык программирования, разработанный для применения в таких областях, как разведочный анализ данных, классические статистические тесты и высокоуровневая графика. R обладает рядом существенных достоинств.

R – свободная программная среда вычислений с открытым исходным кодом. Это реализация языка S с дополнительными моделями, разработанными в языке S-Plus. В некоторых случаях моделями в обоих языках занимались одни и те же люди. R доступен в соответствии с лицензией GNU.

R уже имеет собственную обширную и непрерывно расширяющуюся библиотеку пакетов, содержащую большое количество готовых решений различных задач. Некоторые пакеты имеют ограниченную область применения, другие представляют целые области статистики, а некоторые отражают новейшие разработки. Многие новые разработки в области статистики, эконометрики, биологии, химии, медицины, географии и других прикладных областей сначала появляются как R-пакеты, и только потом реализуются в коммерческих программных продуктах.

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

Таким образом, R является полезным инструментом в области анализа больших массивов данных.

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

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

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

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

Для выполнения практикума читателю необходимо установить:

1. Язык R:

– классический https://cran.rstudio.com/bin/windows/base/

– или PRO https://mran.revolutionanalytics.com/download/

2. Графический интерфейс пользователя RStudio: https://www.rstudio.com/products/rstudio/download/ – набор интегрированных инструментов, разработанных, чтобы наиболее продуктивно использовать R.

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

При сохранении *.R файлов не следует использовать названия файлов или папок, содержащих русские буквы, или пробелы во избежание ошибок. Для работы в R потребуется стабильное Интернет-соединение, т.к. может потребоваться онлайн загрузка дополнительных пакетов для обработки данных.

1. ОСНОВЫ РАБОТЫ С R. ОБРАБОТКА СТАТИСТИЧЕСКИХ ДАННЫХ

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

Задание:

1. Загрузить данные для своего варианта в переменную-вектор.

2. Получить справочную информацию по своим данным, просмотреть их содержимое.

3. Проверить, есть ли среди данных пропуски.

4. Создать новую переменную-вектор, в которой будут: 1, если значение в исходном векторе больше среднего; –1, если значение переменной меньше среднего; и 0, если значение равно среднему.

5. Вывести описательную статистику.

6. Построить графики абсолютных частот и плотности распределения.

Порядок выполнения работы:

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

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

Можно использовать R как калькулятор, например:

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

Результат сразу появится в консоли.

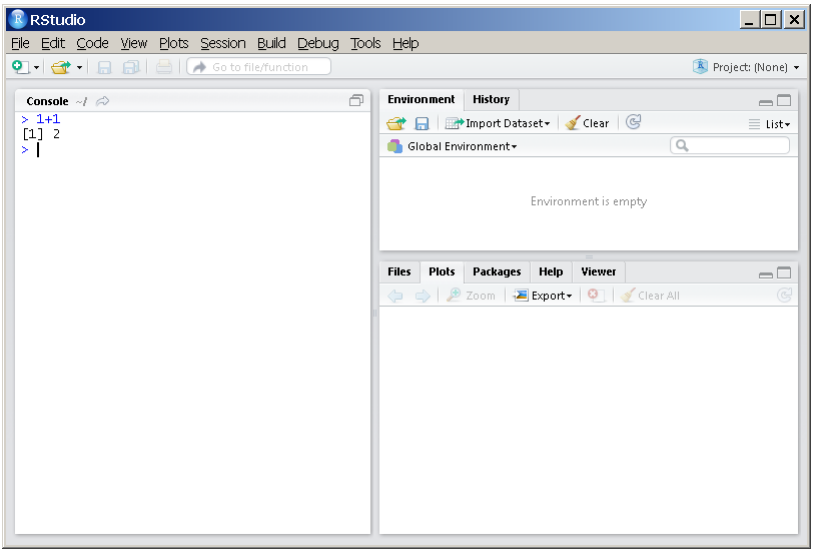

Рисунок 1. Консольный режим в RStudio

Строки печатаются в кавычках: двойных или одинарных:

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

Логические выражения возвращают TRUE или FALSE:

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

Чтобы сравнить два выражения, используется двойной знак равенства:

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

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

```
> x <- 42
```

Или в обратную сторону:

```
> 5 -> X
```

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

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

R чувствителен к регистру: переменные x и X – это разные переменные:

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

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

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

Получить помощь по функции можно командой `help(functionname)` или `?functionname`. В правом нижнем углу на вкладке Help появится справка (рисунок 2):

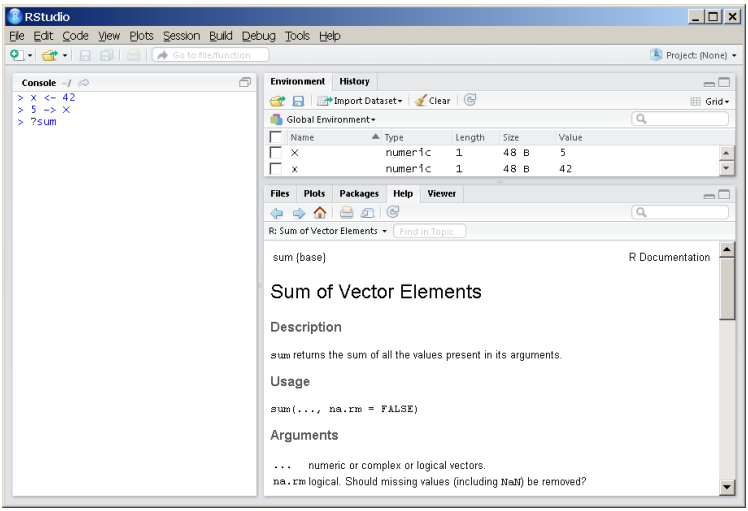

Рисунок 2. Справка по функции `sum`

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

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

NA – это пропущенное наблюдение (от англ. Not Available). Его не следует путать с NaN (Not a Number – «не число», неопределенность):

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

Попробуем просуммировать элементы вектора y:

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

Необязательным аргументом функции sum является na.rm (сокр. от англ. Remove NA), по умолчанию равный FALSE.

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

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

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

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

Обращаться к элементам вектора можно, используя квадратные скобки:

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

Либо можно задать элементам вектора имена:

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

```
 first second  third
     1      2      3
> ranks["first"]
first
    1
```

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

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

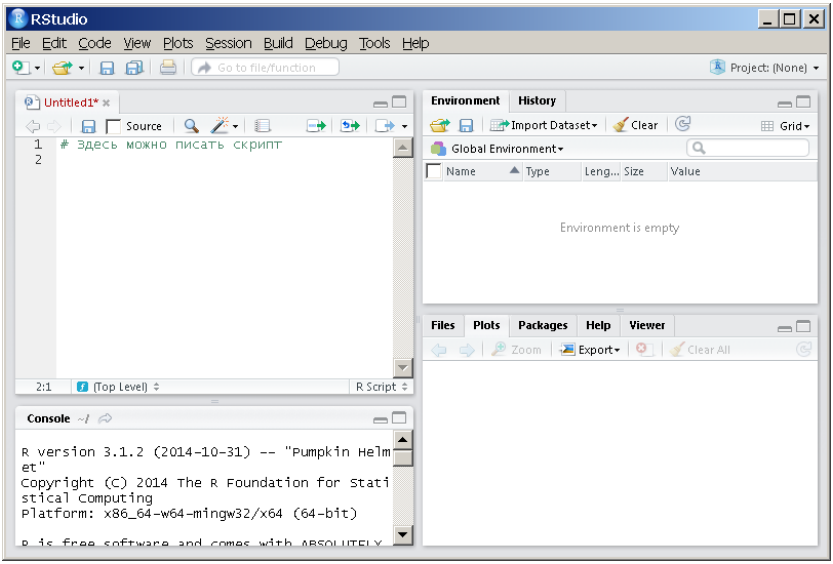

Рисунок 3. Создание нового скрипта

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

Чтобы выполнить команду в режиме скрипта, следует поставить курсор на нужную строку и нажать Ctrl+Enter. Если команда

занимает более одной строки, то необходимо ставить знак + в конце каждой строки. Команда выполняется построчно, в каждой строке требуется нажимать Ctrl+Enter, либо выделить всю команду целиком и нажать Ctrl+Enter один раз.

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

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

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

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

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

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

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

В консоли получим следующую таблицу:

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

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

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

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

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

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

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

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

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

В R существует базовый набор пакетов (библиотек), содержащих самые необходимые функции [3]. Для реализации задач эконометрики требуются дополнительные пакеты.

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

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

Для подключения установленного пакета следует воспользоваться функцией `library`.

Например, установим следующие пакеты:

`psych` – содержит функции для расчета описательных статистик;

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

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

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

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

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

Для выполнения данной работы понадобится подключить следующие пакеты:

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

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

```
help(cars)
```

Результат выполнения команды (в правом нижнем углу на вкладке Help) показан на рисунке 4. В этом наборе данных 50 наблюдений и две переменных (скорость, миль/час и длина тормозного пути в футах).

Поместим в переменную d встроенный в R набор данных по автомобилям[1]:

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

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

Теперь d имеет тип данных data.frame (набор данных), в чем можно удостовериться, посмотрев в правом вернем углу окна таблицу среды Environment (рисунок 5). Для этого должен быть выбран режим Grid.

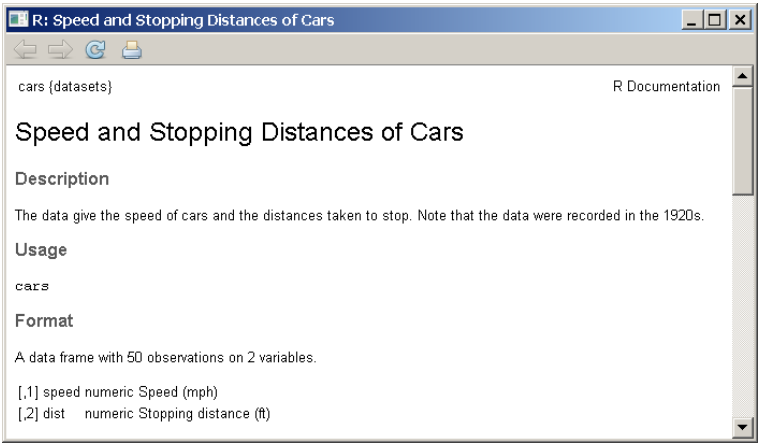

Рисунок 4. Справка по набору данных cars

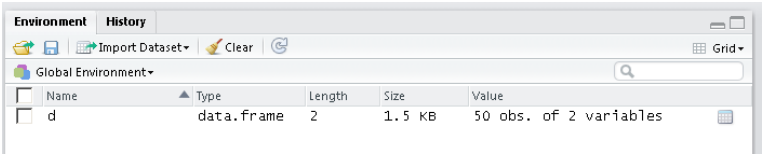

Рисунок 5. Описание набора данных d в таблице среды Environment

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

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

Результат выполнения команды появится в консоли:

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

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

Посмотрим на первые шесть наблюдений набора данных d:

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

и последние шесть наблюдений:

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

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

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

Построим гистограмму абсолютных частот для переменной dist (длины тормозного пути). Воспользуемся функцией qplot, задав источник данных d (аргумент data), переменную для построения графика (dist), подпишем оси (параметры функции xlab и ylab) и название графика (параметр main):

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

Результат выполнения данной функции показан на рисунке 6.

Рисунок 6. Гистограмма абсолютных частот для переменной `dist`

Можно построить так же гистограмму плотности распределения (рисунок 7):

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

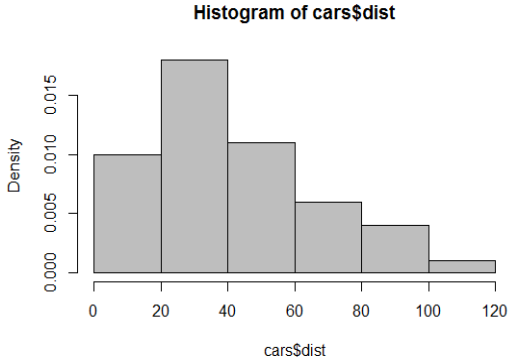

Рисунок 7. Гистограмма плотности распределения для переменной `dist`

2. ЛИНЕЙНАЯ РЕГРЕССИЯ

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

Задание:

1. Загрузить набор данных для своего варианта, ознакомиться с его содержимым.

2. Построить график корреляционного поля для каждого фактора.

3. Построить уравнение парной линейной регрессии для каждого фактора.

4. Проверить значимость каждого из полученных уравнений регрессии. Показать уравнения регрессии с заданным в варианте доверительным интервалом на графиках.

5. Построить прогнозы по каждому из уравнений парной регрессии для заданных в варианте значений факторов.

6. Построить уравнение множественной линейной регрессии и получить корреляционную матрицу.

7. Построить прогноз по уравнению множественной регрессии для заданных в варианте значений факторов.

Порядок выполнения работы:

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

```
d <- cars
```

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

Построим график зависимости длины тормозного пути от скорости автомобиля (рисунок 8):

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

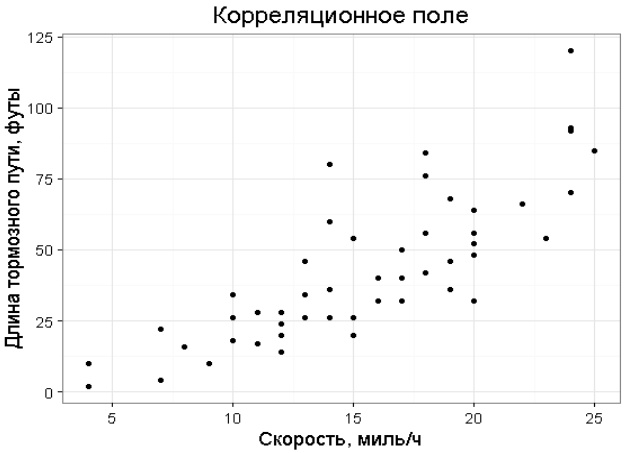

Рисунок 8. График зависимости длины тормозного пути `dist` от скорости автомобиля `speed`

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

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

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

Тип `lm` – это список из 12 элементов (рисунок 9).

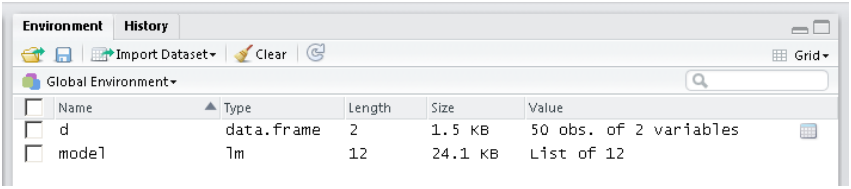

Рисунок 9. Переменная-список `lm` в таблице среды `Environment`

Посмотрим на коэффициенты уравнения линейной регрессии:

```
model$coefficients
```

Результат в консоли:

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

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

Таким образом, уравнение регрессии имеет вид:

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

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

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

Более полный набор расчетов по модели можно получить командой summary:

```
summary(model) # базовый пакет base

Call:
lm(formula = dist ~ speed, data = d)

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

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -17.5791     6.7584  -2.601   0.0123 *
speed         3.9324     0.4155   9.464 1.49e-12 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

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

Помимо коэффициентов регрессии, R выводит:

– стандартные ошибки коэффициентов (`Std. Error`);

– наблюдаемые значения t-критерия при проверке значимости коэффициентов регрессии (`t value`);

– P-значения для коэффициентов регрессии (`P-value`).

Звездочками или точками в столбце справа R показывает значимость или незначимость коэффициентов: *** – значимы на уровне значимости менее 0.001; ** – значимы на уровне значимости 0.001; * – значимы на уровне значимости 0.01; . – значимы на уровне значимости 0.05 и т.д. Эти обозначения приведены в разделе `Signif.codes`. Коэффициент детерминации (`Multiple R-squared`) равен 0.6511; скорректированный коэффициент детерминации (`Adjusted R-squared`) равен 0.6438. Наблюдаемое значения F-критерия проверки значимости уравнения в целом и P-значение:

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

Таким образом, уравнение регрессии получилось значимым.

Проведем на графике полученную линию регрессии с 95% доверительными интервалами (рисунок 10):

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

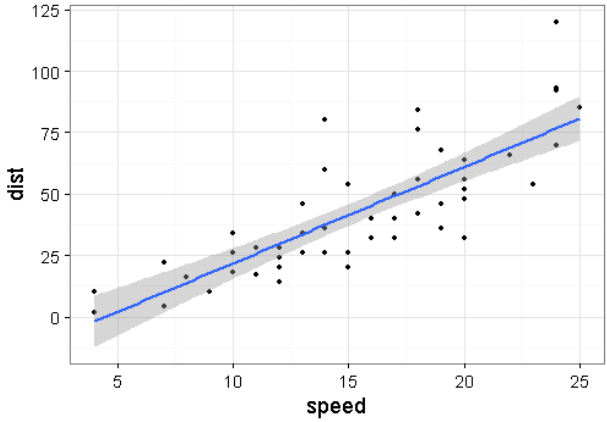

Рисунок 10. График линейной регрессии с доверительными интервалами

Рассчитаем также 95% доверительные интервалы для параметров линейной регрессии:

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

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

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

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

Необъясненная сумма квадратов отклонений:

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

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

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

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

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

Строим прогноз функцией `predict`:

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

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

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

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

`Fertility` – рождаемость;

`Agriculture` – % мужчин, занятых в сельском хозяйстве;

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

`Education` – % призывников, имеющих образование помимо начального;

`Catholic` – % католиков среди населения;

`Infant.Mortality` – % детей, умерших до года.

Посмотрим на этот набор данных:

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

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

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

Результатом будет график, показанный на рисунке 11.

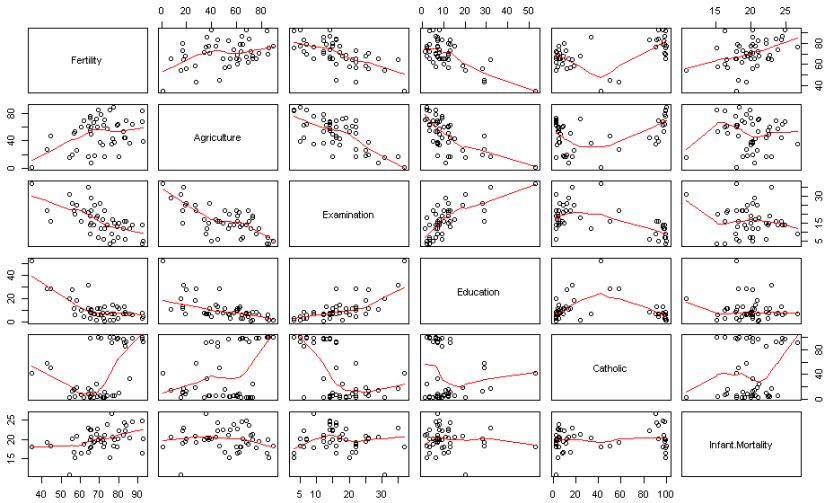

Рисунок 11. Диаграммы рассеяния, полученные с помощью функции `pairs`

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

```
> cor(swiss)
                  Fertility Agriculture Examination Education Catholic
Fertility             1.000      0.3531      -0.646   -0.6638    0.464
Agriculture           0.353      1.0000      -0.687   -0.6395    0.401
Examination          -0.646     -0.6865       1.000    0.6984   -0.573
Education            -0.664     -0.6395       0.698    1.0000   -0.154
Catholic              0.464      0.4011      -0.573   -0.1539    1.000
Infant.Mortality      0.417     -0.0609      -0.114   -0.0993    0.175
                  Infant.Mortality
Fertility                   0.4166
Agriculture                -0.0609
Examination                -0.1140
Education                  -0.0993
Catholic                    0.1755
Infant.Mortality            1.0000
```

Существует еще одна функция, позволяющая получить корреляционную матрицу, диаграммы рассеяния и сглаженные распределения одновременно (рисунок 12):

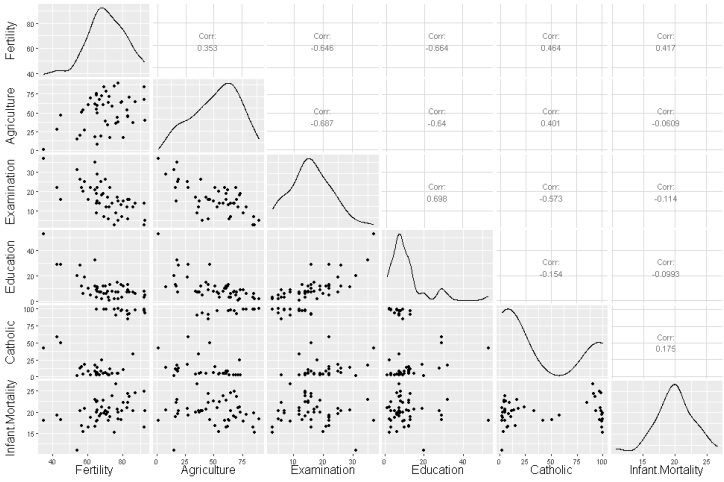

Рисунок 12. Корреляционная матрица, диаграммы рассеяния и сглаженные распределения, полученные с помощью функции ggpairs

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

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

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

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

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

```
> summary(model2)

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

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

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 86.22502    4.73472  18.211  < 2e-16 ***
Agriculture -0.20304    0.07115  -2.854  0.00662 **
Education   -1.07215    0.15580  -6.881 1.91e-08 ***
Catholic     0.14520    0.03015   4.817 1.84e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 7.728 on 43 degrees of freedom
Multiple R-squared:  0.6423, Adjusted R-squared:  0.6173
F-statistic: 25.73 on 3 and 43 DF,  p-value: 1.089e-09
```

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

```
nd2 <- data.frame(Agriculture=0.5,Catholic=0.5, Education=20)
```

```
> predict(model2,nd2)
       1
64.75316
```

Построение прогноза по нескольким точкам выполняется с помощью векторов значений:

```
nd2 <- data.frame(Agriculture=c(0.5,0.8),Catholic=c(0.5, 0.65),
                  Education=c(20, 25))
> predict(model2,nd2)
       1        2
64.75316 59.35330
```

3. ВРЕМЕННЫЕ РЯДЫ

Цель работы: Изучить базовые способы анализа, моделирования и прогнозирования временных рядов.

Задание:

1. Загрузить данные из временного ряда для своего варианта за последние полгода.

2. Построить график временного ряда с графиками автокорреляционной и частной автокорреляционной функций.

3. Подобрать вручную порядок ARMA-модели. Для подбора использовать визуальный анализ графика.

4. Подобрать ARMA-модель автоматически.

5. Построить прогноз на указанное в варианте число шагов. Показать прогноз на графике.

Порядок выполнения работы:

Для работы подключим следующие пакеты:

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

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

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

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

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

Можно построить полученный ряд на графике (рисунок 13):

```
plot(y)
```

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

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

Все три графика одновременно могут быть вызваны командой:

```
tsdisplay(y)
```

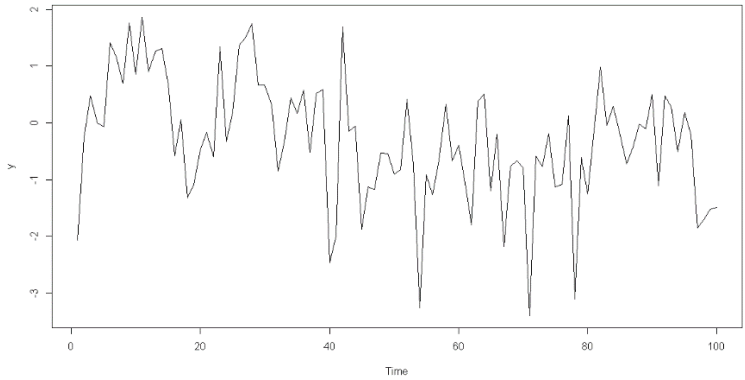

Рисунок 13. График временного ряда

Результат выполнения команды показан на рисунке 14.

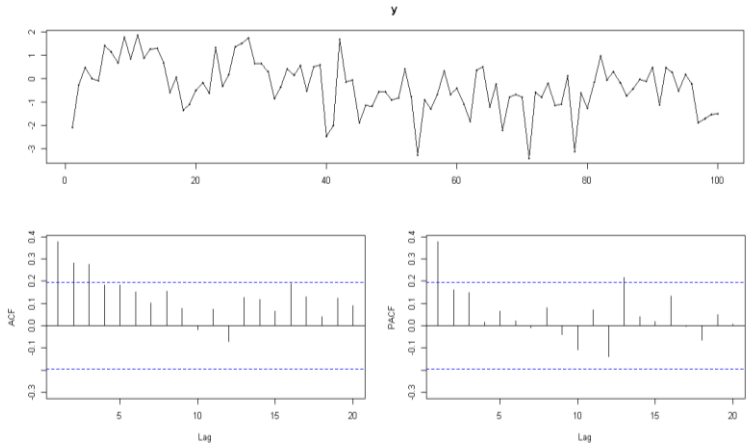

Рисунок 14. График временного ряда с графиками автокорреляционной и частной автокорреляционной функций

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

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

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

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

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

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

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

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

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

Белый шум $y_t = \varepsilon_t$:

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

Попробуем подобрать различные ARIMA-модели под реальные данные. Для примера выберем временной ряд, содержащий 98 ежегодных наблюдений за уровнями воды в озере Гурон с 1875 по 1972 г. Оценим ARMA(2,1) модель с помощью функции `Arima` из пакета `forecast`:

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

Посмотрим на результат оценивания:

```
summary(mod)
Series: y
ARIMA(2,0,1) with non-zero mean
Coefficients:
         ar1      ar2     ma1  intercept
0.7829  -0.0342  0.2857   579.0533
s.e.  0.3262   0.2845  0.3144     0.3467
sigma^2 estimated as 0.4749:  log likelihood=-103.24
AIC=216.48   AICc=217.13   BIC=229.4
Training set error measures:
                     ME     RMSE       MAE          MPE       MAPE      MASE
Training set -0.00863855 0.689106 0.5492008 -0.001635858 0.09486097 0.9378958
                     ACF1
Training set 0.002260998
```

Получили модель $y_t = 579.0533 + 0.7829y_{t-1} - 0.0342y_{t-2} + + \varepsilon_t + 0.2857\varepsilon_{t-1}$. В строке `s.e.` перечислены стандартные ошибки коэффициентов. Оценка дисперсии $\hat{\sigma}^2 = 0.4749$. Строкой ниже перечислены значения критерия Акаике (`AIC=216.48`), скорректированного критерия Акаике (`AICc=217.13`) и критерия Шварца (`BIC=229.4`).

Командой `fitted(mod)` можно получить модельные значения временного ряда. Построим реальные и модельные значения на одном графике (рисунок 15):

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

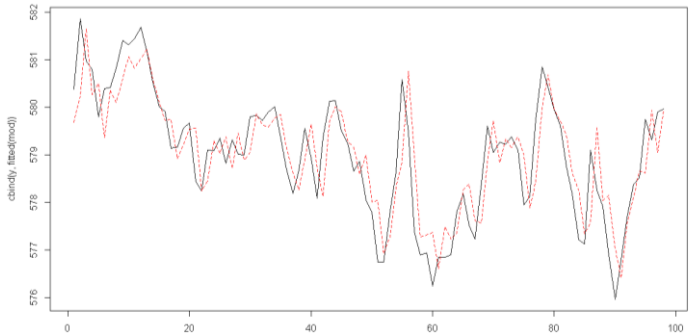

Рисунок 15. График модельных и реальных значений временного ряда

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

Построим прогноз на 5 шагов вперед:

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

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

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

Строим график прогноза (рисунок 16):

```
plot(prognoz)
```

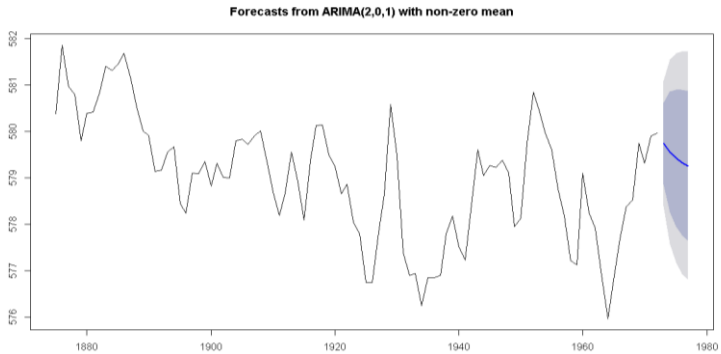

Рисунок 16. График прогноза мо модели ARIMA

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

```
mod_a<- auto.arima(y)
summary(mod_a)
Series: y
ARIMA(0,1,4) with drift
```

```
Coefficients:
         ma1      ma2      ma3      ma4    drift
0.0584  -0.3158  -0.3035  -0.2349  -0.0205
s.e.  0.1031   0.1058   0.1100   0.1299   0.0167
sigma^2 estimated as 0.4806:  log likelihood=-101.09
AIC=214.18   AICc=215.12   BIC=229.63
Training set error measures:
                      ME      RMSE       MAE           MPE      MAPE
MASE
Training set 0.002898915 0.6886758 0.5441565 0.0003830464 0.093983
0.9292814
                    ACF1
Training set 0.01081339
```

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

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

Рекомендуемый библиографический список

1. Айвазян, С.А. Прикладная статистика и основы эконометрики / С.А. Айвазян, В.С. Мхитарян. – Москва: ЮНИТИ, 1998. – 650 с.

2. Зорин, А.В. Введение в прикладной статистический анализ в пакете R: учебно-методическое пособие / А.В. Зорин, М.А. Федоткин. – Нижний Новгород: ННГУ, 2010. – 50 с.

3. Мастицкий, С.Э. Статистический анализ и визуализация данных с помощью R / С.Э. Мастицкий, В.К. Шитиков. – Москва: ДМК Пресс, 2015. – 496 с.

4. Семенычев, В.К. Предложения эконометрического инструментария моделирования и прогнозирования эволюционных процессов / В.К. Семенычев, А.А. Коробецкая, В.Н. Кожухова. – Самара: САГМУ, 2015. – 384 с.

5. Coghlan, A. A Little Book of R for Time Series / A. Coghlan. – 2015. – URL: https://media.readthedocs.org/pdf/a-little-book-of-r-for-time-series/latest/a-little-book-of-r-for-time-series.pdf.

6. ggplot2 Help. – URL: http://docs.ggplot2.org/current/index.html.

7. Package ‘GA’ // The Comprehensive R Archive Network. – URL: https://cran.r-project.org/web/packages/GA/GA.pdf.

8. Peng, R.D. R Programming for Data Science / R.D. Peng. – URL: https://leanpub.com/rprogramming/.

Методические материалы

ЭКОНОМЕТРИЧЕСКОЕ МОДЕЛИРОВАНИЕ И ПРОГНОЗИРОВАНИЕ ВРЕМЕННЫХ РЯДОВ СРЕДСТВАМИ ЯЗЫКА R

Методические указания

Составители:
Коробецкая Анастасия Александровна,
Семенычев Валерий Константинович

Редакционно-издательская обработка А.С. Никитиной

Подписано в печать 11.10.2022. Формат 60×84 1/16.
Бумага офсетная. Печ. л. 2,0.
Тираж 25 экз. Заказ . Арт. 4(Р2МУ)/2022.

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«САМАРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
УНИВЕРСИТЕТ ИМЕНИ АКАДЕМИКА С.П. КОРОЛЕВА»
(САМАРСКИЙ УНИВЕРСИТЕТ)
443086, САМАРА, МОСКОВСКОЕ ШОССЕ, 34.

Издательство Самарского университета.
443086, Самара, Московское шоссе, 34.