МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧЕРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «СЕВАСТОПОЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

кафедра «Информационные системы»

Отчёт

По лабораторной работе №1

По дисциплине «Интеллектуальный анализ данных»

Выполонил:

Ст.гр. ИС/б-20-1-о

Хроменко Д.А.

Принял:

Сырых О.А.

Севастополь

2023 г.

Лабораторная работа №1

«Исследование возможностей языка R для статистического анализа данных»

1.1 Цель работы:

Изучить основные особенности языка R. Исследовать возможности языка R для работы с графикой.

1.2 Постановка задачи

Требуется установить R на ПК. Исследовать команду *'demo()*', полученные результаты вставить в отчет. Исследовать основные функции и команды языка R, представленные в данной лабораторной работе, полученные результаты вставить в отчет. Ответить на контрольные вопросы.

1.3 Ход работы

1.3.1 Был установлен “R” и запущен на персональном компьютере.

1.3.2 Была исследована команда “*demo()*”. *demo()* - это удобный интерфейс для запуска некоторых демонстрационных R-скриптов. *demo()* выдает список доступных аргументов функции (рисунок 1). Например, пример выполнения функции *demo(graphics)* представлен на рисунках 2, 3. На рисунке 4 представлен пример выполнения функции *demo(nlm)*.

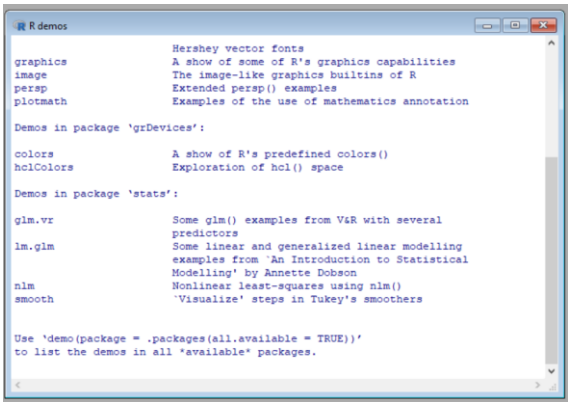

Рисунок 1 – Результат выполнения команды demo()

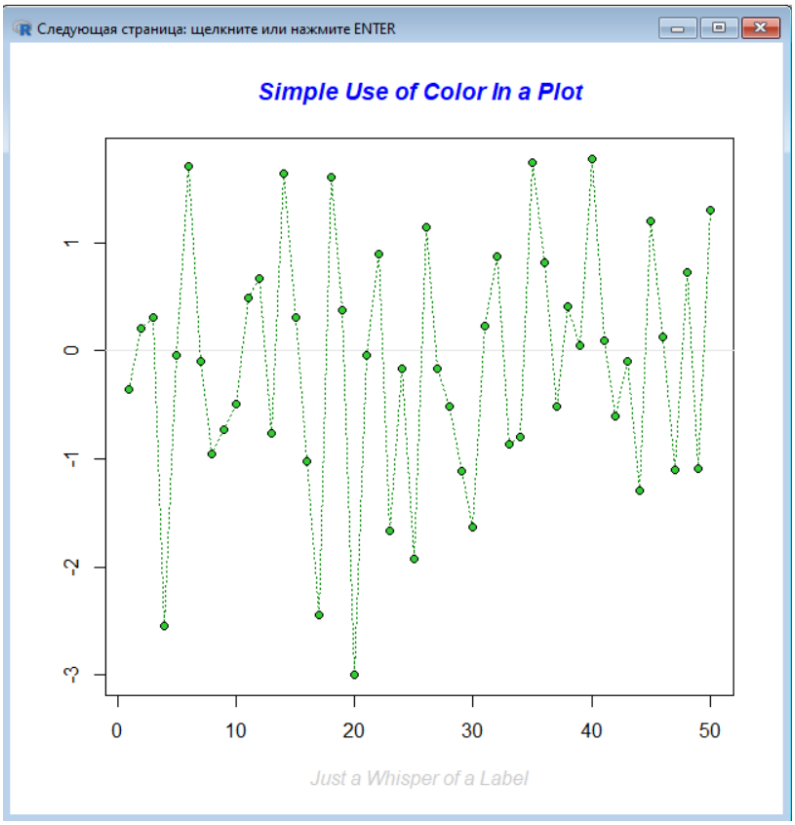

Рисунок 2 – Результат выполнения функции demo(graphics)

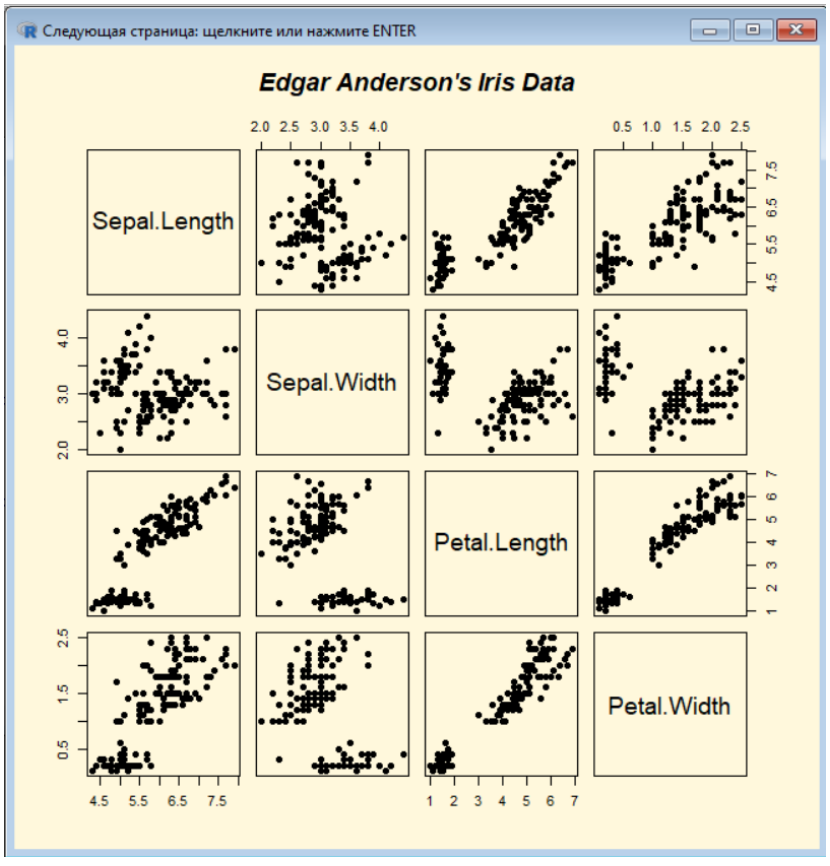

Рисунок 3 – Результат выполнения функции demo(graphics)

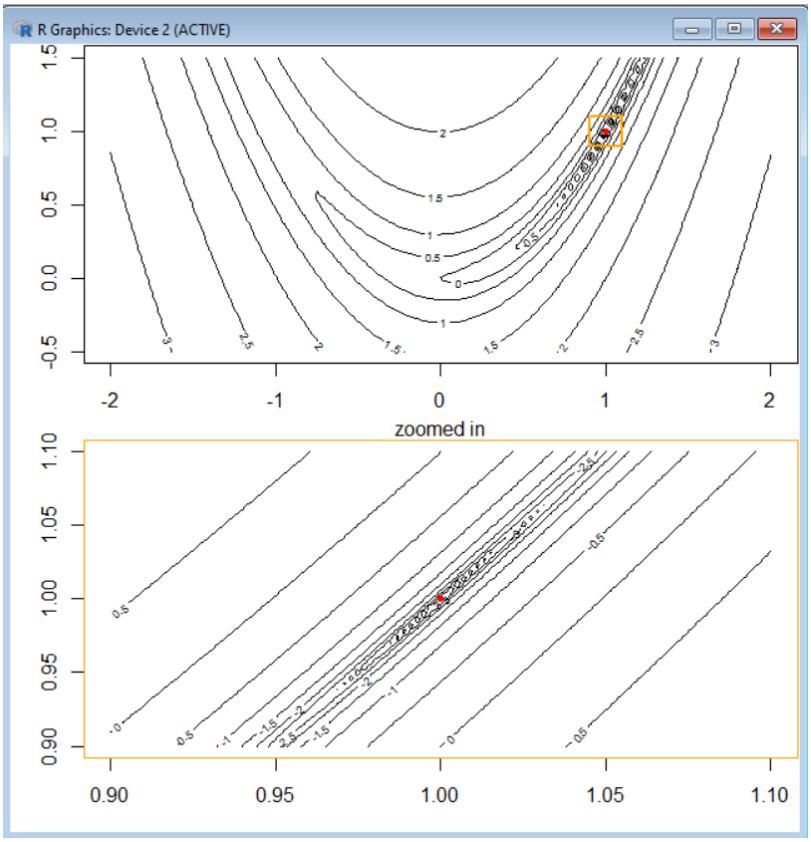

Рисунок 4 – Результат выполнения функции demo(nlm)

1.3.3 Были исследованы основные функции и команды языка R.

С функцией *demo()*, мы уже ознакомились, она служит для запуска демонстрационных программ. Существуют команды *help()* и *help.start()* для получения справки.

Для выхода из программы используется команда q().

Для получения информации о конкретной функции можно использовать конструкцию *help(<имя функции>)* или *?(<имя функции>)*. Пример выполнения функции help с аргументом q представлен на рисунке 5.

Кроме функции *help()*, полезной, если неизвестно точное названия функции, является команда *help.search(<что искать>)*. Например команда *help.search("vector")* в качестве результата будет выдавать список команд, свойственных «векторам», с кратким описанием (Рисунок 6).

Команда *apropos()* выдаст просто список команд, содержащих строку, которая была в кавычках. Например, результат выполнения команды *apropos(“vector”)* представлен на рисунке 7.

quit {base} R Documentation

Terminate an R Session

Description

The function `quit` or its alias `q` terminate the current R session.

Usage

```
quit(save = "default", status = 0, runLast = TRUE)
   q(save = "default", status = 0, runLast = TRUE)
```

Arguments

`save`
: a character string indicating whether the environment (workspace) should be saved, one of `"no"`, `"yes"`, `"ask"` or `"default"`.

`status`
: the (numerical) error status to be returned to the operating system, where relevant. Conventionally `0` indicates successful completion.

`runLast`
: should `.Last()` be executed?

Details

`save` must be one of `"no"`, `"yes"`, `"ask"` or `"default"`. In the first case the workspace is not saved, in the second

Рисунок 5 – Результат выполнения функции *help(q)*

Search Results

Примеры работы кода:

graphics::Hershey	Tables of the characters in the Hershey vector fonts	(Run demo)
graphics::Japanese	Tables of the Japanese characters in the Hershey vector fonts	(Run demo)

Страницы помощи:

base::append	Vector Merging
base::as.data.frame	Coerce to a Data Frame
base::.bincode	Bin a Numeric Vector
base::c	Combine Values into a Vector or List
base::character	Character Vectors
base::double	Double-Precision Vectors
base::eigen	Spectral Decomposition of a Matrix
base::encodeString	Encode Character Vector as for Printing
base::Encoding	Read or Set the Declared Encodings for a Character Vector
base::dir.create	Manipulation of Directories and File Permissions
base::gctorture	Torture Garbage Collector
base::getwd	Get or Set Working Directory
base::grepRaw	Pattern Matching for Raw Vectors
base::iconv	Convert Character Vector between Encodings
base::integer	Integer Vectors

Рисунок 6 – Результат выполнения функции *help.search("vector")*

```
> apropos("vector")
[1] "as.data.frame.vector" "as.vector"            "as.vector.data.frame"
[4] "as.vector.factor"     "as.vector.POSIXlt"    "is.vector"
[7] "vector"               "Vectorize"
```

Рисунок 7 – Результат выполнения функции *apropos("vector")*

С помощью команды *install.packages()* можно установить пакеты, скачанные с официального онлайн-репозитория – CRAN (http://cran.r-project.org/).

Также в RGui есть возможность установить пакеты в меню пакеты (Рисунок 8). Как только пакет установлен, то он сразу готов к работе. Нужно только инициализировать его перед употреблением. Для этого служит команда *library()*.

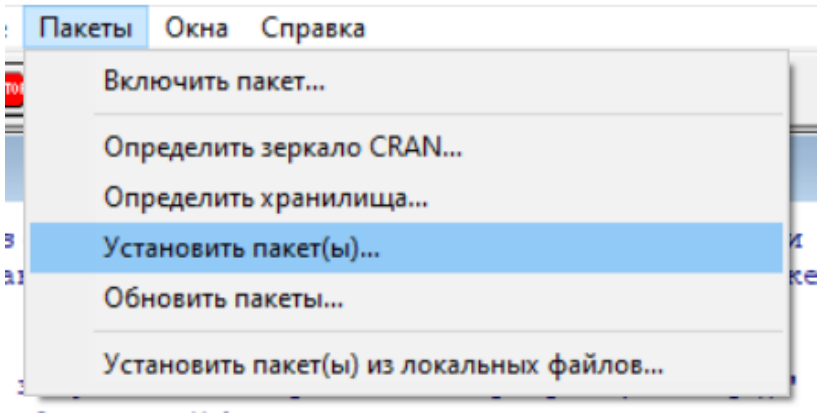

Рисунок 8 – Возможность установить пакеты в RGui

R – регистрозависимый язык, для присваивания используется символы: «->», «<-« и «=». Аргументы функций передаются в круглых скобках через запятую. В R можно выполнять математические операции (Рисунок 9).

```
> 3/7
[1] 0.4285714
> 3/7-0.4285714
[1] 2.857143e-08
> sqrt(2)*sqrt(2)
[1] 2
> (sqrt(2)*sqrt(2))-2
[1] 4.440892e-16
```

Рисунок 9 – Пример выполнения математических операций в R

Существует команда *round()* (округлить). Она имеет два аргумента: число, которое нужно округлить, и значение *digits*, сообщающее, до какого знака округлять. Некоторые аргументы могут иметь имена, благодаря чему их можно перечислять не по порядку, а по имени. Имена можно сокращать вплоть до одной буквы, но только если нет других аргументов, которые от такого сокращения станут неразличимы. При перечислении аргументов по порядку имена можно опускать. Для однострочных комментариев используется символ #. Примеры работы с командой *round()*, представлены на рисунке 10.

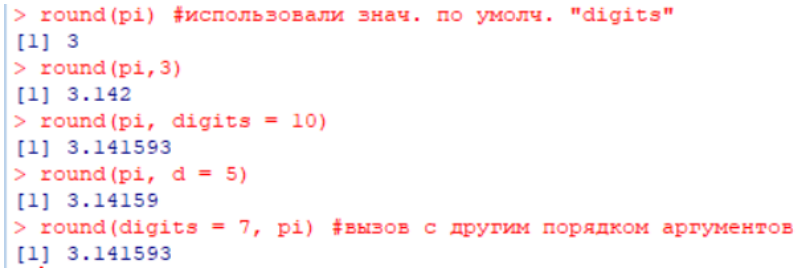

Рисунок 10 – Примеры использования команды *round()*

Команды разделяются точкой с запятой «;» или символом перевода на новую строку. Строковые и числовые значения можно сохранять. На рисунке 11 показан пример использования *";"* и сохранения строковых и числовых значений.

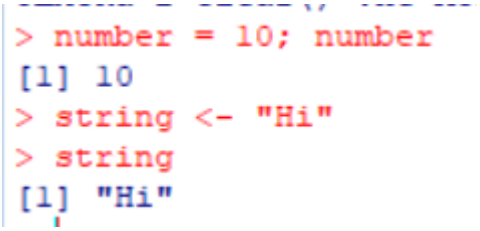

Рисунок 11 – Сохранение строковых и числовых значений

Также можно создавать и сохранять векторы. Вектор создается с помощью функции *c()*, которая объединяет несколько однотипных элементов. Также, с помощью двоеточия *«:»* или функции *seq()* можно создать регулярную последовательность. Функция *rep()* позволяет повторять некоторый образец. На рисунке 12 продемонстрированы примеры работы с векторами.

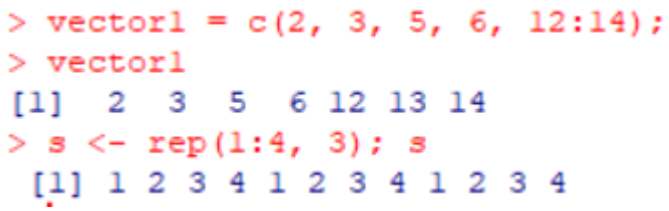

Рисунок 12 – Примеры работы с векторами

Существуют команды *length()* для получения длины вектора, *mean()* для получения среднего значения элементов вектора, *var()* для получения дисперсии элементов вектора. Результат работы этих функций продемонстрирован на рисунке 13.

```
> length(vector1)
[1] 7
> mean(vector1)
[1] 7.857143
> var(vector1)
[1] 25.14286
```

Рисунок 13 – Использование специальных функций для работы с вектором

Матрицы создаются с помощью команды *matrix()*. Пример создания матрицы продемонстрирован на рисунке 14.

```
> matrix(nrow = 8, ncol = 9, data = vector1)
     [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9]
[1,]    2    3    5    6   12   13   14    2    3
[2,]    3    5    6   12   13   14    2    3    5
[3,]    5    6   12   13   14    2    3    5    6
[4,]    6   12   13   14    2    3    5    6   12
[5,]   12   13   14    2    3    5    6   12   13
[6,]   13   14    2    3    5    6   12   13   14
[7,]   14    2    3    5    6   12   13   14    2
[8,]    2    3    5    6   12   13   14    2    3
```

Рисунок 14 – Пример создания матрицы

Основной функцией для рисования объектов в R является функция *plot(x, y, ...)*, где x – координаты точек графика, либо некоторая графическая структура, функция или объект, содержащий методы рисования. *y*-координаты точек графика. У функции *plot()* существует параметр *type*, который изменяет вид точек на графике. Параметры *xlab* и *ylab* задают название осей абсцисс и ординат, соответственно. Параметр *main* задаёт заголовок графика. Пример создания графика с помощью команды *plot(sin, -pi, 2*pi, type = "s", main = "Пример графика")* представлен на рисунке 15. Для наглядности были созданы ещё 3 графика и представлены на рисунках 16-18.

Рисунок 15 – Пример создания графика с помощью команды *plot()*

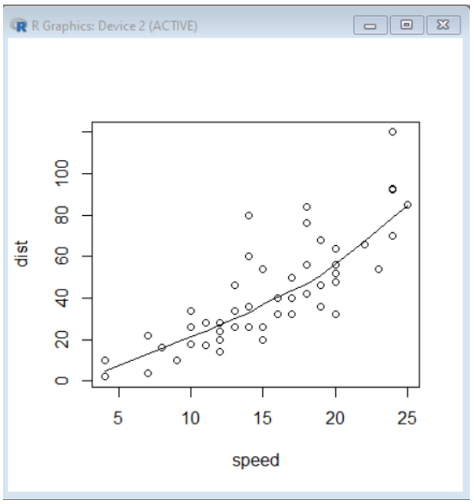

Рисунок 16 – График, созданный с помощью команды: *require(stats) # for lowess, rpois, rnorm; plot(cars); lines(lowess(cars))*

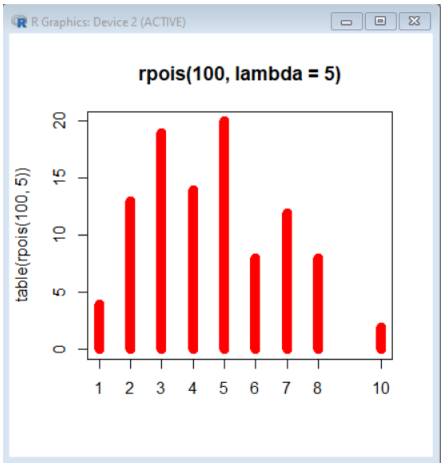

Рисунок 17 – График, созданный с помощью команды: *plot(table(rpois(100, 5)), type = "h", col = "red", lwd = 10, main = "rpois(100, lambda = 5)")*

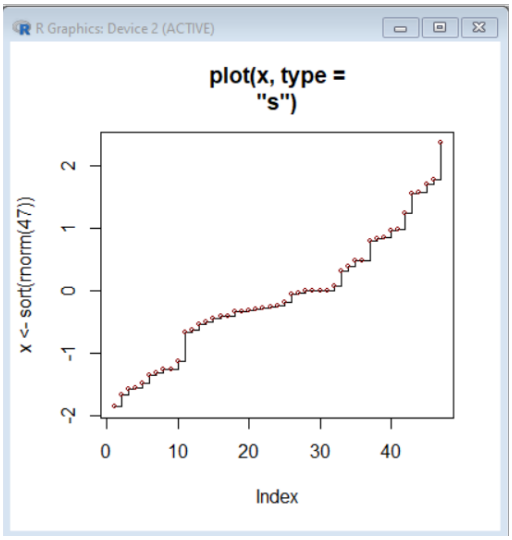

Рисунок 18 – График, созданный с помощью команды: *plot(x <- sort(rnorm(47)), type = "s", main = "plot(x, type = \"s\")"); points(x, cex = .5, col = "dark red")*

Выводы

В ходе выполнения данной лабораторной работы были изучены основные особенности языка R. Изучены основные команды и функции языка R, такие как создание графиков, матриц, векторов, вызов справки и другие вспомогательные функции. Полученные знания помогут в дальнейшем при исследованиях статистического анализа данных.

ОТВЕТЫ НА КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Особенности языка R.

R – статистическая система анализа, созданная Россом Ихакой и Робертом Гентлеманом. R является и языком, и программным обеспечением. Его особенности:

- эффективная обработка данных и простые средства для сохранения результатов;
- набор операторов для обработки массивов, матриц, и других сложных конструкций;
- большая, последовательная, интегрированная коллекция инструментальных средств для проведения статистического анализа,
- многочисленные графические средства;
- простой и эффективный язык программирования, который включает много возможностей.

2. Команда для получения подробной информации о функции в R.

- Команда 'license()' служит для получения более подробной информации.
- Команда 'contributors()' служит для получения дополнительной информации и 'citation()' для ознакомления с правилами упоминания R и его пакетов в публикациях.
- Команда 'demo()' для запуска демонстрационных программ, 'help()' – для получения справки, 'help.start()' - для доступа к справке через браузер.

3. Структура и особенности команды round() в R.

Функция round() в R используется для округления числовых значений до заданного количества знаков после запятой. Она имеет следующую структуру:

round(x, digits = 0)

Здесь x - входное значение, которое нужно округлить, а digits - количество знаков после запятой, до которых нужно округлить x. По умолчанию digits равен 0, то есть округление происходит до целого числа.

Кроме того, round() поддерживает дополнительные аргументы, такие как na.rm, который указывает, нужно ли удалять пропущенные значения из входного вектора x перед округлением, и type, который задает тип возвращаемого значения (целое или с плавающей точкой).

Функция round() также имеет ряд особенностей:

- Если digits меньше нуля, то округление происходит до десятичных, сотых, тысячных и т.д. разрядов слева от запятой.
- Если digits больше нуля, то округление происходит до десятичных, сотых, тысячных и т.д. разрядов справа от запятой.
- Если digits равен нулю, то округление происходит до целого числа.
- Если x равен NaN (Not a Number) или Inf (бесконечность), то round() возвращает тот же самый результат.
- Если x равен -Inf (минус бесконечность), то round() возвращает -Inf для нечетных digits и Inf для четных digits.
- Если x равен Inf (бесконечность), то round() возвращает Inf для нечетных digits и -Inf для четных digits.

4. Команды для работы с векторами в R (изучить команды, не представленные в методических указаниях).

- *sum()* – возвращает сумму элементов вектора.
- *median()* – возвращает медиану элементов вектора
- *sd()* – возвращает стандартное отклонение элементов вектора
- *max()* – возвращает максимальное значение элементов вектора
- *min()* – возвращает минимальное значение элементов вектора.
- *sort()* – сортирует элементы вектора в порядке возрастания.
- *rev()* – меняет порядок элементов вектора на обратный.
- *sample()* – возвращает случайные элементы из вектора.

- *which()* – возвращает индексы элементов вектора, удовлетворяющих заданному условию.
- *unique()* – возвращает уникальные элементы вектора.
- *na.omit()* – удаляет все элементы вектора, содержащие пропущенные значения (NA).

5. Команды для работы с матрицами в R (изучить команды, не представленные в методических указаниях).

Создание матрицы:

- *matrix(data, nrow, ncol, byrow)* – создает матрицу с указанным количеством строк и столбцов, заполненную данными из вектора *data*. Если параметр *byrow* установлен в *TRUE*, то элементы вектора *data* заполняются построчно, в противном случае - по столбцам.
- *rbind(mat1, mat2)* – объединяет две матрицы *mat1* и *mat2* по строкам.
- *cbind(mat1, mat2)* – объединяет две матрицы *mat1* и *mat2* по столбцам.

Индексация матрицы:

- *матрица[row, col]* – доступ к элементу матрицы по номеру строки и столбца.
- *матрица[row,]* – доступ к строке матрицы по номеру строки.
- *матрица[, col]* – доступ к столбцу матрицы по номеру столбца.

Операции над матрицами:

- *t(mat)* – транспонирование матрицы.
- *diag(mat)* – возвращает диагональ матрицы.
- *det(mat)* – возвращает определитель матрицы.
- *solve(mat)* – возвращает обратную матрицу.
- *crossprod(mat1, mat2)* – возвращает результат перемножения транспонированной *mat1* на *mat2*.
- *tcrossprod(mat1, mat2)* – возвращает результат перемножения *mat1* на транспонированную *mat2*.

Функции, применяемые к матрицам:

- *dim(mat)* – возвращает размерность матрицы.
- *rowSums(mat)* – возвращает сумму элементов в каждой строке матрицы.
- *colSums(mat)* – возвращает сумму элементов в каждом столбце матрицы.
- *rowMeans(mat)* – возвращает среднее значение элементов в каждой строке матрицы.
- *colMeans(mat)* – возвращает среднее значение элементов в каждом столбце матрицы.
- *apply(mat, margin, fun)* – применяет функцию *fun* к элементам матрицы *mat* вдоль определенного измерения (строки или столбцы). Если *margin=1*, функция применяется к каждой строке матрицы, если *margin=2* – к каждому столбцу матрицы.
- *eigen(mat)* – возвращает собственные значения и собственные векторы матрицы.
- *svd(mat)* – возвращает сингулярное разложение матрицы.

6. Работа с графикой в R (изучить команды, не представленные в методических указаниях).

Функция *hist()*: позволяет построить гистограмму на основе вектора данных.

```
# Создание вектора данных
x <- rnorm(100)
# Построение гистограммы
hist(x)
```

Библиотека ggplot2: предоставляет возможность построения более сложных графиков с помощью "грамматики графиков". Она предлагает более гибкий подход к построению графиков, чем стандартные функции R.

```
library(ggplot2)
# Создание фрейма данных
data <- data.frame(
  x = c(1, 2, 3, 4, 5),
  y = c(1, 4, 9, 16, 25)
```

```
)

# Построение графика

ggplot(data, aes(x, y)) +

  geom_line()
```