Лабораторная работа №2.1

Корреляционный и регрессионный анализ данных. Создание набора данных

2.1.1 Цель работы

− исследовать возможности языка R для проведения корреляционного и регрессионного анализа данных;

− создание набора данных для проведения корреляционного и регрессионного анализа данных

2.1.2 Порядок выполнения работы

1. Ознакомится с методическими указаниями

2. Исследовать основные функции и команды языка R, представленные в данной лабораторной работе

3. Выполнить все примеры.

4. Выполнить ввод данных с клавиатуры

5. Провести экспорт данных из текстового файла с разделителями

6. Выполнить экспорт данных из Excel.

7. Подобрать экспериментальные данные для анализа (пример данных представлен в Приложении А)

2.1.3 Ход работы

Были проанализированы примеры, представленные в методических указаниях.

Для создания таблицы используется функция data.frame() (листинг 1).

Листинг 1 – Создание таблицы

mydata <- data.frame(col1, col2, col3,…)

В листинге 1 col1, col2, col3,… это векторы любого типа (текстового, числового или логического), которые станут столбцами таблицы. Названия каждому столбцу можно присвоить при помощи функции names().

Пример 1 представлен в листинге 2.

Листинг 2 – Пример 1

patientID <- c(1, 2, 3, 4)

age <- c(25, 34, 28, 52)

diabetes <- c("Type1", "Type2", "Type1", "Type1")

status <- c("Poor", "Improved", "Excellent", "Poor")

patientdata <- data.frame(patientID, age, diabetes, status)

patientdata

Результат выполнения примера 1 представлен на рисунке 1.

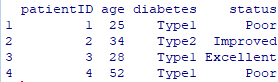


Рисунок 1 – Результат выполнения примера 1

Существует несколько способов обозначить элементы таблицы данных. Можно использовать индексы или можно указывать номера столбцов. Пример кода представлен в листингах 2 – 3.

Листинг 2 – Пример 2

patientdata [1:2]

Листинг 3 – Пример 3

patientdata [c("diabetes", "status")]

Результат выполнения примера 2 и примера 3 представлен на рисунках 2 – 3.

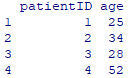


Рисунок 2 – Результат выполнения примера 2

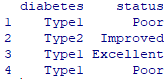


Рисунок 3 – Результат выполнения примера 3

Знак ‘$’ используется, чтобы обозначить определенную переменную в таблице данных. Пример кода представлен в листинге 4.

Листинг 4 – Пример 4

patientdata$age

Результат выполнения примера 4 представлен на рисунке 4.



Рисунок 4 – Результат выполнения примера 4

В R названия строк могут быть назначены при помощи параметра row.names функции создания таблицы данных. Пример кода представлен в листинге 5.

Листинг 5 – Пример 5

patientdata <- data.frame(patientID, age, diabetes, status, row.names=patientID)

patientdata

Результат выполнения примера 5 представлен на рисунке 5.

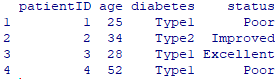


Рисунок 5 – Результат выполнения примера 5

Функция factor() сохраняет категориальные данные в виде вектора из целых чисел в диапазоне от одного до k (где k – число уникальных значений категориальной переменной) и в виде внутреннего вектора из цепочки символов (исходных значений переменной), соответствующим этим целым числам.

По умолчанию уровни фактора присваиваются значениям вектора в алфавитном порядке. Для упорядоченных факторов редко подходит алфавитный порядок уровней, предлагающийся по умолчанию. Установку по умолчанию можно изменить при помощи параметра levels.

Пример использования функции factor() представлены в листинге 6.

Листинг 6 – Пример 6

patientID <- c(1, 2, 3, 4)

age <- c(25, 34, 28, 52)

diabetes <- c("Type1", "Type2", "Type1", "Type1")

status <- c("Poor", "Improved", "Excellent", "Poor")

diabetes <- factor(diabetes)

status <- factor(status, order=TRUE)

patientdata <- data.frame(patientID, age, diabetes, status)

Получить информацию об объекте можно при помощи функции str(object) (листинг 7).

Листинг 7 – Пример использования str(object)

str(patientdata)

Результат использования функции str(patientdata) представлен на рисунке 6.

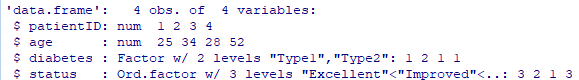


Рисунок 6 – Результат использования функции str(patientdata)

Получить статистику по объекту можно при помощи функции summary(object) (листинг 8).

Листинг 8 – Пример использования summary(object)

summary(patientdata)

Результат использования функции summary(patientdata) представлен на рисунке 7.

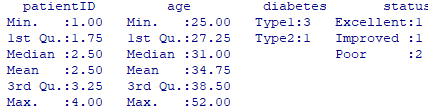


Рисунок 7 – Результат использования функции summary(patientdata)

Списки – это самый сложный тип данных в R. Фактически список – это упорядоченный набор объектов (компонентов). Список может объединять разные (возможно, не связанные между собой) объекты под одним именем. К примеру, список может представлять собой сочетание векторов, матриц, таблиц данных и даже других списков. Примеры использования списков представлены в листинге 9.

Листинг 9 – Пример 7

g <- “My First List”

h <- c(25, 26, 18, 39)

j <- matrix(1:10, nrow=5)

k <- c(“one”, “two”, “three”)

mylist <- list(title=g, ages=h, j, k)

Полученный список отобразили в консоли (рисунки 8 – 10)

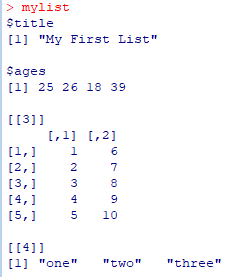


Рисунок 8 – Отображение всего списка



Рисунок 9 – Отображение второго объекта списка



Рисунок 10 – Отображение объекта списка по имени

Ввод данных с клавиатуры.

Самый простой способ введения данных – это ввод с клавиатуры. Функция edit() откроет текстовый редактор, куда можно внести свои данные. Для ввода данных необходимо:

1. Создать пустую таблицу данных (или матрицу), указав названия и типы переменных;

2. Открыть текстовый редактор с этим объектом, ввести экспериментальные данные и сохранить результат в виде объекта с данными.

Пример использования данного механизма представлен в листинге 10. Окно редактора представлено на рисунке 11.

Листинг 10 – Пример 8

mydata <- data.frame(age=numeric(0), gender=character(0), weight=numeric(0)) mydata <- edit(mydata)

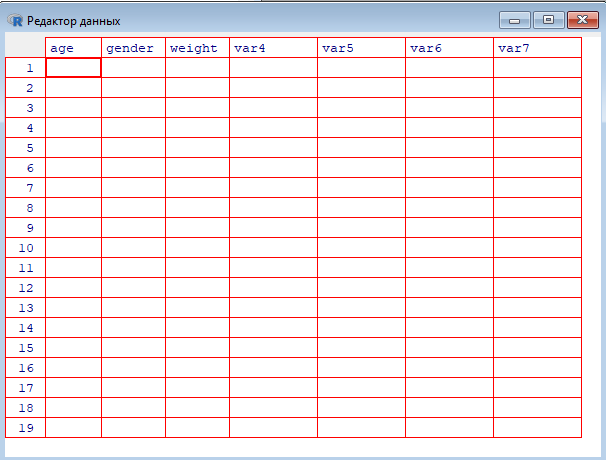


Рисунок 11 – Окно редактора

Импорт данных из текстового файла с разделителями.

Импорт данных из текстовых файлов с разделителями возможен при помощи команды read.table(), функции, которая сохраняет данные в виде таблицы. mydataframe <- read.table(file, header=логичское\_значение, sep=”разделитель”, row.names=”название”) где file – это ASCII файл с разделителями, header – это логическое значение, определяющее, содержит ли первая строка названия переменных (TRUE – да, FALSE – нет), sep указывает, каким символом разделены элементы данных, а row.names – необязательный параметр, для указания столбца (столбцов), в котором содержатся названия строк. Пример кода представлен в листинге 11.

Листинг 11 – Пример 9

grades <- read.table(“studentgrades.csv”, header=TRUE, sep=”,”, row.names=”STUDENTID”)

grades

Результат выполнения примера 9 представлен на рисунке 12.

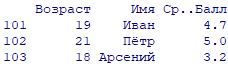


Рисунок 12 – Результат выполнения примера 9

Импорт данных из Excel.

Лучший способ прочесть файл в формате Excel – это сохранить его в формате текстового файла с разделителями и импортировать в R, как это описано выше. Функция read.xlsx() осуществляет импорт нужного листа XLSX-файла в таблицу данных. Проще всего использовать эту функцию по такой схеме: read.xlsx(file, n), где file – это путь к файлу книги Excel, а n – число листов, которые нужно импортировать. Пример кода представлен в листинге 12.

Листинг 12 – Пример 10

library(xlsx)

workbook <- “c:/myworkbook.xlsx”

mydataframe <- read.xlsx(workbook, 1)

mydataframe

Результат выполнения примера 10 представлен на рисунке 13.



Рисунок 13 – Результат выполнения примера 10

Лабораторная работа № 2.2

Корреляционный и регрессионный анализ данных. Работа с диаграммами

2.2.1 Цель работы

− исследовать возможности языка R для проведения корреляционного и регрессионного анализа данных;

− исследовать возможности языка R для создания и изменения вида диаграмм

2.2.2 Порядок выполнения работы

1. Ознакомится с методическими указаниями

2. Исследовать основные функции и команды языка R, представленные в данной лабораторной работе

3. Выполнить все примеры.

4. Реализовать код и прописать комментарии к каждому действию, продемонстрированный в листинге 13.

Листинг 13 – Код программы

x <- c(1:10)

y <- x

z <- 10/x

opar <- par(no.readonly=TRUE)

par(mar=c(5, 4, 4, 8) + 0.1)

plot(x, y, type=”b”, pch=21, col=”red”, yaxt=”n”, lty=3, ann=FALSE)

lines(x, z, type=”b”, pch=22, col=”blue”, lty=2)

axis(2, at=x, labels=x, col.axis=”red”, las=2)

axis(4, at=z, labels=round(z, digits=2), col.axis=”blue”, las=2, cex.axis=0.7, tck=-0.01)

mtext(“y=1/x”, side=4, line=3, cex=1, las=2, col=”blue”)

title(“Пример осей”, xlab=”значение переменной X”, ylab=”Y=X”)

par(opar)

5. Изучить самостоятельно добавление Легенды к диаграмме

6. По экспериментальным данным провести построение 3 - 4 различных диаграмм

7. Используя изученные функции и команды провести модификацию построенных графиков.

2.2.3. Ход работы

Работа с диаграммами.

В стандартной интерактивной сессии R создается диаграмма, вводом по одной команде и добавлением элементов диаграммы, пока не получится то, что необходимо. Пример кода создания диаграммы представлен в листинге 14.

Листинг 14 – Пример 11

attach(mtcars)

plot(wt, mpg)

abline(lm(mpg~wt))

title(“Regression of MPG on Weight”)

detach(mtcars)

Результат работы примера 11 представлен на рисунке 14.

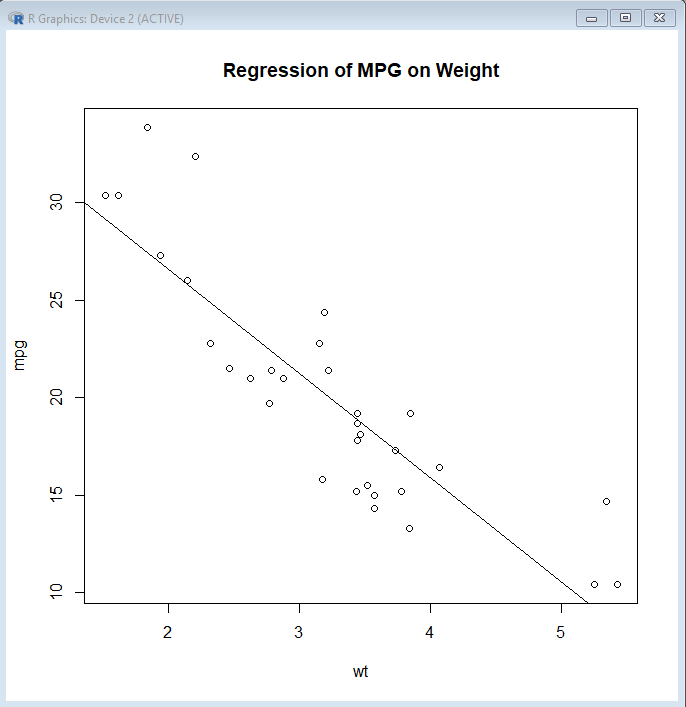


Рисунок 14 – Результат работы примера 11

Чтобы сохранить диаграмму в формате PDF под названием mygraph.pdf в текущей рабочей директории можно использовать код, представленный в листинге 15.

Листинг 15 – Пример 12

pdf(“mygraph.pdf”)

attach(mtcars)

plot(wt, mpg)

abline(lm(mpg~wt))

title(“Regression of MPG on Weight”)

detach(mtcars)

dev.off()

Результат работы примера 12 представлен на рисунке 15.

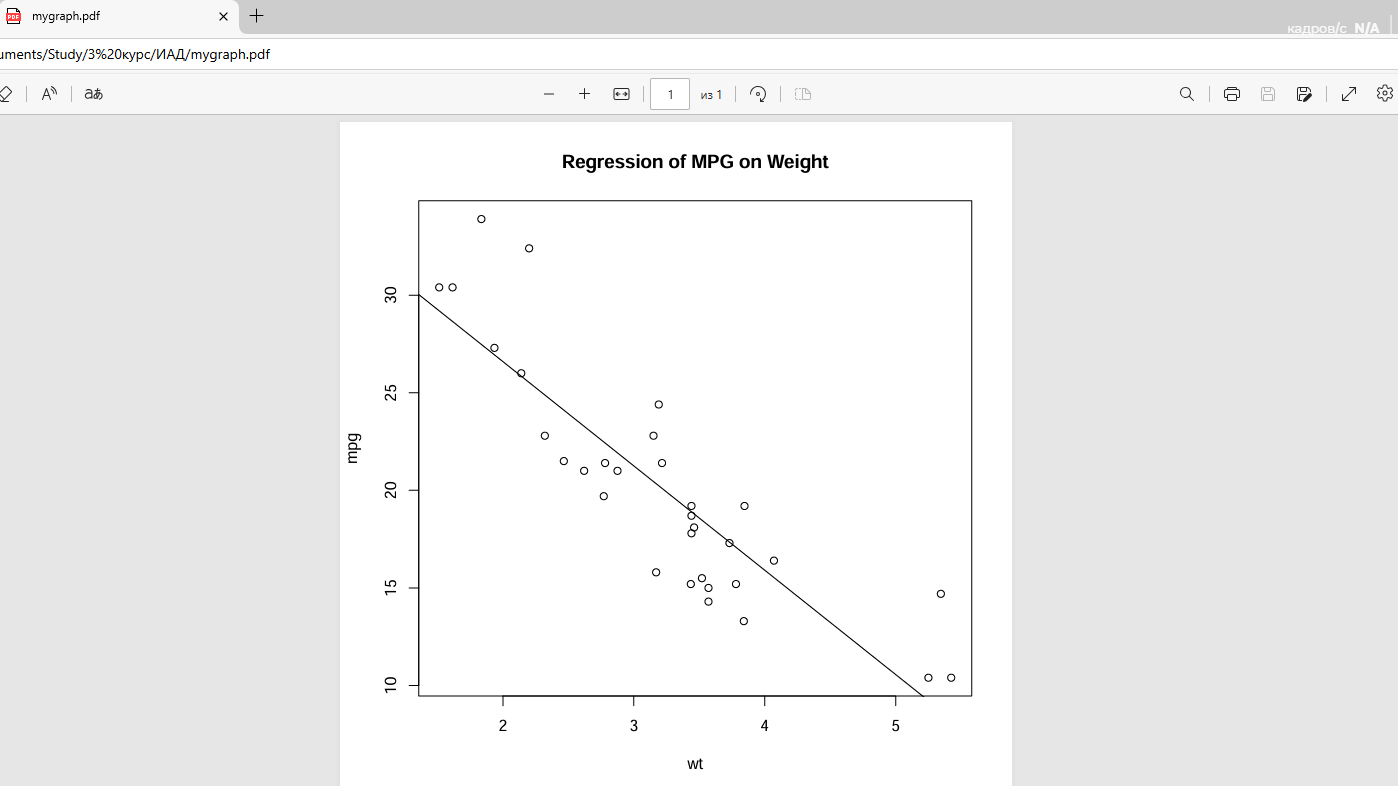


Рисунок 15 – Результат работы примера 12

Представлен набор данных, который описывает реакцию пациента на два лекарства в пяти дозировках (таблица 1).

Таблица 1 – Реакция пациента на два лекарства в пяти дозировках

|  |  |  |
| --- | --- | --- |
| Дозировка | Реакция на лекарство А | Реакция на лекарство В |
| 20 | 16 | 15 |
| 30 | 20 | 18 |
| 40 | 27 | 25 |
| 45 | 40 | 31 |
| 60 | 60 | 40 |

Код ввода и отображения данных представлен в листинге 16.

Листинг 16 – Ввод и отображение данных о пациенте

dose <- c(20, 30, 40, 45, 60)

drugA <- c(16, 20, 27, 40, 60)

drugB <- c(15, 18, 25, 31, 40)

plot(dose, drugA, type=”b”)

Результат работы кода из листинга 16 представлен на рисунке 16.

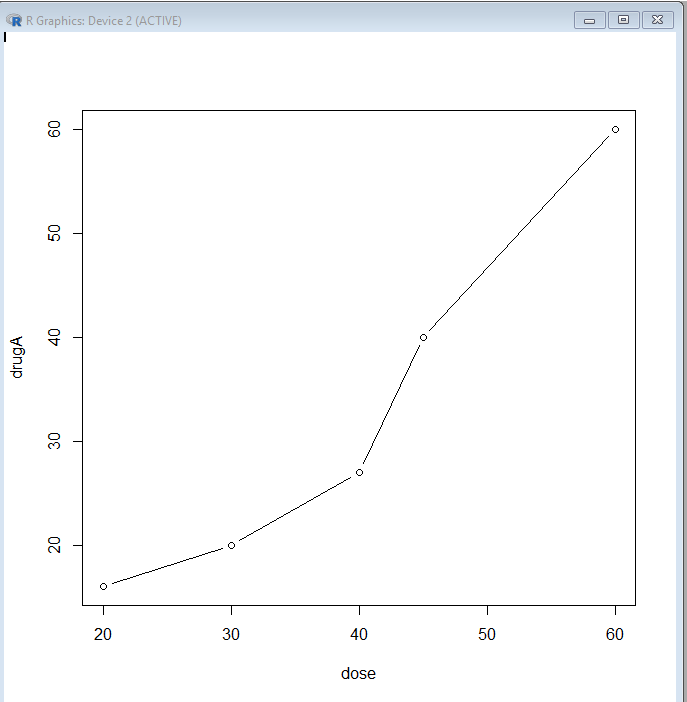


Рисунок 16 – Отображение данных о пациенте

Графические параметры.

Многие характеристики диаграмм (шрифты, цвета, оси, названия) можно изменять при помощи опций, которые называются «графические параметры». Один способ назначить эти параметры – использовать функцию par(). Значения параметров, заданные таким способом, будут действовать на протяжении всей сессии, пока вы не измените их. Формат применения функции таков: par(название параметра=назначение, название параметра=назначение, ...). Функция par() без аргументов выводит на экран действующие значения графических параметров. Добавление аргумента no.readonly=TRUE позволяет увидеть только те графические параметры, которые можно изменять. Пример использования данной функции представлен в листинге 17.

Листинг 17 – Пример 13

opar <- par(no.readonly=TRUE)

par(lty=2, pch=17)

plot(dose, drugA, type=”b”)

par(opar)

Результат выполнения примера 13 представлен на рисунке 17.

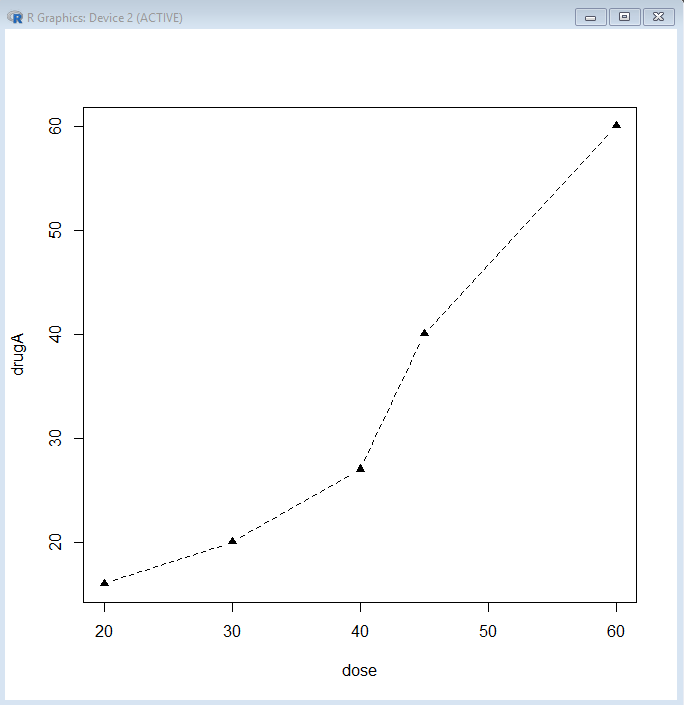


Рисунок 17 – Результат выполнения примера 13

Графические параметры также используются для определения размера, шрифта и стиля текста.

Пример использования функции par() для определения размера, стиля и шрифта текста представлен в листинге 18.

Листинг 18 – Пример 14

opar <- par(no.readonly=TRUE)

dose <- c(20, 30, 40, 45, 60)

drugA <- c(16, 20, 27, 40, 60)

drugB <- c(15, 18, 25, 31, 40)

opar <- par(no.readonly=TRUE)

par(pin=c(2, 3))

par(lwd=2, cex=1.5)

par(cex.axis=.75, font.axis=3)

plot(dose, drugA, type=”b”, pch=19, lty=2, col=”red”)

plot(dose, drugB, type=”b”, pch=23, lty=6, col=”blue”, bg=”green”)

par(opar)

Результат выполнения примера 14 представлен на рисунках 18–19.

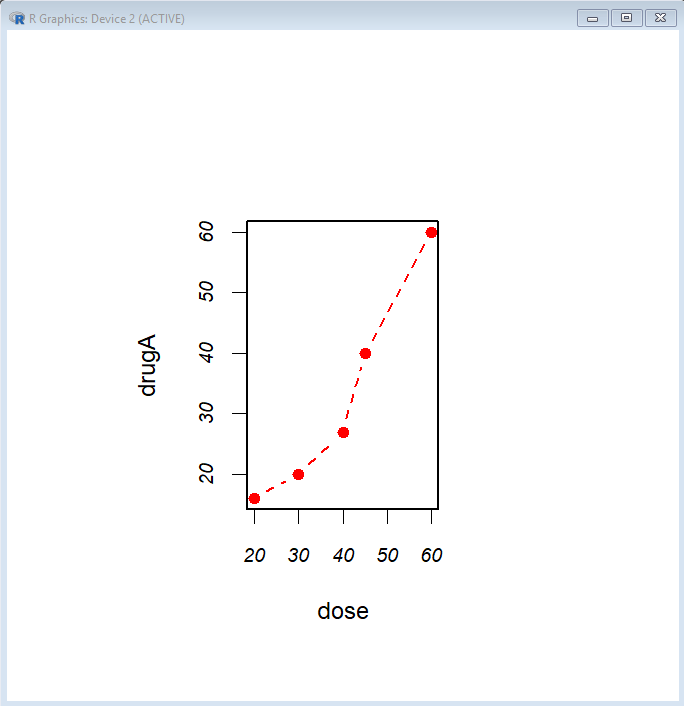


Рисунок 18 – Результат выполнения примера 14

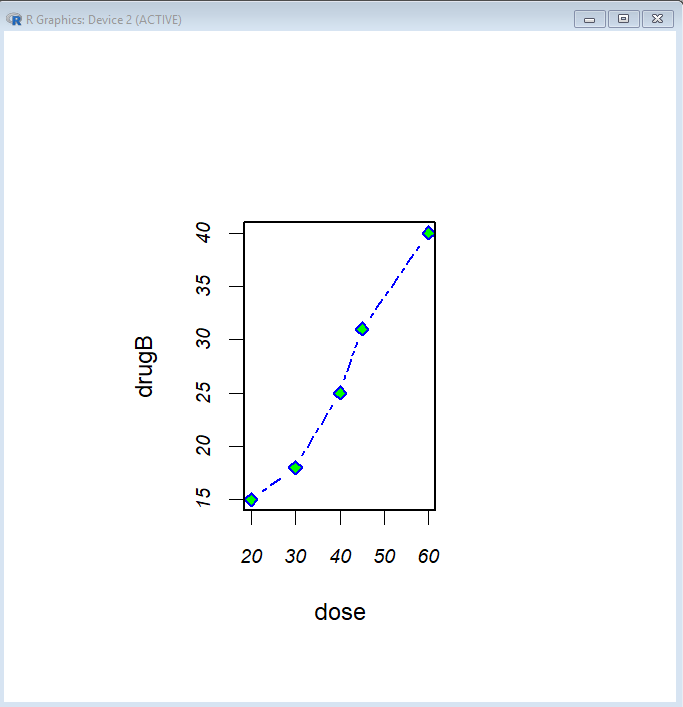


Рисунок 19 – Результат выполнения примера 14

Добавление текста, настройка параметров осей и условных обозначений.

Для многих графических функций высокого уровня (например, plot, hist, boxplot) возможен контроль не только графических параметров, но и параметров осей и надписей. К примеру, при помощи приведенного ниже программного кода можно разместить на диаграмме заголовок (main), подзаголовок (sub) и подписи осей (xlab, ylab), а также задать диапазон значений на осях (xlim, ylim). Пример кода представлен в листинге 19.

Листинг 19 – Пример 15

plot(dose, drugA, type="b",

+ col="red", lty=2, pch=2, lwd=2,

+ main="клинические испытания прпарата A",

+ sub="это вымышленные данные",

+ xlab="Доза", ylab="Эффект от препарата",

+ xlim=c(0, 60), ylim=c(0, 70))

Результат выполнения примера 15 представлен на рисунке 20.

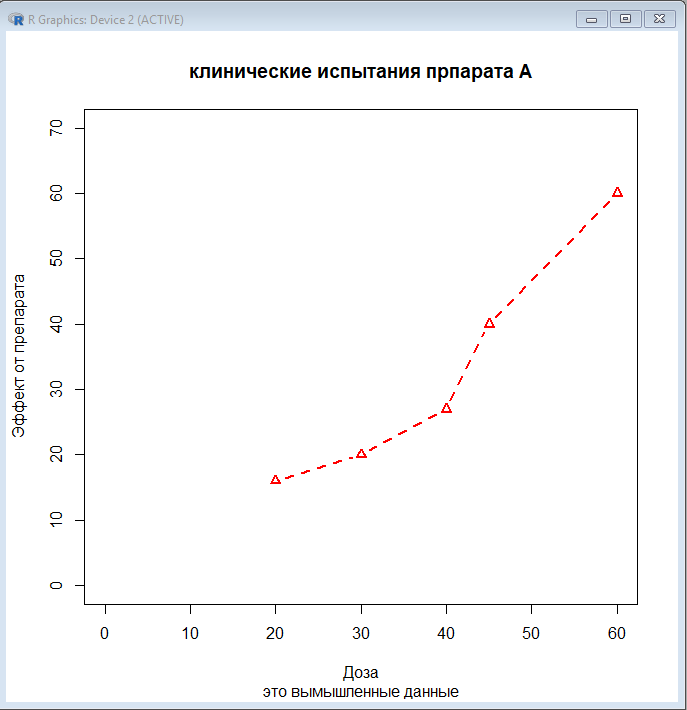


Рисунок 20 – Результат выполнения примера 20

Заголовки.

Для размещения заголовков и подписей осей на диаграмме используется функция title(). Графические параметры (такие как размер и тип шрифта, ориентация и цвет текста) также можно задать при помощи функции title(). Пример использования функции title() представлен в листинге 20.

Листинг 20 – Пример 16

> plot(dose, drugA, type = "b",

+ lty = 2, pch = 2, lwd = 2,

+ ann=FALSE)

> title(main=" Мой\_заголовок ", col.main="red",

+ sub="мой подзаголовок ", col.sub="blue",

+ xlab="моя\_подпись\_по\_оси\_x ", ylab="моя\_подпись\_по\_оси\_Y",

+ col.lab="green", cex.lab=0.75)

Результат выполнения примера 16 представлен на рисунке 21.

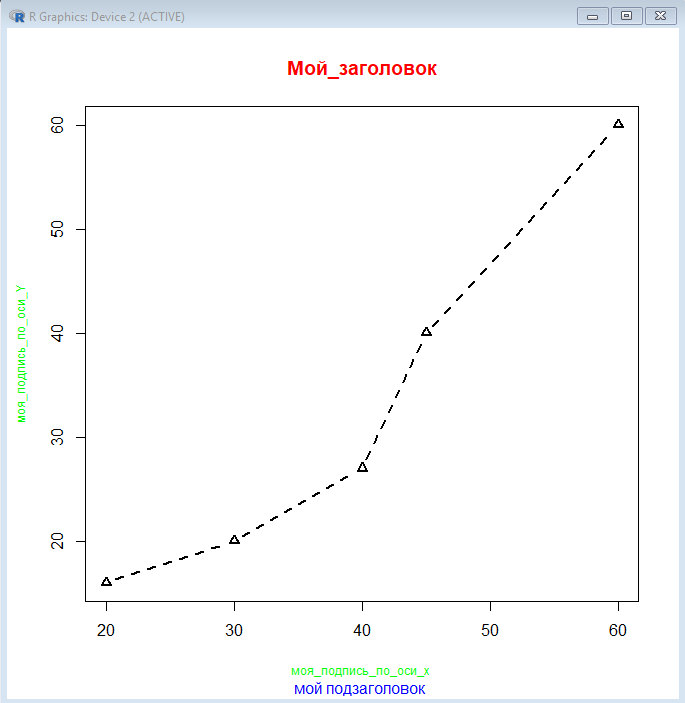


Рисунок 21 – Результат выполнения примера 16

Цвета.

В R цвета можно обозначать номером, названием, в шестнадцатеричной системе, а также в системах RBG или HSV. Например, col=1, col=”white”, col=”#FFFFFF”, col=rgb(1,1,1) и col=hsv(0,0,1) – взаимозаменяемые способы обозначить белый цвет. Функция rgb() определяет цвета по значениям красного, зеленого и синего, а hsv() основана на значениях оттенка и насыщенности.

Функция colors() выводит на экран список всех доступных цветов.

В R также реализован ряд функций, которые позволяют создавать векторы из близких цветов. К таким функциям относятся rainbow(). Например, rainbow(10) создает 10 соседних "радужных" цветов. Оттенки серого создаются функцией gray(). В этом случае задаются оттенки серого в виде вектора чисел от 0 до 1. Команда gray(0:10/10) создаст 10 оттенков серого.

Пример использования функций rainbow() и gray() представлен в листинге 18.

Листинг 18 – Пример 14

n <- 10

mycolors <- rainbow(n)

pie(rep(1, n), labels=mycolors, col=mycolors)

mygrays <- gray(0:n/n)

pie(rep(1, n), labels=mygrays, col=mygrays)

Результат выполнения примера 14 представлен на рисунках 18–19.

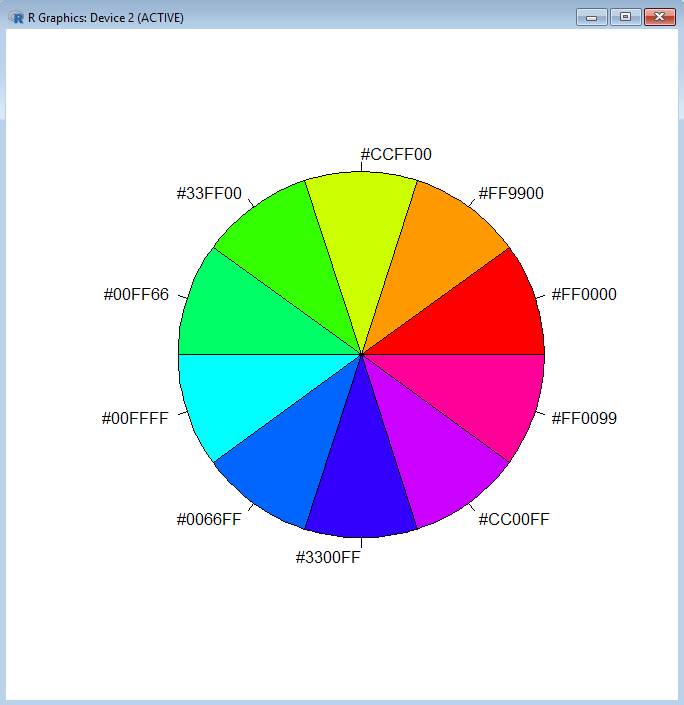


Рисунок 18 – Результат выполнения примера 14

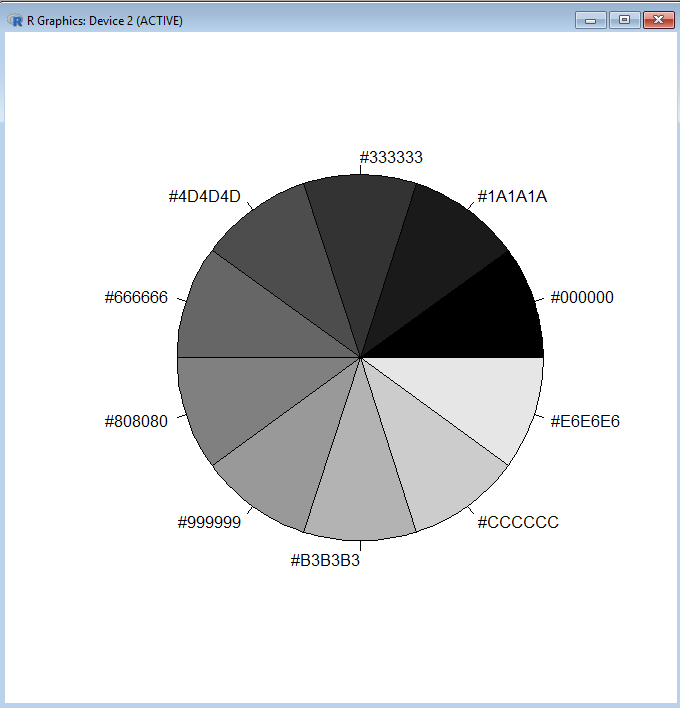


Рисунок 19 – Результат выполнения примера 14

Был проанализирован код из листинга 13. К каждому действию написали комментарий (листинг 19).

Листинг 19 – Код программы с комментариями

x <- c(1:10) #Создать вектор целых чисел от 1 до 10. x теперь ссылается на него

y <- x #y ссылается на тоже объект, что и x

z <- 10/x #z ссылается на новый вектор с результатом поэлементного деления 10/x

opar <- par(no.readonly=TRUE) #Сохранить все изменяемые параметры в переменную opar

par(mar=c(5, 4, 4, 8) + 0.1) #Задать поля: низ=5, лево=4, верх=4, право=8

plot(x, y, #Новый график, значения y по оси Y, значения x по оси X

type="b", #Отобразить точки и линии

pch=21, #Тип маркера круг без заливки

col="red", #Красный цвет линий и маркеров

yaxt="n", #Отключить ось Y

lty=3, #Тип линии - короткий пунктир

ann=FALSE) #Отключить автоматическую подпись осей

lines(x, z, type="b", pch=22, col="blue", lty=2) #Создать линию на графике. Маркер - Квадрат без заливки. Линия - пунктир. Цвет – синий

axis(2, at=x, labels=x, col.axis="red", las=2) #Создать 2ю ось(Y). Позиции для делений и их подписи из x. Цвет подписей - красный. Подписи горизонтальны.

axis(4, at=z, #Создать ось. Вертикальная ось справа. Значения по z

labels=round(z, digits=2), #Подписи - округлённые значения до двух знаков вектора z

col.axis="blue", #Цвет оси - синий

las=2, cex.axis=0.7, tck=-0.01) #Горизонтальные подписи. Текст меньше. Деления немного меньше (часть внутрь графика)

mtext("y=1/x", side=4, line=3, cex=1, las=2, col="blue") #Создать подпись справа от графика с отступом 3, масштабом 1, горизонтальной ориентацией и синего цвета

title("Пример осей", xlab="значение переменной X", ylab="Y=X") #Задать заголовок первым параметром, подпись оси X вторым, и подпись левой вертикальной оси третьим

par(opar) #вернуть параметры к исходным

Результат работы программы представлен на рисунке 20.

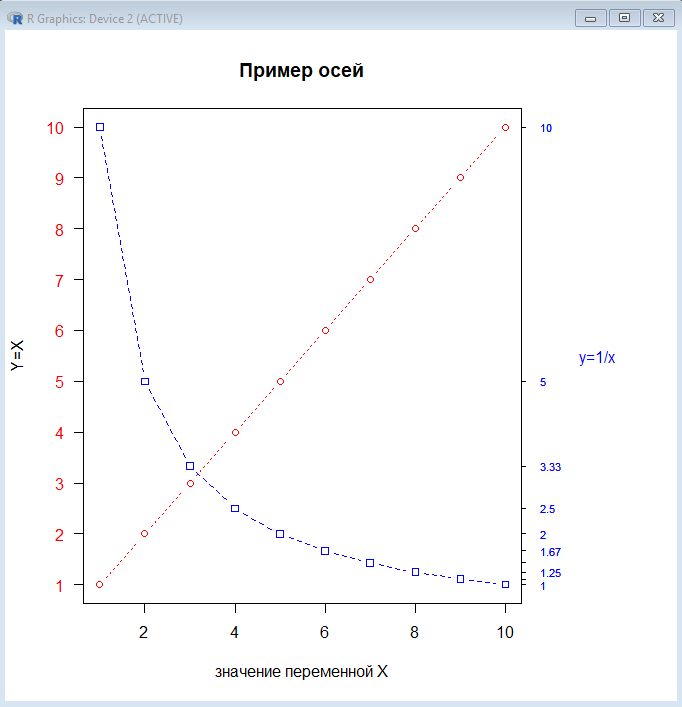


Рисунок 20 – Результат работы программы

Лабораторная работа №2.3

Корреляционный и регрессионный анализ данных. Исследование тесноты взаимосвязей данных в среде R

2.3.1 Цель работы

− исследовать возможности языка R для определения тесноты взаимосвязей экспериментальных данных;

2.3.2 Порядок выполнения работы

1. Выполнить пример указанный в методических указаниях

2. Провезти анализ собственного датасета и выявить корреляцию между объектами