МИНИСТЕРСТВО СЕЛЬСКОГО ХОЗЯЙСТВА РОССИЙСКОЙ ФЕДЕРАЦИИ

ДЕПАРТАМЕНТ НАУЧНО-ТЕХНОЛОГИЧЕСКОЙ ПОЛИТИКИ И ОБРАЗОВАНИЯ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ «ДОНСКОЙ ГОСУДАРСТВЕННЫЙ АГРАРНЫЙ УНИВЕРСИТЕТ»
(ФГБОУ ВО Донской ГАУ)

АЗОВО-ЧЕРНОМОРСКИЙ ИНЖЕНЕРНЫЙ ИНСТИТУТ – ФИЛИАЛ
ФЕДЕРАЛЬНОГО ГОСУДАРСТВЕННОГО БЮДЖЕТНОГО ОБРАЗОВАТЕЛЬНОГО
УЧРЕЖДЕНИЯ ВЫСШЕГО ОБРАЗОВАНИЯ
«ДОНСКОЙ ГОСУДАРСТВЕННЫЙ АГРАРНЫЙ УНИВЕРСИТЕТ» В Г. ЗЕРНОГРАДЕ
(Азово-Черноморский инженерный институт ФГБОУ ВО Донской ГАУ)

Кафедра высшей математики
и механики

Д.В. Степовой, А.А. Емелин, А.П. Жогалев

# СТАТИСТИЧЕСКИЕ МЕТОДЫ ОБРАБОТКИ ЭКСПЕРИМЕНТАЛЬНЫХ ДАННЫХ И ПЛАНИРОВАНИЕ ЭКСПЕРИМЕНТА

*Лабораторный практикум*

Зерноград – 2016

УДК 519.252

*Печатается по решению методической комиссии по специальности 23.05.01 – «Наземные транспортно-технологические средства» Азово-Черноморского инженерного института – филиала Федерального государственного бюджетного образовательного учреждения высшего образования «Донской государственный аграрный университет» в г. Зернограде*

**Рецензенты:**
докт. техн. наук, профессор кафедры высшей математики ***Коптева Н.А.***;
канд. техн. наук, доцент кафедры «Тракторы и автомобили» ***Оберемок В.А.***

**Степовой, Д.В**. Статистические методы обработки экспериментальных данных и планирование эксперимента / Д.В. Степовой, А.А. Емелин, А.П. Жогалев. – Зерноград: Азово-Черноморский инженерный институт ФГБОУ ВО Донской ГАУ, 2016 – 115 с.

Лабораторный практикум содержит базовые теоретические и практические сведения, овладев которыми любой студент сможет обрабатывать экспериментальные данные и строить на их основе модели. Состоит из семи лабораторных работ, проделав их и освоив материал, студент сможет получить первичные знания и навыки в статистической обработке экспериментальных данных с использованием информационных технологий.

Лабораторный практикум предназначен для студентов обучающихся по специальности 23.05.01 «Наземные транспортно-технологические средства» очной и заочной форм обучения при изучении дисциплины «Статистические методы обработки экспериментальных данных».

Рассмотрено и одобрено на заседании кафедры высшей математики.
Протокол № 2 от 26 сентября 2016 г.

Рассмотрено и одобрено методической комиссией по специальности 23.05.01 «Наземные транспортно-технологические средства» методического совета агротехнологического факультета.
Протокол № 1 от 27 сентября 2016 г.

© Степовой Д.В., Емелин А.А., Жогалев А.П., 2016
© Азово-Черноморский инженерный институт – филиал ФГБОУ ВО Донской ГАУ, 2016

# Содержание

Предисловие ..... 4

Лабораторная работа № 1. Векторные данные ..... 6

Лабораторная работа № 2. Простейшие операции. Операторы цикла и условия. Создание собственных функций ..... 20

Лабораторная работа № 3. Базовая графика в R ..... 32

Лабораторная работа № 4. Математическая статистика. Числовые характеристики вариационного ряда ..... 48

Лабораторная работа № 5. Законы распределения случайных величин ..... 57

Лабораторная работа № 6. Обработка данных − выделение категорий. Построение вариационного ряда ..... 79

Лабораторная работа № 7. Проверка статистических гипотез ..... 93

Литература ..... 107

Приложение ..... 108

# Предисловие

«*Статистика знает все*»
Ильф И.А., Петров Е.П. «Двенадцать стульев»

Лабораторный практикум предназначен для студентов СПО, изучающих дисциплину «Статистические методы обработки экспериментальных данных».

Освоение студентами материла учебного пособия позволит им освоить следующие компетенции:

– способность представить современную картину мира на основе целостной системы естественнонаучных и математических знаний, ориентироваться в ценностях бытия, жизни, культуры (ОК-1);

– способность самостоятельно применять методы и средства познания, обучения и самоконтроля для приобретения новых знаний и умений, в том числе в новых областях, непосредственно не связанных со сферой деятельности, развития социальных и профессиональных компетенций (ОК-8);

– способность проводить теоретические и экспериментальные научные исследования по поиску и проверке новых идей совершенствования автомобилей и тракторов (ПСК-1.2).

Целью лабораторного практикума является: целостное представление о теории математической статистики, раскрыть возможности программного обеспечения для обработки статистических данных и построения моделей и эмпирических зависимостей. В качестве такого программного средства выбрано программное обеспечение R.

R – среда для статистических расчётов. R возник как свободный аналог среды S-PLUS, которая, в свою очередь, является коммерческой реализацией языка расчётов S.

В августе 1993 г. двое молодых новозеландских ученых анонсировали свою новую разработку, которую назвали R. По замыслу создателей, Роберта Джентльмена (Robert Gentleman) и Росса Ихака (Ross Ihaka), она должна была стать новой реализацией языка S, отличающейся от S-PLUS некоторыми деталями, например, обращением с глобальными и локальными переменными, а также работой с памятью. Фактически, они создали не полный аналог S-PLUS, а новую «ветку» на «дереве S»[1].

В настоящее время реализации R существуют для трёх наиболее распространённых семейств операционных систем: GNU/Linux, Apple Mac OS X и Microsoft Windows, а в распределённых хранилищах системы CRAN по состоянию на конец сентября 2010 года были доступны для свободной загрузки 2548 пакетов расширения, ориентированных на специфические задачи обработки данных, возникающие в эконометрике и финансовом анализе, генетике и молекулярной биологии, экологии и геологии, медицине и фармацевтике и многих других прикладных областях [2]. Значительная часть европейских и американских университетов в последние годы активно переходят к исполь-

зованию R для обработки массивов данных в учебной и научно-исследовательской деятельности вместо дорогостоящих коммерческих разработок.

Серьезным недостатком R считают использование команд и необходимость знания их. Отсюда трудность обучения работы с программным приложением. Но эта слабость является одновременно и его преимуществом. Например, задача преобразования выборки объёмом из девяти элементов в матрицы размером 3×3 решается одной командой

```
> b<-matrix(b,ncol=3)
>b
     [,1] [,2] [,3]
[1,]    1  4.2 -2.0
[2,]    3  5.0  0.8
[3,]   -3  5.7  2.0
```

Чтобы сделать это в программе с визуальным интерфейсом, например STATISTICA, требуется (1) создать две новые переменные, (2–3) скопировать дважды тройки из выборки в буфер, (4–5) скопировать из буфера в созданные переменные и (6) и уничтожить лишние строки [3].

***Как работать с лабораторным практикумом?***

Лабораторный практикум построен таким образом, чтобы активизировать познавательную деятельность студента.

Каждая лабораторная работа завершается разделом «Самостоятельная работа». Выполняя задания из него, учащийся может проверить приобретенные навыки, закрепить пройденный материал и подготовится самостоятельно применять их в своей дальнейшей учебе или работе.

Упражнения в каждой лабораторной состоят из отдельных пронумерованных шагов (1, 2 и т.д.). Меню, названия команд, названия кнопок, диалоговых окон выделены жирным шрифтом, или взяты в кавычки («»).

Наиболее важные понятия и термины выделены курсивом. Названия клавиш, которые требуется нажать на клавиатуре, взяты в кавычки. Например, <Enter>.

Лабораторные работы нужно выполнять в строгом соответствии с методическими указаниями, внимательно читая теоретическую часть. Иллюстрации помогут оценить правильность результатов вашей работы.

Для лучшего закрепления материала ведите отчет по лабораторным работам, содержащий название лабораторной работы, её цель, результаты выполнения упражнений из раздела «Самостоятельная работа».

# Лабораторная работа № 1
# Векторные данные

**Цель работы**: изучить типы данных, их сохранение в файлы, основные команды.

**Оборудование**: ПК, оснащенный необходимым программным обеспечением.

**Объём лабораторной работы и порядок её выполнения**. Лабораторная работа выполняется в течение двух академических часов. Изучите теоретическую часть, выполните работу и упражнения для самостоятельной работы.

## 1.1 Теоретическая часть

После запуска R откроется окно приложения с приглашением:

```
R версия3.0.1 (2013-05-16) - "Good Sport"
Copyright (C) 2006 The R Foundation for Statistical
Computing
Platform: i386-w64-mingw32/i386 (32-bit)
R - это свободное ПО, и оно поставляется безо вся-
ких гарантий. Вы вольны распространять его при соблюде-
нии некоторых условий. Введите 'license()' для получе-
ния более подробной информации.
R - это проект, в котором сотрудничает множество
разработчиков. Введите 'contributors()' для получения до-
полнительной информации и 'citation()' для ознакомления с
правилами упоминания R и его пакетов в публикациях.
Введите 'demo()' для запуска демонстрационных про-
грамм, 'help()' - для получения справки, 'help.start()'
- для доступа к справке через браузер. Введите 'q()',
чтобы выйти из R.
>
```

Символ «>» является приглашением к работе. Именно после этого знака вводятся команды, которые указывают системе, что нужно делать. Например, выполнение команды округления числа (скаляра) даст следующий результат:

```
>round(d=0,1.5)
[1] 2
>round(1.5,d=0)
[1] 2
```

### Данные

Все данные в R рассматриваются как векторы. Скаляры или числа так же являются векторами, состоящими из одного элемента. Например, в предыдущем примере в строке результатов вычислений единица в квадратных скобках означала, что число 2 является вектором, состоящим из одной строки.

Ввести данные можно, например, при помощи команды c:

```
> b<-c(1, 5, 3.2, 7, 15)
> b
[1]  1.0  5.0  3.2  7.0 15.0
```

Разберем этот пример. b – это переменная, <- – это оператор присвоения, c() – это функция создания вектора.

Для ввода данных можно воспользоваться специальной процедурой R – встроенной электронной таблицей наподобие листа Excel. Вызвать её можно командой data.entry(x). Использование аргумента в этой команде обязательно. В этой таблице можно не только вводить значения переменной, но и создавать новые переменные. Аналогично работает и команда de().

Третий путь ввода данных в R проходит через чтение значения вариант из обычного текстового файла. Например, загрузить данные из файла можно командой

```
>read.table("data/mydata.txt", sep=";", head=TRUE)
```

Помимо этих способов в R можно задать последовательность чисел с единичным шагом. Например,

```
> k=2:7
> k
[1] 2 3 4 5 6 7
> k=6:2
> k
[1] 6 5 4 3 2
```

Задать числовую последовательность с произвольным шагом можно при помощи команды seq(from, to, by= ). from – это число, задающее начальное значение, to – число, задающее конечное значение, число стоящее после by = задает шаг последовательности. Например,

```
> x=seq(-2.5, 1, by=0.5)
> x
[1] -2.5 -2.0 -1.5 -1.0 -0.5  0.0  0.5  1.0
```

Как в языках программирования для обращения к элементу выборки (вектора) нужно указать имя выборки (переменной) и в квадратных скобках указать порядковый номер элемента. Например,

```
> b[3]
[1]  3.2
```

Если у элементов выборки определены имена, то обращаться к ним и по имени. Более подробно этот вариант обращения будет рассмотрен при выполнении лабораторной работы.

## Калькулятор-перероcток

Также R можно использовать как большой консольный калькулятор:

```
> 3+2
[1] 5
```

```
> 16+3/5-11*8^2
[1] -687.4
> ((((16+3)/5)-11)*8)^2
[1] 3317.76
```

Математические функции, применяемые в R по синтаксису не отличаются от функций в различных языках программирования, в том числе и VBA. Например,

```
>pi
[1] 3.141593
>sin(pi/2)*cos(pi/4)/tan(2)
[1] -0.3236128
>exp(1)
[1] 2.718282
>log(exp(1))
[1] 1
>log10(10)
[1] 1
>atan(1)
[1] 0.7853982
```

***Замечание.*** *Обращаем ваше внимание на то, что аргумент тригонометрической функции необходимо определять в радианах*.

### Работа со скриптами

Скрипт – это самостоятельно написанная программа с использованием всех возможностей R. Для запуска окна по созданию скриптов необходимо в командном меню подать команду **File->Newscript(Ctrl+N)**. Откроется новое окно, представляющее из себя текстовый редактор, в котором можно набирать и редактировать текст скрипта (программы). Сохраняется скрипт в файле с расширением *.R, или расширениями *.q, *ssc, *.S.

Запустить скрипт из окна редактора скриптов на исполнение можно командой в командном меню:

– **Edit->Run line or selection** (Ctrl+R) – запускает на исполнение отдельную выделенную строку и выделенный блок;

– **Edit->Run all** – запускает на исполнение выбранной в скрипте программы.

## 1.2 Выполнение лабораторной работы

С данными лучше работать, если они сохранены в файле. Программное обеспечение R работает с файлами только из своей рабочей папки. Установим в качестве рабочей папки корневой каталог вашей флешки. Используем для этого команды `getwd()` и `setwd()`. Команда `getwd()` показывает полное имя рабочей папки, а команда `setwd()` позволяет его изменить.

### *Упражнение 1*

1) Введите команду `getwd()` и отправьте её на исполнение нажатием клавиши «Enter».

2) Убедитесь, что рабочей папкой не является корневой каталог вашей флешки. Используя **Проводник** найдите название диска вашей флешки, например, **G:**.

3) Наберите на клавиатуре команду `setwd("g:")` и нажмите клавишу «Enter».

Результаты выполнения упражнения 1 приведены ниже.

```
>getwd()
[1] "C:/Documents and Settings/Admin/Мои документы"
>setwd("g:")
>getwd()
[1] "G:/"
```

Создадим переменную (выборку) *x*, соответствующие росту некоторых людей и введем значения вариант.

### *Упражнение 2*

1) Присвоим переменной `x` первоначальное значение командой `x = 1`.

2) Введите команду `data.entry(x)`.

3) В открывшемся окне ввода данных – электронной таблицы заполните первый столбец в соответствии с рисунком 1.1.

4) Закройте окно электронной таблицы.

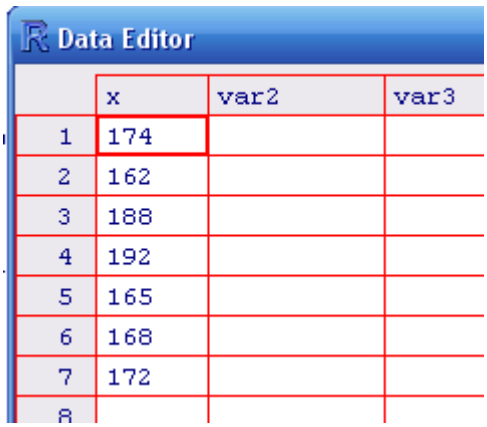



Рисунок 1.1 – Окно процедуры ввода данных

5) Для проверки ввода данных введите команду `str(x)`, позволяющую просмотреть значения выборки. Результаты выполнения этой команды будут следующие:

```
>str(x)
num [1:7] 174 162 188 192 165 168 172
```

num (numerical) означает, что вектор содержит числовые значения, [1:7] – порядковые номера (индексы) изменяются от 1 до 7.

Введем новую переменную sex, описывающую пол этих людей, набирая значения прямо в командной строке.

### *Упражнение 3*

1) Наберите в командной строке

```
sex<- c("male", "female", "male", "male",
+"female", "male", "male")
```

2) Результат ввода данных проверьте командой str(sex):

```
>sex<- c("male", "female", "male", "male",
+"female", "male", "male")
>str(sex)
chr [1:7] "male" "female" "male" "male" "female" "male" ...
```

chr означает, что варианты переменной есть текстовые данные.

Однако в таком варианте использовать выборку sex нельзя. Может обрабатывать только числовую информацию. Выборку sex нужно разбить на категории, то есть ввести цифровую индексацию текстовых значений.

### *Упражнение 4*

1) Введите команду sex.f<-factor(sex).

2) Просмотрите изменение значений выборки командой sex.f. Их нет. Все значения сохранились.

3) Еще просмотрите информацию о выборке при помощи команды str(sex.f).

Вы увидите, что основными значениями её элементов являются цифры 1 и 2.

4) Постройте график этой выборки командой plot(sex.f).

Результаты работы команд будут следующими:

```
>sex.f<-factor(sex)
>sex.f
[1] male   female male   male   female male   male
Levels: female male
>str(sex.f)
 Factor w/ 2 levels "female","male": 2 1 2 2 1 2 2
>plot(sex.f)
```

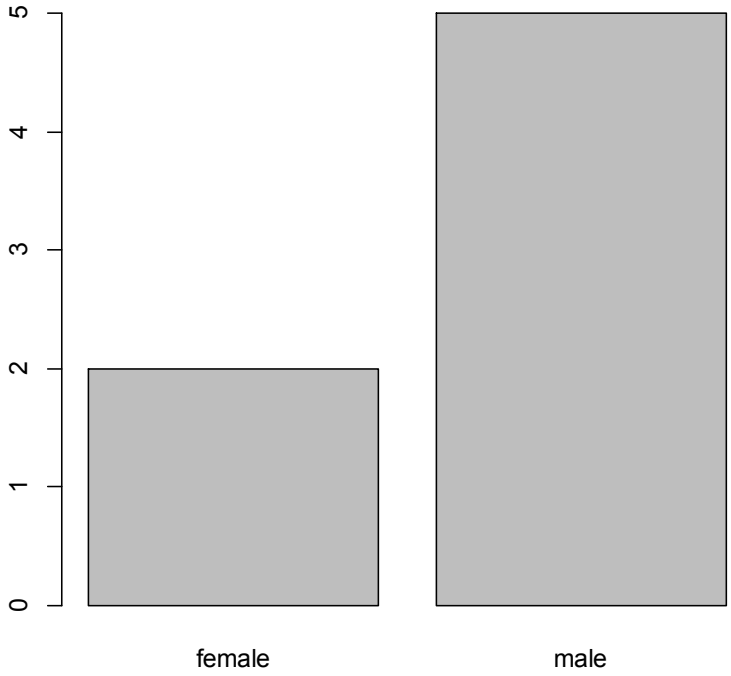



Рисунок 1.2 – Диаграмма выборки `sex.f`

Очень удобно перед вводом команды описать её действие при помощи комментария. Комментарии обозначаются символом «#» в начале строки. Введем вес этих людей и их имена, как идентификаторы значений веса.

### *Упражнение 5*

1) Опишем ввод значений веса при помощи комментария. Наберите в командной строке `# Вектор веса`.

2) Для ввода веса наберите команду `w<-c(69,68,93,87,59,82, 72).`

3) Зададим имена людей командой:

```
names(w)<-c("Коля","Женя","Петя","Саша","Катя", "Вася", "Жо-
ра")
```

4) Просмотрим полученный результат командой `w`.

Результаты выполнения этого упражнения:

```
> # Вектор веса
>w<- c(69, 68, 93, 87, 59, 82, 72)
>names(w) <- c("Коля", "Женя", "Петя", "Саша",
+"Катя", "Вася", "Жора")
>w
Коля Женя Петя Саша Катя Вася Жора
  69   68   93   87   59   82   72
```

Теперь к элементам выборки можно обращаться по имени. Например,

```
> w["Женя"]
Женя
  68
```

Обратимся к одному из самых важных типу данных – таблицам данных. «***Таблицы данных*** – *это гибридный тип представления, одномерный список из векторов одинаковой длины*» [1].

Создают таблицы данных командой `data.frame()`. Аргументами этой функции являются имена переменных (столбцов), из которых и состоит таблица.

Создадим таблицу данных, состоящую из переменных `w`, `x` и `sex.f`.

### *Упражнение 6*

1) Введите команду

```
d <- data.frame(weight=w, height=x, sex=sex.f).
```

2) Командой `d` просмотрите результат.

Результаты выполнения этого упражнения:

```
>d<- data.frame(weight=w, height=x, sex=sex.f)
>d
weight height    sex
Коля     69    174   male
Женя     68    162 female
Петя     93    188   male
Саша     87    192   male
Катя     59    165 female
Вася     82    168   male
Жора     72    172   male
```

Для обращения к данным из таблицы необходимо указывать не только её имя, но и имя, или порядковый номер столбца.

### *Упражнение 7*

1) Просмотрите данные второго столбца командой `d$height`.

2) Просмотрите данные этого столбца можно и командой `d[[2]]`.

3) Эти же данные можно просмотреть и командой `d[,2]`.

4) Просмотрите данные первого столбца командой `d[,1]`.

5) Для обращения к элементу таблицы нужно указать не только номер столбца, но и номер таблицы. Например, `d[3,2]`.

6) Командой `d[4,]` просмотрите данные из четвертой строки.

7) Просматривать данные из таблицы можно используя в качестве индексации – логические операторы. Для просмотра данных о женщинах введите команду `d[d$sex=="female",]`.

Результаты выполнения этого упражнения:

```
>d$height
[1] 174 162 188 192 165 168 172
>d[[2]]
[1] 174 162 188 192 165 168 172
```

```
>d[,2]
[1] 174 162 188 192 165 168 172
>d[,1]
[1] 69 68 93 87 59 82 72
>d[3,2]
[1] 188
>d[4,]
weight height  sex
Саша     87    192 male
>d[d$sex=="female",]
weight height    sex
Женя     68    162 female
Катя     59    165 female
```

Последний тип данных, который разберем в этой лабораторной работе – это матрицы. В алгебре под матрицей понимают прямоугольную таблицу, которая содержит однотипные данные. Матрица в R – это просто специальный тип вектора, обладающий некоторыми добавочными свойствами (атрибутами), позволяющими интерпретировать его как совокупность строк и столбцов.

### *Упражнение 8*

1) Создайте командой `k = 1:9` вектор – числовую последовательность от 1 до 9.

2) Создадим матрицу `ma` размерности 3×3, разбивая вектор `k` построчно командой `ma<- matrix(k, ncol = 3, byrow = TRUE)`.

3) Командой `ma` просмотрим результат.

4) Присвоим новому вектору `mb` значения вектора `k` командой `mb = k`.

5) Преобразуем этот вектор `mb` в матрицу 3×3, заполняя в ней данные по столбцам, командой `attr(mb, "dim")<-c(3,3)`.

6) Просмотрим результат сначала командой `mb`.

7) Просмотрим информацию об этой матрице командой `str(mb)`. Обратите внимание на результат её выполнения. Убедитесь, что преобразование вектора в матрицу имеет лишь виртуальный характер.

Результаты выполнения этого упражнения:

```
> k = 1:9
>ma<- matrix(k, ncol=3, byrow=TRUE)
>ma
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9
>mb = k
>attr(mb, "dim")<-c(3,3)
```

```
>mb
[,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
>str(mb)
int [1:3, 1:3] 1 2 3 4 5 6 7 8 9
```

Над этими матрицами можно выполнять любые операции.

### *Упражнение 9*

1) Введите команду `2*ma-4*mb`. Просмотрите результат вычислений.

2) Введите `mb/5`.

3) Транспонируется матрица командой `t(mb)`.

4) Введите команду `mb*ma`. Обратите внимание на то, что умножение произошло только по каждому элементу. В алгебре результат умножения этих матриц будет следующим:

$$mb \cdot ma = \begin{matrix} 66 & 78 & 90 \\ 78 & 93 & 108 \\ 90 & 108 & 126 \end{matrix}.$$

5) То же самое произойдет при делении 1 на матрицу `mb`. Система разделит 1 на каждый элемент матрицы `mb` и поместит его в новую матрицу.

Результаты выполнения этого упражнения:

```
> 2*ma-4*mb
[,1] [,2] [,3]
[1,]   -2  -12  -22
[2,]    0  -10  -20
[3,]    2   -8  -18
>mb/5
     [,1] [,2] [,3]
[1,]  0.2  0.8  1.4
[2,]  0.4  1.0  1.6
[3,]  0.6  1.2  1.8
>t(mb)
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9
>mb*ma
     [,1] [,2] [,3]
[1,]    1    8   21
[2,]    8   25   48
[3,]   21   48   81
> 1/mb
[,1]       [,2]       [,3]
```

```
[1,] 1.0000000 0.2500000 0.1428571
[2,] 0.5000000 0.2000000 0.1250000
[3,] 0.3333333 0.1666667 0.1111111
```

Большим преимуществом является возможность чтения данных из файла и сохранения их в файл. Эти файлы могут быть как бинарные, так и текстовые. Для чтения данных из текстового файла используют функцию `read.table()`, её аргументы:

– `file` – имя файла, обязательный аргумент;

– `header` – логический параметр; при значении TRUE считываются имена переменных из файла;

– `sep` – разделитель полей; по умолчанию – пробел;

– `quote` – вид кавычек (двойные или одинарные);

– `dec` – десятичный разделитель в числах (точка или запятая);

– `row.names` – вектор имён строк; представляет собой либо вектор с именами строк итоговой таблицы; либо число – номер столбца исходной таблицы с названиями строк; либо имя столбца считываемой таблицы, где приведены названия строк; если этот параметр не задан, то строки в итоговой таблице будут пронумерованы;

– `col.names` – вектор имён столбцов в итоговой таблице;

– `as.is` – нужно ли символьные переменные, непреобразованные в числовые или логические, переводить в факторы;

– `colClasses` – символьный вектор; определяет классы данных в столбцах (символьные, логические, числовые, даты). Возможные значения: `NA` – автоматическая конвертация типов данных, NULL – столбец пропускается (данные не преобразовываются), тип данных в который будут переведены элементы столбца, factor;

– `na.strings` – символьный вектор, элементы которого при чтении исходной таблицы в файле будут интерпретироваться как NA;

– `nrows` – целочисленный аргумент; определяет максимальное число считываемых строк;

– `skip` – положительный целочисленный аргумент; определяет число строк, пропускаемых перед чтением;

– `check.names` – логический аргумент; при значении TRUE имена переменных будут проверены на синтаксическую правильность и отсутствие дублирования;

– `fill` – логический аргумент; при значении TRUE строки разной длины будут приведены к единой (максимальной), путем добавления пустых полей.

### *Упражнение 10*

1) Создайте в **Блокноте** файл со следующим содержимым:
a;b;c

1;2;3
4;5;6
7;8;9

2) Сохраните его в рабочей папке под именем mydata.txt. Закройте **Блокнот**.

3) Командой `file.show("mydata.txt")` просмотрите содержимое файла.

4) Закройте окно «RInformation».

5) Считаем данные из этого файла в таблицу данных `d2` командой

```
>d2 = read.table("mydata.txt", sep =";", head=TRUE)
```

6) Командой `d2`просмотрите данные этой таблицы.

7) Командой просмотрите информацию об этой таблице данных.

Результаты выполнения этого упражнения:

```
>file.show("mydata.txt")
> d2=read.table("mydata.txt", sep=";", head=TRUE)
Warning message:
In read.table("mydata.txt",sep = ";", head = TRUE):
incomplete final line found by readTableHeader on
'mydata.txt'
> d2
a b c
1 1 2 3
2 4 5 6
3 7 8 9
>str(d2)
'data.frame':   3 obs. of  3 variables:
 $ a: int  1 4 7
 $ b: int  2 5 8
 $ c: int  3 6 9
```

Создадим простой скрипт и отправим его на исполнение.

### *Упражнение 11*

1) В командном меню подайте команду **File->Newscript (Ctrl+N)**.

2) В открывшемся окне «REditor» наберите текст скрипта в соответствии с рисунком 1.3.

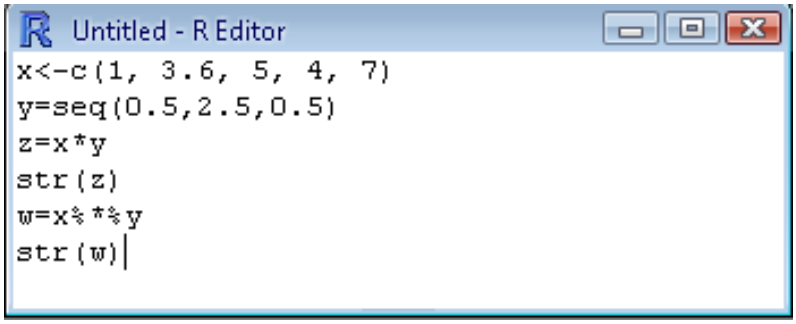



Рисунок 1.3 – Окно «REditor»

3) Отправьте его на исполнение. Результаты его работы приведены ниже:

```
> x<-c(1, 3.6, 5, 4, 7)
> y=seq(0.5,2.5,0.5)
> z=x*y
>str(z)
num [1:5] 0.5 3.6 7.5 8 17.5
> w=x%*%y
>str(w)
num [1, 1] 37.1
```

4) Сохраните его под именем **script1**.

Стоит пояснить, что операция `%*%` отвечает за скалярное произведение векторов.

Создадим новый скрипт и запишем его как текстовый файл.

***Упражнение 12***

1) Откройте приложение **Блокнот**.

2) Запишите в нем текст программы согласно рисунку 1.4.

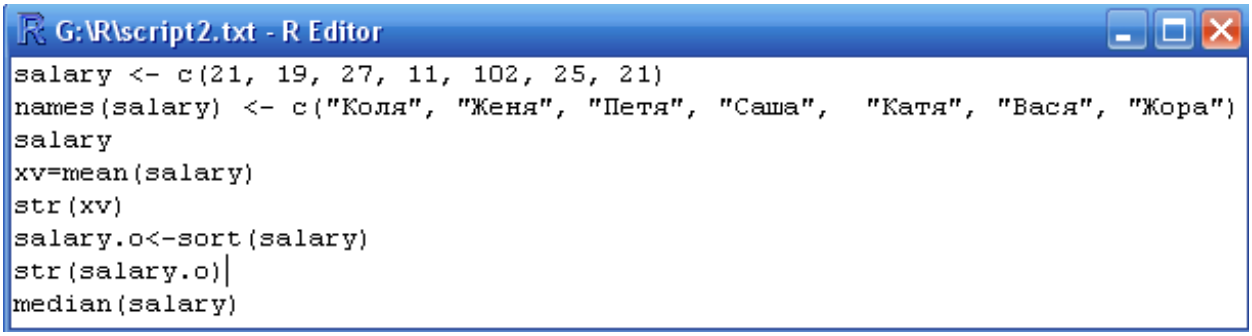



Рисунок 1.4 – Тест программы нового скрипта **script2**

3) Сохраните этот документ под именем **script2**.

4) Закройте **Блокнот**.

5) В командной строке подайте команду **File->Openscript…(Ctrl+O)**.

6) В открывшем окне «Openscript»выберите тип файла «Allfiles (*.*)» и укажите файл **script2.txt**, откройте его.

7) Отправьте скрипт на исполнение.

Результаты исполнения скрипта будут следующими:

```
>salary<- c(21, 19, 27, 11, 102, 25, 21)
>names(salary)<-c("Коля","Женя","Петя","Саша",
+"Катя","Вася","Жора")
>salary
Коля Женя Петя Саша Катя Вася Жора
21   19   27   11  102   25   21
>xv=mean(salary)
>str(xv)
```

```
num 32.3
>salary.o<-sort(salary)
>str(salary.o)
 Named num [1:7] 11 19 21 21 25 27 102
 - attr(*, "names")= chr [1:7] "Саша" "Женя" "Коля"
"Жора" ...
>median(salary)
[1] 21
```

Следует пояснить действие некоторых команд. Команда `mean()` вычисляет среднее выборочное, `median()` – медиану выборки, команда `sort()` сортирует элементы вектора по возрастанию.

## 1.3 Самостоятельная работа

1) Создайте в своей директории файл, содержащий вашу выборку. Данные заполняйте в один столбец.

2) Считайте эти данные из файла в новую переменную `y`.

3) Просмотрите информацию о ней.

4) Создайте две матрицы:

$$AB=\begin{matrix}2 & -3 & 4\\ 5 & -1 & 6\end{matrix} \text{ и } BA=\begin{matrix}0.5 & -1\\ 6 & 7\\ 8 & 9\end{matrix}.$$

5) Вычислите $5BA+4AB^{T}$.

6) Напишите скрипт, который:

a) создает числовую последовательность g=1, 1/2, 1/3, 1/4, 1/5, 1/6, 1/7, 1/8, 1/9;

b) преобразует эту последовательность g в матрицуB размерности 3×3;

c) транспонирует эту матрицу.

7) Покажите работу скрипта преподавателю.

## 1.4 Контрольные вопросы

1) Какие типы данных используются в R?

2) При помощи каких операторов в R переменной присваивается числовые значения?

3) Чем отличается вектор от матрицы?

4) Какими функциями вектор преобразуется в матрицу?

5) Что такое скрипт?

6) Чем отличается произведение матриц в R от произведения матриц в алгебре?

7) Опишите синтаксис обращения к элементу матрицы.

8) Дайте определения таблиц данных.

9) При помощи какого оператора таблица данных  сохраняется в файл?

10) При помощи какого оператора таблица данных читается из файла?

# Лабораторная работа № 2
# Простейшие операции. Операторы цикла и условия. Создание собственных функций

**Цель работы**: простейшие операции и встроенные функции, изучить операторы цикла и условия, научиться создавать свои функции.

**Оборудование**: ПК, оснащенный необходимым программным обеспечением.

**Объём лабораторной работы и порядок её выполнения**. Лабораторная работа выполняется в течение двух академических часов. Изучите теоретическую часть, выполните работу и упражнения для самостоятельной работы.

## 2.1 Теоретическая часть
## Логические операции

Логические операции в R представлены в таблице 2.1. Результатом их выполнения будет значение (истина), или (ложь).

Таблица 2.1 − Логические операции

| Логическая операция | Описание |
|---|---|
| !*x* | инверсия (отрицание) |
| *x*&*y* | поэлементная конъюнкция (логическое умножение, логическое **И**) |
| *x*&&*y* | конъюнкция только для первых элементов |
| *x*\|*y* | поэлементная дизъюнкция (логическое сложение, логическое умножение **ИЛИ**) |
| *x*\|\|*y* | дизъюнкция только для первых элементов |
| *xor*(*x*; *y*) | если *x* и *y* различны, то получим `TRUE`, иначе `FALSE` |

Ниже приведен пример использования логических операций.

```
>z<-FALSE
>z
[1] FALSE
> x<-1>2
>x
[1] FALSE
> !x
[1] TRUE
> y=!x
>y&x
[1] FALSE
>y|x
[1] TRUE
```

## Математические функции

В таблице 2.2 приведены математические функции, реализованные в R.Следует обратить свое внимание на то, что аргументы тригонометрических функций задаются в радианах, а не в градусах. Результаты вычисления значений обратных тригонометрических функции приводятся также в радианах.

Таблица 2.2 – Элементарные математические функции

| Функция в R | Описание |
|---|---|
| `exp(x)` | $e^x$ |
| `log(x)`и`logb(x)` | $\ln x$ |
| `log10(x)` | $\lg x$ |
| `log2(x)` | $\log_2 x$ |
| `log(x, base=)` | вычисление логарифма по произвольному основанию |
| `abs(x)` | $\|x\|$ |
| `sqrt(x)` | $\sqrt{x}$ |
| `sin(x)` | sin $x$(значение $x$ задают в радианах) |
| `cos(x)` | cos $x$(значение $x$ задают в радианах) |
| `tan(x)` | tg $x$($x\neq\pi/2+k\pi$, k=0, ±1, ±2,...) |
| `asin(x), acos(x)` | arcsin $x$, arcos $x$($\|x\|\leq 1$) |
| `atan(x)` | arctg $x$ |
| `factorial(x)` | $x!$ |

Ниже приведены примеры вычислений тригонометрических функций.

```
>asin(4)
[1] NaN
Warning message:
In asin(4) : NaNs produced
>asin(1)
[1] 1.570796
>acos(1)
[1] 0
>acos(0.5)
[1] 1.047198
>acos(5)
[1] NaN
Warning message:
In acos(5) : NaNs produced
>atan(pi/2)
[1] 1.003885
>tan(pi/2)
[1] 1.633178e+16
```

```
>tan(3*pi/2)
[1] 5.443926e+15
>sqrt(-4)
[1] NaN
Warningmessage:
Insqrt(-4) : NaNsproduced
```

Следует дать небольшое пояснение по объекту `NaN`. Таких специальных объектов в R четыре. Каждый из них означает следующее:

`Inf`– это бесконечность∞;

`NA`– «отсутствующее значение» (Not Available);

`NaN`– нечисло (Not a Number);

`NULL`– «ничто».

***Замечание.*** *Гиперболические и специальные функции не будут рассмотрены в этом практикуме, поэтому их заинтересованным обучающимся придется разбирать самостоятельно*.

**Функции округления**

В доступны следующие функции округления:

– `ceiling(x)` – округляет до минимального целого не меньшего *x*;

– `floor(x)` – возвращает максимальное целое, не превосходящее *x*;

– `trunc(x)` – отбрасывает дробную часть *x*;

– `round(x)` – округляет до ближайшего целого *x*;

– `round(x, dig)` – основная форма функции, параметр показывает число цифр после запятой, до которой нужно провести округление;

– `signif(x, digits = 6)` – округляет *x* до заданного числа (`digits`) значимых цифр, то есть отличных от нуля (по умолчанию параметр `digits` равен шести).

–

**Оператор `if`.**

Кратный синтаксис оператора `if` следующий:

```
if(условие) выражение
```

`условие` – любой оператор условия (<, >, >=, <=, ==, !=), результатом выполнения которого является логический вектор единичной длины. Если значение вектора `TRUE`, то выполняется выражение.

`Выражение` – одно или несколько выражений, выполняемых в случае верности условия. Если задано несколько выражений, то они должны быть заключены в фигурные скобки `{}` и разделяться точкой с запятой (если на одной строке).

Оператор возвращает значение `выражения` в случае верности условия, или ничего не возвращает (NULL).

Пример использования оператора `if` приведен ниже.

```
> x<-4;y<-5
>if (x<y) {z=x+y;z}
[1] 9
>if (x>y) w=x-y
>w
Error: object 'w' not found
```

Полный синтаксис оператора `if` следующий:

```
if(условие) {выражение1} else{выражение2}
```

Если `условие` выполняется, то вычисляется `выражение1`, в противном случае `выражение2`. Пример использования этого оператора приведен ниже.

```
>x=5;y=4
>if(x<y) {x+y} else {x-y}
[1] 1
```

### Оператор цикла `for`

Синтаксис этого оператора следующий:

```
for(переменная in последовательность) выражение
```

`Выражение` выполняется, пока значение `переменной` изменяется в пределах `последовательности`. Пример применения оператора цикла для создания гармонического ряда приведен ниже.

```
> t=vector(length=10,mode='numeric')
>for(k in 1:10)
+ t[k]=1/k
>t
 [1] 1.0000000 0.5000000 0.3333333 0.2500000
0.2000000 [8] 0.1666667 0.1428571 0.1250000 0.1111111
0.1000000
```

### Создание функции пользователя

Несмотря на большое количество встроенных функций пользователю бывает необходимо создать свои функции. Синтаксис создания функции в R пользователя:

```
function(аргументы) {выражение},
```

где

- `function` – это служебное слово, которое сообщает R о создании функции;
- `аргументы` – список аргументов функции;
- `выражение` (*тело функции*) представляет собой команду, или блок команд, зависящих от аргументов функции.

Функция в качестве своего значения возвращает результат последнего выражения. Рассмотрим пример функции, вычисляющей норму – корень квадратный от скалярного произведения двух векторов.

```
>norm<-function(x,y) {sqrt(x%*%y)}
```

```
> y<-c(2,1.5,2,9,5.6,7,9,4)
> x=c(1,3,1,5,1,7,1,9)
>norm(x,y)
[,1]
[1,] 12.37336
```

**Полная форма функции в R**

Синтаксис полной формы записи функции пользователя в R:

```
> имя = function(аргументы) {выражение
return(значение)
}
```

где `return (значение)` – возвращаемое функцией значение (вектор единичной длины, в противном случае будет выведено предупреждение).

Например, функция вычисления начального момента второго порядка будет иметь следующий синтаксис:

```
> nu2=function(x){x2<-x^2;sx<-sum(x2);
+ n=length(x);x2<-sx/n
+ return(x2)}
>str(x.f)
int [1:9] 1 6 8 2 9 3 4 7 5
>nu2(x.f)
[1] 31.66667
```

Напомним формула вычисления начального момента второго порядка:

$$\nu_2 = \frac{1}{n}\sum_{k=1}^{n} x_k^2.$$

## 2.2 Выполнение лабораторной работы

Рассмотрим как работает функция `factorial(x)`.

### *Упражнение 1*

1) Наберите в командной строке команду `n<-4;n` и отправьте её на исполнение, нажав клавишу <Enter>.

2) Вычислите *n*!, отправив на исполнение в командной строке функцию `factorial(n)`.

3) Присвоим переменной *n* значение 0,5 и выведем её значение на экран командой `n<-0.5;n`.

4) Вычислим значение факториала этой функции при помощи функции `factorial(n)`.

5) Присвоим переменной *n* значение 1,5 и выведем её значение на экран командой `n<-1.5;n`.

6) Присвоим переменной *f* значение факториала переменной *n* командой `f=factorial(n)`.

7) Просмотрим её значение командой `str(f)`.

Результаты работы команд будут следующими:

```
> n<-4;n
[1] 4
>factorial(n)
[1] 24
> n<-0.5;n
[1] 0.5
>factorial(n)
[1] 0.8862269
> n<-1.5;n
[1] 1.5
> f=factorial(n)
>str(f)
num 1.33
```

Рассмотрим работу функций округления. Начнем с функции `ceiling()`.

### *Упражнение 2*

1) Наберите в командной строке команду `ceiling(1.8)` и отправьте её на исполнение, нажав клавишу <Enter>.

2) Наберите в командной строке команду `ceiling(1.2)` и отправьте её на исполнение, нажав клавишу <Enter>. Обратите внимание, что результаты округления совпадают.

3) Наберите в командной строке команду `ceiling(-1.2)` и отправьте её на исполнение, нажав клавишу <Enter>.

4) Наберите в командной строке команду `ceiling(-1.8)` и отправьте её на исполнение, нажав клавишу <Enter>. Обратите внимание на то, что последние два результата округления совпадают по значению.

Результаты работы команд будут следующими:

```
>ceiling(1.8)
[1] 2
>ceiling(1.2)
[1] 2
>ceiling(-1.2)
[1] -1
>ceiling(-1.8)
[1] -1
```

Эту функцию удобно будет использовать при вычислении количества интервалов *k*-интервального вариационного ряда, например, по формуле Стерджесса:

```
k=ceiling(1+3.2*log(n)).
```

Рассмотрим функцию `floor()`.

### *Упражнение 3*

1) Наберите в командной строке команду `floor(1.8)` и отправьте её на исполнение, нажав клавишу <Enter>.

2) Наберите в командной строке команду `floor(1.2)` и отправьте её на исполнение, нажав клавишу <Enter>. Обратите внимание, что результаты округления совпадают.

3) Наберите в командной строке команду `floor(-1.2)` и отправьте её на исполнение, нажав клавишу <Enter>.

4) Наберите в командной строке команду `floor(-1.8)` и отправьте её на исполнение, нажав клавишу <Enter>. Обратите внимание на то, что последние два результата округления совпадают по значению.

Результаты работы команд будут следующими:

```
>floor(1.8)
[1] 1
>floor(1.1)
[1] 1
>floor(-1.1)
[1] -2
>floor(-1.8)
[1] -2
```

Рассмотрим функции округления `trunc()` и `round()`.

### *Упражнение 4*

1) Наберите в командной строке команду `trunc(-1.5)` и отправьте её на исполнение, нажав клавишу <Enter>.

2) Наберите в командной строке команду `trunc(1.5)` и отправьте её на исполнение, нажав клавишу <Enter>. Обратите внимание, что в обоих примерах эта функция просто отбросила дробную часть.

3) Применим для округления этих чисел функцию `round()`. Наберите в командной строке сначала функцию `round(-1,5)`, а затем – `round(1,5)`.

4) Округлите при помощи этой же функции числа 1,2 и -1,2.

Результаты работы команд будут следующими:

```
>trunc(-1.5)
[1] -1
>trunc(1.5)
[1] 1
>round(-1.5)
[1] -2
```

```
>round(1.5)
[1] 2
>round(1.2)
[1] 1
>round(-1.2)
[1] -1
```

Используем для округления значения числовой константы `pi`, соответствующей действительной константе π≈3.14159265358979. Понаблюдаем отличие функций округления `round()` и `signif()`.

### *Упражнение 5*

1) Округлим константу π при помощи функции `round(pi)`.

2) Округлим до одной тысячной при помощи функции `round(pi,3)`. В ответе приведена дробная часть из трех цифр.

3) Округлим до одной стотысячной при помощи функции `round(pi,5)`. В ответе приведена дробная часть из пяти цифр.

4) Проведем округление функцией `signif()`. В командной строке наберите `signif(pi,3)` и её на исполнение. В ответе приведены три цифры: одна в целой части числа, а две в дробной части.

5) Отправьте последовательно на исполнение функции `signif(pi,5)` и `signif(pi*100,5)`.Результатом выполнения этих функций будет число из пяти цифр, но распределение этих цифр между целой и дробной частями для каждого числа будет разным.

6) Результатом исполнения функции `round(pi*100,4)` будет число, дробная часть которого состоит из четырех цифр.

Результаты работы команд будут следующими:

```
>round(pi)
[1] 3
>round(pi,3)
[1] 3.142
>round(pi,5)
[1] 3.14159
>signif(pi,3)
[1] 3.14
>signif(pi,5)
[1] 3.1416
>signif(pi*100,5)
[1] 314.16
>round(pi*100,4)
[1] 314.1593
```

Пусть даны два вектора *x* и *y* одной размерности. Попытаемся создать новый вектор *z*, элемента которого равны

$$z_j = \begin{cases} \dfrac{x_j}{y_j}, \text{если } x_j > y_j; \\ x_j \cdot y_j, \text{если } x_j \le y_j. \end{cases}$$

### *Упражнение 6*

1) Создадим вектор `x` командой `x=c(1,3,1,5,1,7,1,9).`
2) Создадим вектор `y` командой `y<-c(2,1.5,2,9,5.6,7,9,4).`
3) Создадим команду `if (x>y) {z=x/y} else {z=x*y}.`
4) Проверим результат командой `z`.

Результаты работы команд будут следующими:

```
> x=c(1,3,1,5,1,7,1,9)
> y<-c(2,1.5,2,9,5.6,7,9,4)
>if (x>y) {z=x/y} else {z=x*y}
Warning message:
In if (x > y) { :
the condition has length > 1 and only the first el-
ement will be used
>z
[1]  2.0  4.5  2.0 45.0  5.6 49.0  9.0 36.0
```

Причина того, что результат получился не тот, заключается d следующем. При выполнении `x>y` сравнение происходит только первых элементов. Воспользуемся оператором `for`.

### *Упражнение 7*

1) Командами `str(x)` и `str(y)` проверим наличие и совпадение по размерности векторов `x` и `y`.
2) Постройте цикл командой

```
>for(k in 1:8)
+ if(x[k]>y[k]) {z[k]=x[k]/y[k]} else
+ {z[k]=x[k]*y[k]}
```

3) Проверьте результат вычислений командами `z` и `str(z).`

Результаты работы команд будут следующими:

```
>str(x)
num [1:8] 1 3 1 5 1 7 1 9
>str(y)
num [1:8] 2 1.5 2 9 5.6 7 9 4
>for(k in 1:8)
+ if(x[k]>y[k]) {z[k]=x[k]/y[k]} else
+ {z[k]=x[k]*y[k]}
>z
[1]  2.00  2.00  2.00 45.00  5.60 49.00  9.00  2.25
>str(z)
```

```
num [1:8] 2 2 2 45 5.6 49 9 2.25
```

Создадим функцию пользователя, вычисляющую выборочную дисперсию

$$D\ X\ = \frac{1}{n}\sum_{k=1}^{n} x_i - x_{\text{в}}\ ^2.$$

Будем использовать для этого функцию `mean()`, вычисляющую среднее арифметическое вектора $a = \{a_1; a_2; \ldots; a_n\}$. Таким образом,

$$mean\ a\ = \frac{1}{n}\sum_{i=1}^{n} a_i.$$

### *Упражнение 8*

1) Создадим функцию пользователя:

```
disp<-function(x) {xv=mean(x);x1=(x-xv)^2;mean(x1)}
```

2) Используем вектор `x`. Проверим его наличие командой `str(x)`. Если его то создайте командой `x=c(1,3,1,5,1,7,1,9)`.

3) Используем эту функцию `disp(x)`.

4) Вычислим уточненное значение дисперсии $s^2 = n/(n\text{-}1)\text{D}[X]$ командой `disp(x)*8/7`

5) Проверим правильность вычислений встроенной командой `var(x)`.

Результаты работы команд будут следующими:

```
>disp<-function(x){xv=mean(x);x1=(x-xv)^2;mean(x1)}
>str(x)
num [1:8] 1 3 1 5 1 7 1 9
>disp(x)
[1] 8.75
>disp(x)*8/7
[1] 10
>var(x)
[1] 10
```

### *Упражнение 9*

1) Найдите для вектора `x` среднее квадратичное отклонение и уточненное его значение.

2) Напишите функцию, вычисляющую среднюю выборочную по формуле

$$x_{\text{в}} = \frac{1}{n}\sum_{k=1}^{n} x_k.$$

3) Вычислите среднюю выборочную для вектора `x`.

4) Напишите функцию для вычисления выборочной дисперсии по формуле

$$D[X]=\nu_2-(\nu_1)^2.$$

Создайте функцию пользователя, вычисляющую нижнюю квартиль вектора *x*, в полной форме.

### *Упражнение 10*

1) Проверьте командой `str(x)` наличие вектора. Если его нет, то введите командой `x<-c(1,3,1,5,1,7,1,9)`.

2) Введите начало команды для формирования функции пользователя `nkv=function(x){` и для перехода на новую строку одновременно нажмите клавиши <Ctrl>и <Enter>.

3) Введите `n=length(x);k<-trunc(n/4)+1;kv<-x[k]` и для перехода на новую строку одновременно нажмите клавиши <Ctrl>и <Enter>.

4) Введите `return(kv)}` и для завершения ввода функции нажмите клавишу<Enter>.

5) Командой `>nkv(x)` вычислим нижнюю квартиль вектора `x`

```
>str(x)
num [1:8] 1 3 1 5 1 7 1 9
>nkv=function(x){
+ n=length(x);k<-trunc(n/4)+1;kv<-x[k]
+ return(kv)}
>nkv(x)
[1] 1
```

## 2.3 Самостоятельная работа

1) Создает функцию для вычисления третьего центрального момента

$$\mu_3=\frac{1}{n}\sum_{k=1}^{n}\left(x_k-x_{\text{в}}\right)^3.$$

2) Создает функцию для вычисления четвертого центрального момента

$$\mu_4=\frac{1}{n}\sum_{k=1}^{n}\left(x_k-x_{\text{в}}\right)^4.$$

3) Используя эти функции, напишите функции для вычисления асимметрии и эксцесса.

$$a_s=\mu_3/\sigma^3,\ e_к=\mu_4/\sigma^4.$$

4) Напишите функцию пользователя в полной форме, вычисляющую верхнюю квартиль.

5) Сохраните эти функции в качестве скрипта.

6) Проверьте его работу для своего варианта данных.

## 2.4 Контрольные вопросы

1) Какие логические операции использовались в R?

2) Какие математические функции используются в R?

3) В каких единицах измеряются аргументы тригонометрических функций в R?

4) Какие специальные величины используются в R?

5) Какие функции округления применяются в R?

6) Какой оператор применяется в R для описания разветвляющегося алгоритма?

7) Какой оператор применяется в R для описания циклического алгоритма?

8) Опишите синтаксис создания функции пользователя в R.

9) Опишите синтаксис полной формы функции пользователя в R.

# Лабораторная работа № 3
# Базовая графика в R

**Цель работы**: научиться строить различные типы графиков и изучить команды и опции их форматирования.

**Оборудование**: ПК, оснащенный необходимым программным обеспечением.

**Объём лабораторной работы и порядок её выполнения**. Лабораторная работа выполняется в течение двух академических часов. Изучите теоретическую часть, выполните работу и упражнения для самостоятельной работы.

## 3.1 Теоретическая часть

В R много графических пакетов. В базовой комплектации за построение графики отвечает пакет **graphics**. Функции, определенные в нем разбиты на три уровня:

- функции высокого уровня;
- функции низкого уровня;
- интерактивные функции.

К функциям высокого уровня относятся функции:

- `par()` – создание графического окна с заданными параметрами
- `barplot()` – построение мозаичной диаграммы (статистика);
- `boxplot()` – построение boxplot – особой диаграммы для статистического анализа данных относительно предполагаемого распределения (статистика);
- `boxplot.matrix()` – аналогично `boxplot`;
- `bxp()` – аналогично `boxplot`;
- `contour()` –построение графика с контурными линиями (линиями уровня);
- `curve()` – построение линии (кривой);
- `dotchart()` – точечные диаграммы (статистика);
- `frame()` – задание нового графического окна;
- `plot.new()` – задание нового графического окна;
- `hist()` – построение гистограммы (статистика);
- `image()` – построение цветной прямоугольной сетки согласно заданным цветам;
- `matplot()` – изображение элементов столбцов одной матрицы относительно элементов соответствующих столбцов другой матрицы;
- `mosaicplot()` – построение мозаичной диаграммы (статистика);
- `persp()` – построение трёхмерных графиков;

– `pie()` – построение круговой диаграммы;

– `plot()` – основная функция построения двумерных графиков;

– `plot.data.frame()` – графический анализ таблиц данных;

– `plot.default()` – построение диаграммы рассеивания;

– `plot.factor()` – построение диаграммы рассеивания для факторов;

– `plot.formula()` – построение диаграммы рассеивания с помощью структуры `formula`;

– `plot.histogram()` – построение гистограммы (статистика);

– `plot.table()` – графический анализ таблицы данных (построение мозаичных диаграмм);

– `screen` – управление графическим окном;

– `stem()` – построение древесной диаграммы (статистика);

– `stripchart()` – построение диаграммы рассеивания (аналог boxplot())

Функция `par()` зависит от 70 аргументов, но пользователь ***не может изменять значения только пяти аргументов***:

– `cin` – размер (ширина, высота) символов в дюймах;

– `cra` – размер (ширина, высота) символов в растрах (пикселях) (некоторые графические устройства устанавливают произвольный размер пикселя, обычно около 1/72 дюйма);

– `csi` – высота символа в дюймах;

– `cxy` – размер символа в пользовательских единицах (единицы размера, определённые пользователем;

– `din` – размеры графического устройства в дюймах.

Оставшиеся 65 аргументы могут быть заданы только в функции par(). В этой работе мы рассмотрим только два параметра `mfcol` и `mfrow`. Оба представляют собой числовые вектора вида `c(nr, nc)`. В этом случае функция разбивает графическое окно на `nr×nc` подокна, в которых будут выведены изображения. Порядок заполнения по столбцам и строкам, соответственно (при этом уменьшаются размеры выводимого в изображениях текста).

Переход из окна в другое окно при построении графиков осуществляется автоматически.

Основной функцией для построения двумерных графиков является `plot(x, y, …)`.

Аргументы функции:

- *x* – координаты точки на графике.
- *y* – координаты по оси y, если заданы соответствующие значения по оси x.
- ... – дополнительные аргументы. К ним относятся:

– `type` – символьный аргумент, определяющий тип построения графика. Возможные значения: ″p″– точки (круги), ″l″ – линии, ″b″ – линии и точки, ″c″– строятся только линии из ″b″, ″o″– точки и линии пересекаются, ″h″ – гистограммо-подобные вертикальные линии, ″s″ – ступенчатая линия, ″S″ – другая ступенчатая линия, ″n″ – ничего не строится (но зато определяются области о-осям).

– `log` – символьный аргумент – задаётся название оси (″x″, ″y″ или ″xy″), координаты по которой переводятся в логарифмическую шкалу.

– `Lty` – стиль линии (должен быть задан параметр `type=″l″`). Возможные значения: ″`solid`″ (1) – сплошная линия; ″`blank`″ (0) – отсутствует линия; ″`dashed`″ (2) – штриховая линия; ″`dotdash`″ (4) – штрих-пунктир; ″`dotted`″ (3) – пунктирная линия; ″`longdash`″ (5) – длинный штрих; ″`twodash`″ (6) – двойной штрих.

– `xlab`, `ylab`, `main` и `sub` – символьные переменные – названия осей, основного и дополнительного заголовков графика.

– `col` – символьный или числовой аргумент – цвет графика.

– `bg` – символьная переменная, отвечающая за цвет фона графического окна. По умолчанию цвет белый – `bg=″white″`.

– `col.axis` – символьная переменная – цвет осей. По умолчанию чёрный – ″`black`″.

– `col.main` – символьная переменная – цвет основного заголовка. По умолчанию чёрный –″`black`″.

– `col.sub` – символьная переменная – цвет дополнительного заголовка. По умолчанию цвет чёрный –″`black`″.

– `fg` – символьная переменная – цвет рамки вокруг графика.

– `font.axis`, `font.lab`, `font.main` и `font.sub` – числовые аргументы – тип шрифта для названия осей, основного и дополнительного заголовков. Возможные значения: 1 – простой (по умолчанию), 2 – жирный; 3 – курсив и 4 – жирный курсив.

– `lwd` – положительное целое число – толщина линии; по умолчанию `lwd`=1.

– `xlim` – числовой вектор `c(x1, x2)` – пределы по оси *x*.

– `ylim` – числовой вектор `c(y1, y2)` – пределы по оси *y*.

– `frame` – логический аргумент – нужно ли строить рамку вокруг графика.

### Управление графическим окном

Управление графическим окном (разбиение на несколько подокон) можно осуществлять не только при помощи функции `par()` с её аргументами `mfrow` и `mfcol`, но и при помощи следующих функций:

– `split.screen(figs, screen, erase = TRUE)` – графическое окно разбивается на заданное число окон, каждое из которых рассматривается как отдельное графическое окно; полезно при построении большого количества графиков в одном окне.

– `screen(n = , new = TRUE)` – выбор подокна, в котором будет построен график.

– `erase.screen(n =)` – очистка определённого подокна (закрашивается цветом фона)

– `close.screen(n, all.screens = FALSE)` – удаление всех выбранных подокон.

Опишем аргументы этих функций:

– `figs` – числовой вектор, состоящий из двух элементов – число подокон по строкам и по столбцам. Например, аргумент `figs=c(2,3)` означает, что графическое окно разбивается на 2 строки, в каждой из которых 3 подокна.

– `screen` – номер разбиваемого подокна.

– `erase` – логический аргумент, отвечающий нужно ли очищать данное окно.

– `n` – номер окна, в котором строится график (функция `screen(n)`), либо которое очищается (`erase.screen()`) или закрывается (`close.screen()`).

– `new` – логический аргумент; при значении TRUE графическое окно будет очищено перед построением нового графика.

– `all.screens` – логический аргумент. Он указывает на закрытие всех окон.

## 3.2 Выполнение лабораторной работы

Функция `plot(x, y,…)` строить графическую зависимость переменной `y` от переменной `x`. Вместо многоточия указывают параметры построения. Все разбирать не будем. Остановимся только на параметрах `type`, `lty`, `col`, `lwd`. С каждым из них разберемся, выполняя упражнения.

В первом упражнении построим график функции $y=\cos(2x)$

### *Упражнение 1*

1) Создаем вектор (переменную) $x$ командой `x<-seq(-2*pi,2*pi,0.1)`.

2) Командой `y=cos(2*x)` вычисляем значения функции для всех значений $x$.

3) Построим график функции командой `plot(x,y,type="p")`, используя тип графика «`p`».

4) Построим график функции командой `plot(x,y,type="l")`, используя тип графика «`l`».

5) Построим график функции командой `plot(x,y,type="b")`, используя тип графика «`b`».

6) Построим график функции командой `plot(x,y,type="c")`, используя тип графика «`c`».

7) Построим график функции командой `plot(x,y,type="o")`, используя тип графика «`o`».

8) Построим график функции командой `plot(x,y,type="h")`, используя тип графика «`h`».

9) Построим график функции командой `plot(x,y,type="s")`, используя тип графика «`s`».

10) Построим график функции командой `plot(x,y,type="S")`, используя тип графика «`S`».

Результаты построения приведены на рисунке 3.1.

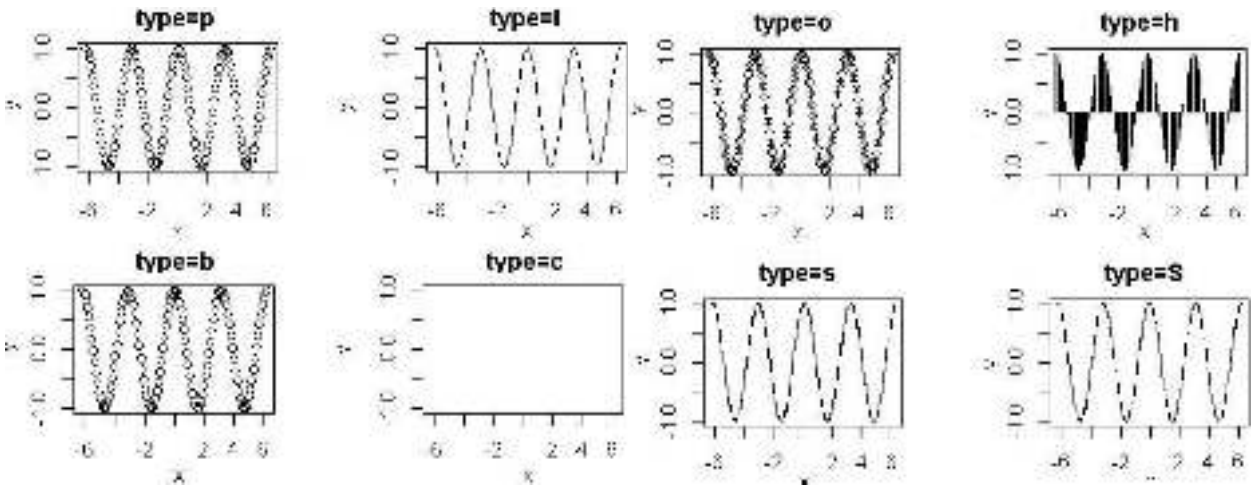



Рисунок 3.1 – Применение различных типов графиков функции `plot()`

Построим круговую диаграмму.

### *Упражнение 2*

1) Командой `salary<- c(21, 19, 27, 11, 102, 25, 21)` создадим переменную, содержащую значения зарплат в тыс. руб.

2) Командой `names(salary) <- c("Коля", "Женя", "Петя", "Саша", "Катя", "Вася", "Жора")` присвоим имена сотрудников соответствующим значениям зарплат.

3) Командой `salary` просмотрим эту переменную.

4) Создайте командой `pie(salary)` круговую диаграмму.

5) Изменим цветовую гамму диаграммы, используя аргумент `col`. Выполните команду `pie(salary, col=2:8)`.

Результаты построения представлены на рисунке 3.2.

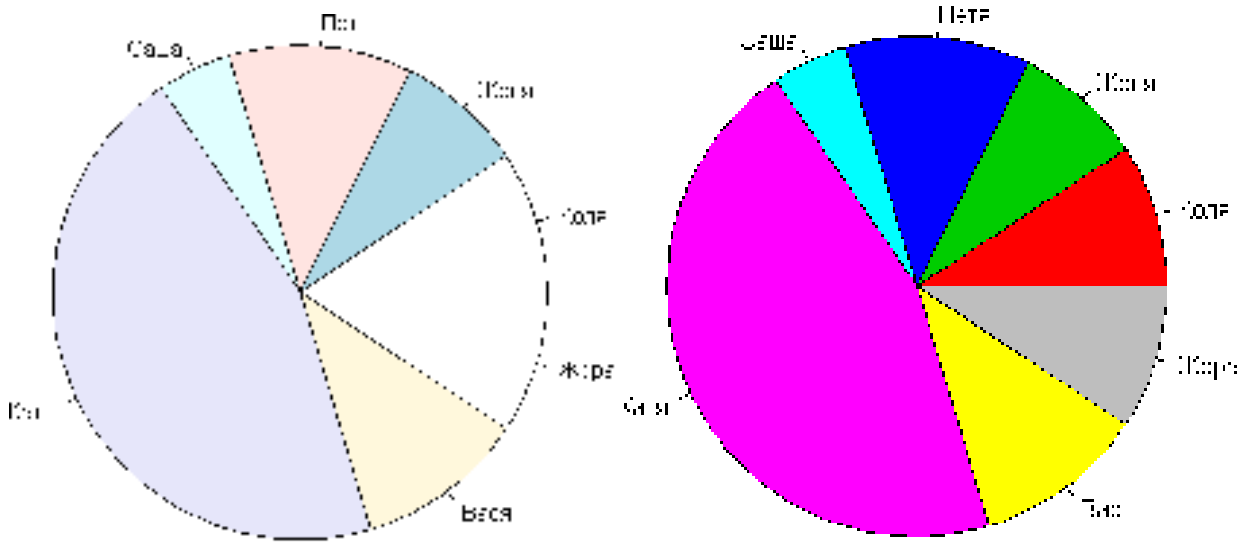


Рисунок 3.2 – Круговые диаграммы

Используя функцию `par()`, разобьем графическое окно на два.

***Упражнение 3***

1) Создадим новое графическое окно командой `par(mfrow=c(2,1))`.

2) Присвоим командой `x=seq(-pi,pi,by=0.1)` переменной (вектору) *x* значения от –π до π с шагом 0,1.

3) Присвоим командой`y=sin(2*x)` переменой *y*значения функции sin 2*x*.

4) Построим в первом окне график функции командой `plot(x,y,type="l",col="brown",sub="sin 2x")`.

5) Построим во втором окне график функции y = sin*x* при помощи команды `plot(sin,-pi,pi,type="l",col="blue",sub="sin()")`.

6) Закройте графическое окно.

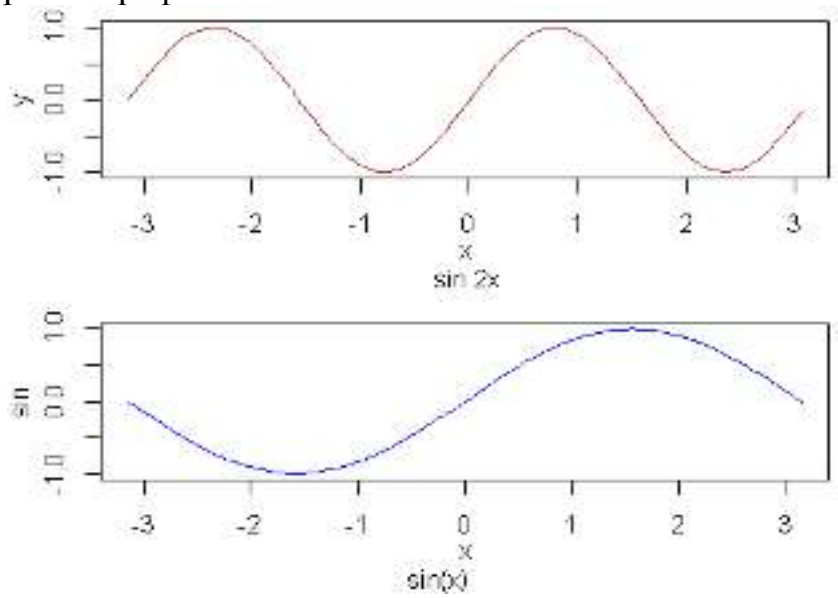



Рисунок 3.3 – Разбиение графического окна функцией `par()`

Результаты построения представлены на рисунке 3.3. Следует обратить внимание на использование аргумента `sub`, выводящего под рисунком его название.

Построим столбиковую диаграмму, обращая внимание на применение аргумента `main`. Первым используем для этого функцию с аргументом `type`, присваивая ему последовательно значения `h`,`s` и `S`.

### *Упражнение 4*

1) Создадим вектор имен командой
```
salary.n<- c("Коля", "Женя", "Петя", "Саша",
+ "Катя", "Вася", "Жора").
```
2) Выполнение команды `plot(salary.n,salary,type="h")` приведет к сообщению об ошибке.

3) Командой `salary.o<-factor(salary.n)` факторизуем вектор имен. Теперь R видит не текстовый вектор, а числовой.

4) Командой `plot(salary.o,salary,type="h")` строим диаграмму.

5) Закройте графическое окно.

6) Командой `plot(salary.o,salary,type="s")` строим диаграмму.

7) Закройте графическое окно.

8) Командой `plot(salary.o,salary,type="S")` строим диаграмму.

Все три диаграммы идентичны и не до конца соответствуют понятию столбиковой диаграммы. Результаты построения приведены на рисунке 3.4.

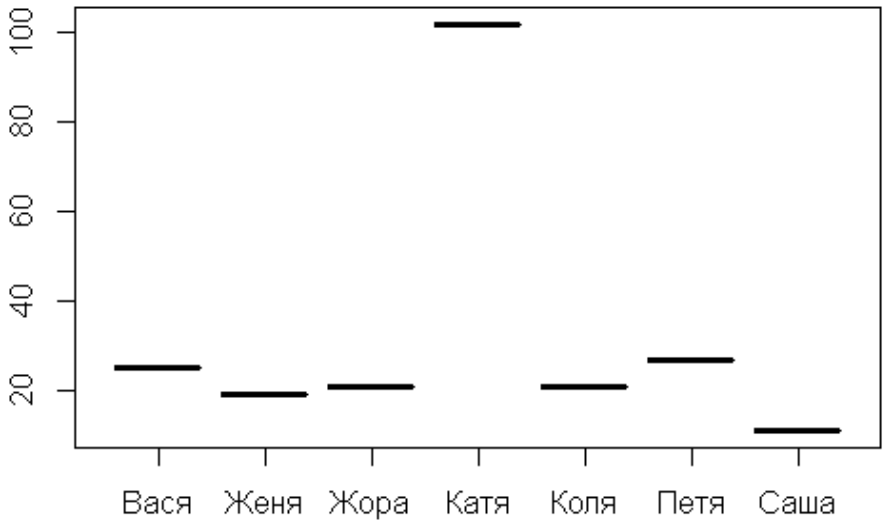



Рисунок 3.4 – Диаграмма зарплат

Построим диаграмму при помощи функции `barplot()`. При построении обратим внимание на аргументы `xlab`, `ylab`, `main`, `col.main` и `font.main`.

### *Упражнение 5*

1) Командой `barplot(salary,col="blue")` постройте диаграмму.

2) Введем команду аргументы `xlab` и `ylab`, отвечающие за подписи осей. Команда должна быть следующей:

```
barplot(salary,col="blue",xlab="Имена сотрудников",
                    + ylab="Зарплата")
```

3) Добавим название диаграммы над диаграммой при помощи аргумента `main`. Изменим цвет названия на красный, и сделаем начертание шрифта курсивом. Синтаксис команды будет следующим:

```
barplot(salary,col="blue",xlab="Имена сотрудников",
   + ylab="Зарплата",main="Зарплата сотрудников",
        + col.main="tomato1",font.main = 3)
```

Результаты построения приведены на рисунке 3.5.

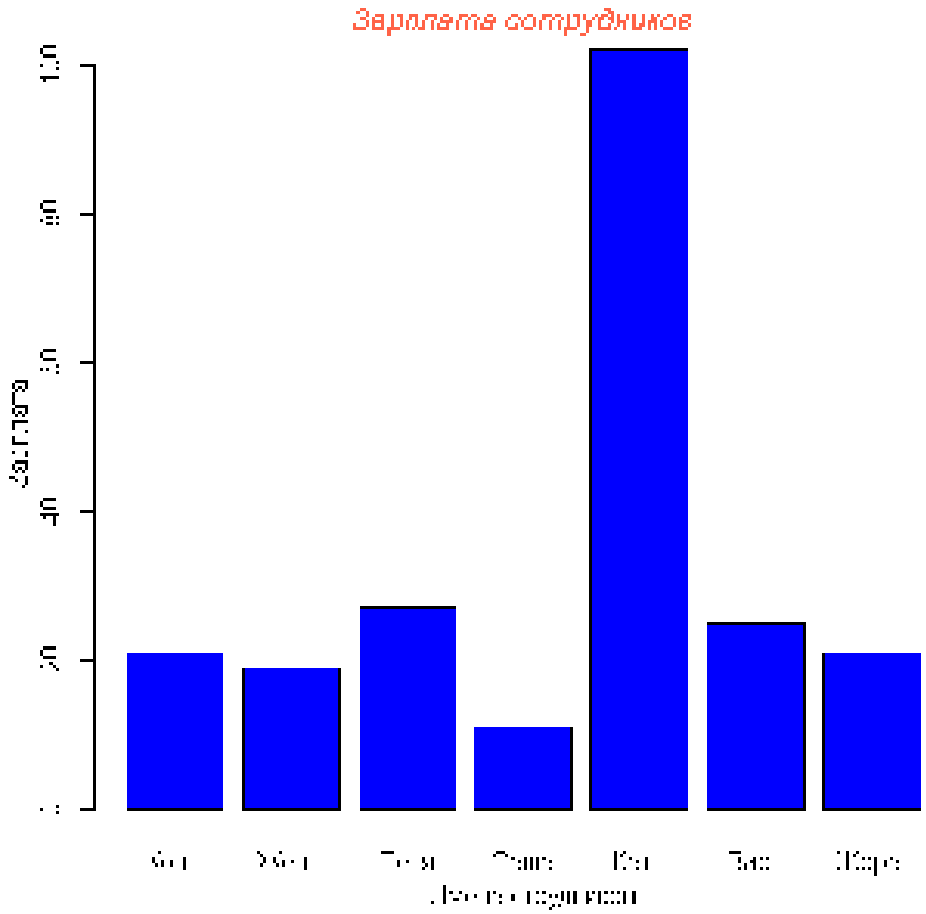



Рисунок 3.5 – Диаграмма `barplot()`

Построим диаграмму при помощи функции `hist()`.

### *Упражнение 6*

1) Командой
`hist(salary,col="yellow",main="Распределение зарплаты")`
постройте гистограмму.

2) Обратите внимание на ось *y*. На ней показано количество людей, получающих зарплату в каждом интервале.

3) Изменим команду, добавив в неё название оси *y*. Синтаксис её приведен ниже.

```
hist(salary,col="yellow",main="Распределение зарплаты",
                + ylab="количество людей")
```

Результаты построения приведены на рисунке 3.6.

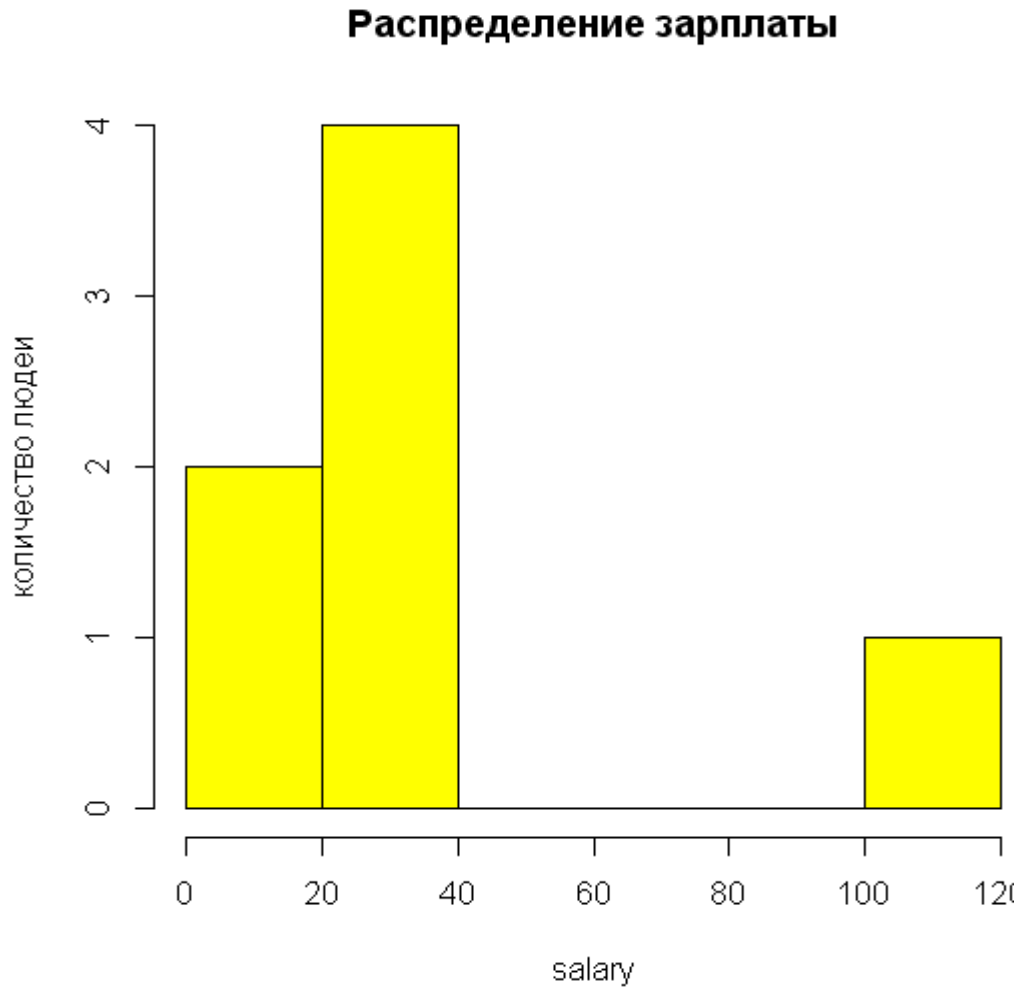



Рисунок 3.6 – Гистограмма

Более подробно с данной функцией разберемся позднее. Построим трехмерный график при помощи функции `persp()`.

### *Упражнение 7*

1) Создадим командой `x<-seq(-10,10,0.5)` вектор значений переменной *x*.

2) Создадим командой `y<-x` вектор значений переменной *y*.

3) Введем функцию, чей график будем строить командой `f <-function(x, y) { r<- sqrt(x^2+y^2); 10 * sin(r)/r}`.

4) Командой `z<-outer(x, y, f)` создадим матрицу, которая ставит в соответствие переменной *z* значения переменных *x* и *y* и функцию *f*(*x*,*y*).

5) Командой `op<- par(bg = "white")` создадим графическое окно с белым фоном.

6) Построим трехмерное изображение функции z=f(x,y) командой `persp(x,y,z,theta=30,phi=30,expand=0.5,col="lightblue")`

7) Изменяя значения аргументов `theta`, `phi` и `expand`, определите как изменение их значений влияет на график.

Результаты построения приведены на рисунке 3.7.

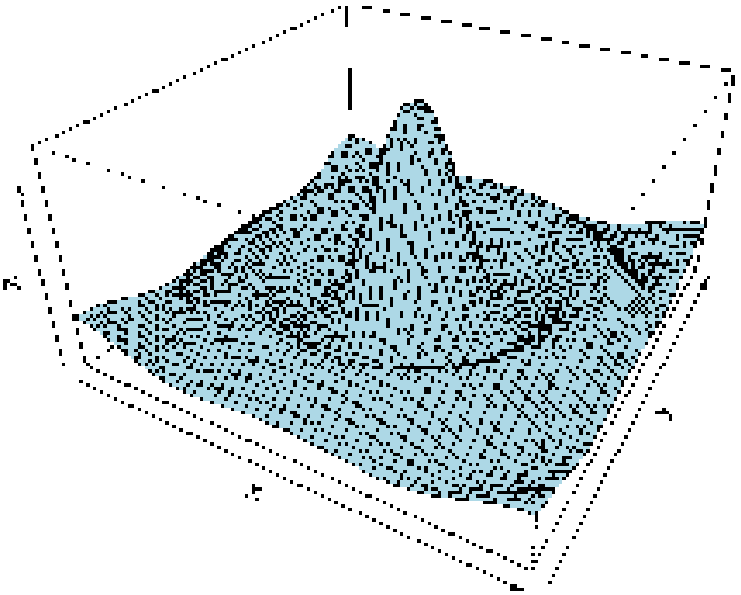


Рисунок 3.7 – Трехмерный график

Следующие упражнения посвящены функциям управления графическим окном. В восьмом упражнении разобьем его на четыре части, в каждом из которых построим график функции.

### *Упражнение 8*

1) Командой `split.screen(c(2,2))` разобьем графическое окно на четыре части.

2) Командой `screen(1)` переместимся в верхнее левое окно.

3) Командой `plot(sin,-pi,pi,col=3)` построим в нем график функции sin *x*.

4) Командой `screen(3)` переместимся в нижнее левое окно.

5) Командой `plot(atan,-2*pi,2*pi,col=5,main="arctg(x)")` построим в нем график функции arctg *x*.

6) Командой `screen(4)` переместимся правее.

7) Командой `plot(exp,-2,1,lwd=2,sub="экспонента")` построим в нем график функции $e^x$.

8) Командой `screen(2)` переместимся в верхнее правое окно.

9) Построим в нем график функции cos *x* командой `plot(cos,-pi,pi,col="gold",lwd=2,ylab="y=cos(3x)")`.

Результаты построения приведены на рисунке 3.8.

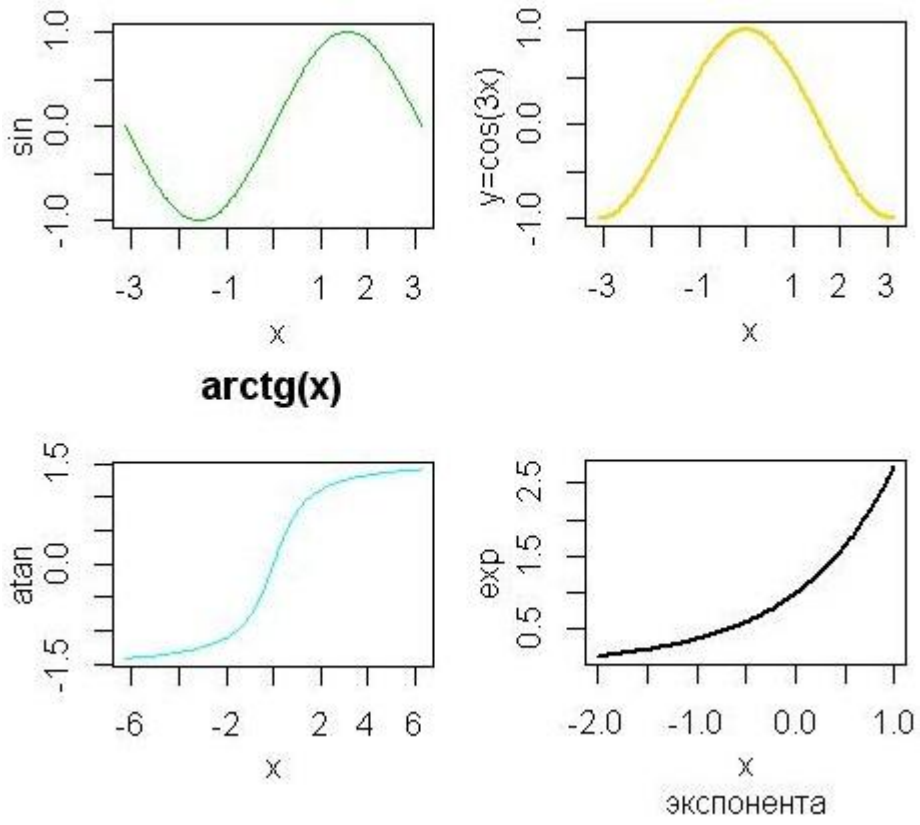



Рисунок 3.8 – Использование функции `screen()`

***Замечание***. К сожалению, команда `erase.screen()` в данной версии системы R не работает.

### *Упражнение 9*

1) Командой `close.screen(all.screens = "TRUE")` закроем все окна.

2) Командой `plot(1:10,ylab=' ')` проверьте это.

3) Закройте графическое окно.

Разберем построение нескольких графиков в графическом окне. Сначала сделаем это при помощи функции `par()`, определив значение её аргумента `new` как `TRUE`.

### *Упражнение 10*

1) Определим значения переменной `x` командой `x<-c(-1,0,1,1.5,2)`.

2) Командой `y<-c(0.8,0.1,1.05,2.3,3.8)` определим значения переменной `y`, полученные экспериментально.

3) Командой `y1<-x^2` вычислим значения функции $y=x^2$.

4) Командой `plot(x,y,lwd=2)` построим график функции по экспериментальным данным, увеличив толщину линий кругов до двух.

5) Командой `par(new='TRUE')` разрешим построение в графическом окне еще один график. Более правильно говорить, что этой командой указываем R, что запрещаем очистку графического окна.

6) Командой `plot(x,y1,type='l',col='blue',lwd=2)` построим график функции $y=x^2$ сплошной линией синего цвета толщиной, равной двум, в этом же графическом окне.

Результаты выполнения десятого упражнения приведены на рисунке 3.9.

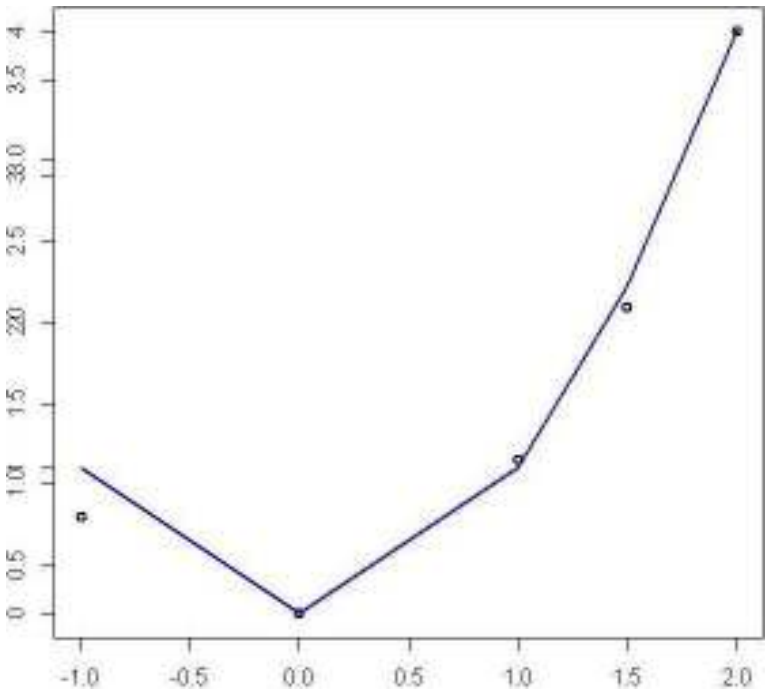



Рисунок 3.9 – Результаты выполнения десятого упражнения

Функция `curve()` строит график заданной функции на некотором интервале. Её синтаксис:

```
curve(expr, from = NULL, to = NULL, n = 101,
add = FALSE, type = "l", ylab = NULL, log = NULL,
                xlim = NULL, ...)
```

Аргументы этой функции:

`expr` – выражение как функция от x или имя функции, график которой нужно построить;

`from` и `to` – числовые аргументы – границы отрезка, на котором строится график. Если не заданы эти параметры, то границы определяются значением аргумента `x` заданного выражения. Если же и `x` не определён, то график строится на отрезке [0;1];

`n` – числовой аргумент – число точек, в которых определяется значение `expr`;

`add` – логический аргумент – нужно ли добавлять строящийся график к уже существующему (`add=TRUE`) или нужно создать новое графическое окно (`add=FALSE`).

Выполним предыдущее упражнение, используя функцию `curve()`.

### *Упражнение 11*

1) Командой `plot(x,y,lwd=2)` построим график функции $y=f(x)$ по экспериментальным данным, увеличив толщину линий кругов до двух.

2) Командой `curve(x^2,-1,2,col='red',lwd=2,add=TRUE)` построим график функции $y = x^2$ сплошной линией красного цвета толщиной, равной двум, в этом же графическом окне.

Результаты выполнения десятого упражнения приведены на рисунке 3.10.

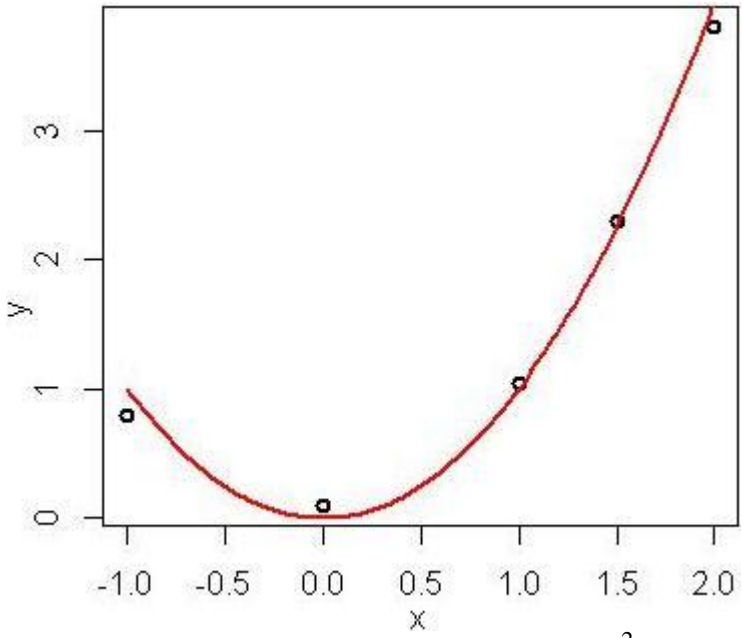



Рисунок 3.10 – Графики $y(x)$ и $x^2$

Построим графики функции sin($x$) и cos($x$) в одном графическом окне, используя функцию `curve()`.

### *Упражнение 12*

1) Командой `curve(sin,-2*pi,2*pi,col='blue',lwd=2)` построим график синего цвета функции sin($x$).

2) Командой `curve(cos,-2*pi,2*pi,col='red',lwd=2, add=TRUE)` построим график синего цвета в том же самом окне.

Результаты выполнения десятого упражнения приведены на рисунке 3.11.

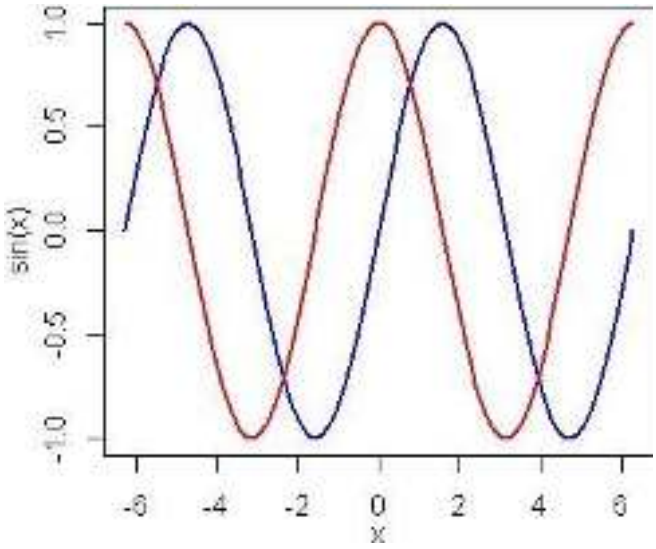



Рисунок 3.11 – Графики функций sin(*x*) и cos(*x*)

Рассмотрим последнюю графическую функцию `windows()`. Она позволяет создавать новое графическое окно, не закрывая предыдущего.

### *Упражнение 13*

1) Командой `x<-seq(0.1,4.1,0.05)` создадим переменную (вектор) `x`.

2) Командой `kn<-length(x)` присвоим переменной `kn` размерность вектора `x`.

3) Создадим вектор `y`, значения которого соответствуют значениям переменной `x` ряда Тейлора функции sin(*x*) командой

```
for(k in 1:kn)y[k]=x[k]-x[k]^3/6+x[k]^5/120-x[k]^7/5040
```

4) Командой `plot (x,y,type='l',lwd=2)` построим график функции *y*(*x*).

5) Командой `windows()` создадим новое графическое окно, не закрывая предыдущего.

6) Командой `curve(sin,0.1,4.1,col='blue', lwd=2)` в новом окне построим график функции sin(*x*).

Результаты выполнения десятого упражнения приведены на рисунке 3.12.

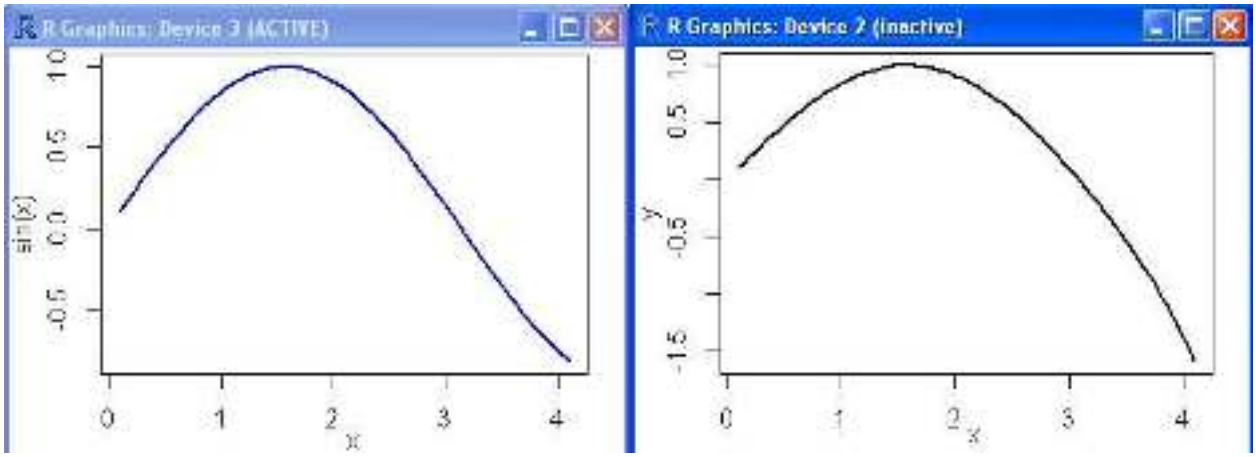



Рисунок 3.12 – Графики функций sin(*x*) и *y*(*x*)

### 3.3 Самостоятельная работа

1) Постройте график функции `y=f(x)`, заданной в приложении 2.

2) Постройте в том же графическом окне график функции *y=f(x)*, заданной таблично в приложении 2.

3) Постройте в новом графическом окне графики функций реализации запчастей 1, 2 и 3 групп за 1999 год по месяцам:

| группа | январь | февраль | март | апрель | май | июнь |
|---|---|---|---|---|---|---|
| I группа | 30350,4 | 30350,4 | 94351,8 | 75810,6 | 57860,4 | 104943,6 |
| II группа | 6525 | 6525 | 73198,8 | 14310 | 29550 | 18716,4 |
| III группа | 10230 | 10230 | 69480 | 18874,8 | 43176,6 | 70900,2 |
| группа | июль | август | сентябрь | октябрь | ноябрь | декабрь |
| I группа | 80710,5 | 91177,2 | 113397,6 | 89148 | 60709,2 | 49272,3 |
| II группа | 26282,4 | 39085,8 | 96808,8 | 215320,8 | 73904,4 | 75894 |
| III группа | 50358 | 77707,2 | 87267 | 86396,4 | 24232,8 | 17842,8 |

4) Постройте в новом графическом окне графики функций *y=*`tg x`, *y=*`ctg x`.

5) В новом графическом окне постройте столбчатую диаграмму функции реализации фильтра патрон СМД-18 за 2000 год по месяцам

## 3.4 Контрольные вопросы

1) На какие уровни делятся функции для построения графиков?

2) Какие функции относятся к функциям высокого уровня?

3) Какой аргумент функции `plot` отвечает за цвет графика?

4) Как можно разбить графическое окно на несколько?

5) С помощью какой функции можно добавить новое графическое окно?

6) Какой аргумент функции `plot` отвечает за тип линий?

7) Какие аргументы функции `plot` отвечают за названия осей, основного и дополнительного заголовков графика?

8) Какими аргументами функции `plot` определяют пределы осей?

9) Как можно разместить в одном графическом окне нескольких графиков?

# Лабораторная работа № 4
# Математическая статистика.
# Числовые характеристики вариационного ряда

**Цель работы**: изучить команды вычисляющие числовые характеристики вариационного ряда.

**Оборудование**: ПК, оснащенный необходимым программным обеспечением.

**Объём лабораторной работы и порядок её выполнения**. Лабораторная работа выполняется в течение двух академических часов. Изучите теоретическую часть, выполните работу и упражнения для самостоятельной работы.

## 4.1 Теоретическая часть

Для обработки статистических данных в R используются следующие функции:

`as.array(x)`, `as.data.frame(x)`, `as.numeric(x)`, `as.logical(x)`, `as.complex(x)`, `as.character(x)` – преобразование переменной к заданному типу;

`is.na(x)`, `is.null(x)`, `is.array(x)`, `is.data.frame(x)`, `is.numeric(x)`, `is.complex(x)`, `is.character(x)` – проверка на то, что данный объект обладает указанным типом;

`length(x)` – число элементов в x;

`summary(object)` – общая информация об объекте;

`min(x)`, `max(x)` – минимальный и максимальный элементы объекта;

`range(x)` – вектор из минимального и максимального элемента объекта;

`sum(x)` – сумма элементов объекта;

`prod(x)` – произведение элементов объекта;

`diff(x)` – возвращает вектор из разниц между соседними элементами;

`mean(x)` – среднее арифметическое элементов объекта;

`median(x)` – медиана (средний элемент) объекта;

`sd(x)` – стандартное отклонение;

`var(x)` – дисперсия;

`cor(x)` – корреляционная матрица;

`var(x, y)` – ковариация между x и y;

`cor(x, y)` – линейная корреляция между x и y;

`aov(formula)` – дисперсионный анализ;

`anova(fit,...)` – дисперсионный анализ для подогнанных моделей fit;

`density(x)` – ядерные плотности вероятностей;

`binom.test()` – точный тест простой гипотезы о вероятности успеха в испытаниях Бернулли;

`pairwise.t.test()` – попарные сравнения нескольких независимых или зависимых выборок;

`prop.test()` – проверка гипотезы о том, что частоты какого-либо признака равны во всех анализируемых группах;

`t.test()` – тест Стьюдента.

К числовым характеристикам выборки отнесем к ним медиану, квартили, выборочную среднюю, выборочную дисперсию, эмпирические выборочные начальные и центральные момента k-го порядка, исправленные значение выборочной дисперсии, исправленные значение среднего квадратического отклонения и т.д. [7, 8].

**Определение.** *Выборочной средней* называют среднее арифметическое значение выборочной совокупности:

$$x_{\text{в}} = \frac{1}{n}\sum_{i=1}^{n} x_i.$$

**Определение.** *Выборочной дисперсией* называют среднее арифметическое квадратов отклонений значений наблюдаемого признака от их среднего значения $x_{\text{в}}$

$$D_{\text{в}} = \frac{1}{n}\sum_{i=1}^{m} n_i \left(x_i - x_{\text{в}}\right)^2.$$

**Определение.** *Выборочным средним квадратическим отклонением* (***стандартом***) называется квадратный корень из выборочной дисперсии:

$$\sigma = \sqrt{D_{\text{в}}}.$$

**Определение.** *Обычным эмпирическим моментом порядка k* называют среднее значение k-х степеней разностей $x_i - C$:

$$M'_k = \frac{1}{n}\sum_{i=1}^{m} n_i \left(x_i - C\right)^k,$$

где $n$ – объём выборки, $C$ – произвольное постоянное число (ложный нуль).

**Определение.** *Начальным эмпирическим моментом порядка k* называют обычный момент порядка $k$ при $C = 0$:

$$\nu_k = \frac{1}{n}\sum_{i=1}^{m} n_i x_i^k.$$

**Определение.** *Центральным эмпирическим моментом порядка k* называют обычный момент порядка $k$, при $C = x_{\text{в}}$:

$$\mu_k = \frac{1}{n}\sum_{i=1}^{m} n_i \left(x_i - x_{\text{в}}\right)^k.$$

## 4.2 Выполнение лабораторной работы

### *Упражнение 1*

1) Командой x<-c(10.3,10.5,9.8,11.4,12.0,11.6,10.8,11.5,10.1,10.6,10.4,9.9) введите вектор данных.

2) Просмотрите его данные командой `str(x)`.

3) Найдем количество элементов, присвоив это значение переменной `n`, командой `n<-length(x)`.

4) Просмотрите значение переменной `n` командой `str(n)`.

5) Найдем среднее выборочное $x_в$ этих данных командой `xv<-mean(x)`.

6) Просмотрите полученное значение переменной `xv` командой `str(xv)`.

7) Вычислим среднее выборочное командой `sum(x)/length(x)` согласно формуле

$$x_в = \frac{1}{n}\sum_{i=1}^{n} x_i.$$

Результаты работы команд будут следующими:

```
>x<-c(10.3,10.5,9.8,11.4,12.0,11.6,10.8,11.5,10.1,
+10.6,10.4,9.9)
> str(x)
 num [1:12] 10.3 10.5 9.8 11.4 12 11.6 10.8 11.5
10.1 10.6 ...
> n<-length(x)
> n
[1] 12
> xv=mean(x)
> xv
[1] 10.74167
> sum(x)
[1] 128.9
> sum(x)/n
[1] 10.74167
```

Для вычисления выборочной дисперсии создадим функцию пользователя, вычисляющую центральный эмпирический момент порядка k.

### *Упражнение 2*

1) Для её создания используем функцию, вычисляющую выборочный центральный момент порядка k:

`muk<-function(x,k){r<-x-mean(x);r<-r^k; mean(r)}`.

2) Воспользуемся ею для вычисления выборочной дисперсии $D_в = \mu_2$. Проделаем это командой `D<-muk(x,2)`.

3) Проверьте полученное значение командой `str(D)`.

4) Вычислим выборочную дисперсию и по другой формуле $D_в=\nu_2-(\nu_1)^2$, где

$$\nu_k = \frac{1}{n}\sum_{i=1}^{n} x_i^k.$$

Выполните команду `mean(x^2)-mean(x)^2`.

5) Найдем исправленное значение дисперсии $s^2=n/(n-1)\cdot D_в$. Выполните команду `S2<- n/(n-1)*D`.

6) Проверьте полученное значение командой `str(S2)`.

7) Поверим работу функции `var()`. Выполните команду `var(x)`

Результаты работы команд будут следующими:

```
> D<-muk(x,2)
> str(D)
 num 0.477
> mean(x^2)-xv^2
[1] 0.4774306
> S2<-n/(n-1)*D
> str(S2)
 num 0.521
> var(x)
[1] 0.5208333
```

Сравнение результатов вычислений показывает, что ***функция*** `var()` ***вычисляет исправленное значение дисперсии*** $s^2$.

Вычислим выборочный стандарт (выборочное среднее квадратическое отклонение) $\sigma=\overline{D_в}$ и его исправленное значение $s=\overline{s^2}$.

### *Упражнение 3*

1) Командой `sigma<-sqrt(D)` вычислим выборочный стандарт.

2) Командой `str(sigma)` просмотрите результат вычислений.

3) Командой `s<-sqrt(S2)` вычислим исправленное значение выборочного стандарта.

4) Командой `s` просмотрите результат вычислений.

5) Командой `sd(x)` проверим работу функции `sd()`.

Результаты работы команд будут следующими:

```
> sigma<-sqrt(D)
> str(sigma)
 num 0.691
> s<-sqrt(S2)
> s
[1] 0.7216878
> sd(x)
[1] 0.7216878
```

Анализ результатов вычислений показывает, что ***функция*** `sd()` ***вычисляет исправленное значение выборочного стандарта s***.

Вычислим выборочный коэффициент вариации $V$, асимметрию $a_s$, эксцесс $e_k$:

$$V = \frac{s}{x_в}, \quad a_s = \frac{\mu_3}{\sigma^3}, \quad e_k = \frac{\mu_4}{\sigma^4} - 3.$$

### *Упражнение 4*

1) Командой `V<-s/xv` вычислим выборочный коэффициент вариации.
2) Командой `V` просмотрите результат вычисления.
3) Командой `mu3<-muk(x,3)` вычислим выборочный центральный момент третьего порядка.
4) Командой `as<-mu3/s^3` вычислим асимметрию.
5) Командой `as` просмотрите результат вычисления.
6) Командой `mu4<-muk(x,4)` вычислим выборочный центральный момент четвертого порядка.
7) Командой `ek<-mu4/s^4` вычислим эксцесс.
8) Командой `ek` просмотрите результат вычисления.

Результаты работы команд будут следующими:

```
> V<-s/xv
> V
[1] 0.06718583
> mu3<-muk(x,3)
> as<-mu3/s^3
> as
[1] 0.3343243
> mu4<-muk(x,4)
> ek<-mu4/s^4-3
> ek
[1] -1.438916
```

Просмотрим следующие характеристики: минимальное и максимальное значения, размах выборки, медиану, нижнюю и верхнюю квартили, интерквантильный размах.

### *Упражнение 5*

1) Командой `summary(x)` просмотрим характеристики выборки. Обратите внимание, что эта команда вычисляет минимальное значение, нижнюю квартиль, медиану, среднюю выборочную, верхнюю квартиль, максимальное значение.

2) Выполняя последовательно команды `Xmin<-min(x); Xmax<-max(x); R<-Xmax-Xmin`, вычислите минимальное и максимальное значения, размах выборки.

3) Командой `R;Xmax;Xmin` просмотрите результат вычисления.

4) Командой `Med<-median(x)` вычислите медиану.

5) Командой `Med` просмотрите результат вычисления.

6) Командой `qu<-quantile(x,c(0.25,0.75))` вычислите нижнюю и верхнюю квартили.

7) Командой `qu` просмотрите результат вычисления.

8) Командой `qu[2]-qu[1]` вычислите интерквантильный размах.

9) Интерквантильный размах вычислите другой командой `IQR(x)`.

Результаты работы команд будут следующими:

```
> summary(x)
   Min. 1st Qu.   Median  Mean   3rd Qu.   Max.
   9.80   10.25   10.55   10.74   11.42   12.00
> Xmin<-min(x)
> Xmax<-max(x)
> R<-Xmax-Xmin
> R;Xmax;Xmin
[1] 2.2
[1] 12
[1] 9.8
> Med<-median(x)
> Med
[1] 10.55
> qu<-quantile(x,c(0.25,0.75))
> qu
   25%    75%
10.250 11.425
> qu[2]-qu[1]
  75%
1.175
> IQR(x)
[1] 1.175
```

### *Упражнение 6*

1) Соберите все команды в один скрипт. Проверьте его работу.

2) Сохраните его под именем `lab4_1`.

Научимся обрабатывать данные из текстового файла.

### *Упражнение 7*

1) Скопируйте из сетевой папки **Учебная на ASHP\R** файл **lab5_dan.txt** в свою рабочую папку.

2) Просмотрите его содержимое. Убедитесь, что в нем находятся только 30 чисел, записанных в столбик.

3) Командой `d2=read.table("lab5_dan.txt")` считайте данные из файла в переменную-таблицу данных `d2`.

4) Командой `x<-d2$V1` преобразуйте данные в вектор переменную `x`.

5) Командой `str(x)` просмотрите её значения.

6) Проверьте его размерность командой `n<-length(x);n`.

Создадим скрипт, вычисляющий числовые характеристики этой выборки. Запишем результаты вычислений в текстовый файл. Используем для этого функцию `sink()`. Использовать эту функцию будем, задавая значение её аргумента `split`, равное `TRUE`. При таком значении аргумента печать результата вычислений будет производиться не только в файл, но и на экран.

### *Упражнение 8*

1) При помощи функций `setwd()` и `getwd()` проверьте рабочая папка находится на вашей флешке.

2) Создайте новый скрипт и запишите его текст:

```
sink("rezult.txt",split=TRUE)
xv<-mean(x)
xv
D<-mean((x-xv)^2)
D
sigma=sqrt(D)
sigma
s2<-n/(n-1)*D
s2
s<-sqrt(D)
s
sink()
```

3) Отправьте скрипт на исполнение.

4) Проверьте результаты исполнения этого скрипта, просмотрев файл ***result.txt***.

Его содержание будет следующим:

```
[1] 60.97667
[1] 0.1824556
[1] 0.4271482
[1] 0.1887471
[1] 0.4271482
```

Эти цифры ничего не показывают и требуют пояснения. Изменим скрипт так, чтобы он записывал в файл ***result.txt*** кроме значений характеристик и их обозначения. Используем для этого функцию print(), печатающую текстовый аргумент. Её синтаксис: `print("argument")`.

Например, результаты её работы.

```
> print("Пример работы")
[1] "Пример работы"
```

```
> argument<-"Вычисление модуля "
> print(argument)
[1] "Вычисление модуля "
```

### *Упражнение 9*

1) Измените скрипт следующим образом:

```
sink("rezult.txt",split=TRUE)
xv<-mean(x)
print("xv=")
xv
D<-mean((x-xv)^2)
print("D=")
D
sigma=sqrt(D)
print("sigma=")
sigma
s2<-n/(n-1)*D
print("s2=")
s2
s<-sqrt(D)
print("s=")
s
sink()
```

2) Проверьте результаты исполнения этого скрипта, просмотрев файл ***result.txt***.

Его содержание будет следующим:

```
[1] "xv="
[1] 60.97667
[1] "D="
[1] 0.1824556
[1] "sigma="
[1] 0.4271482
[1] "s2="
[1] 0.1887471
[1] "s="
[1] 0.4271482
```

## 4.3 Самостоятельная работа

1) Создайте скрипт, позволяющий вычислить числовые характеристики для вашего варианта расчетной работы.

2) Проверьте результаты его работы. Сохраните его под именем Sam4.

3) Исправьте его так, чтобы результаты расчета записывались в текстовый файл.

4) Покажите работу скрипта преподавателю.

## 4.4 Контрольные вопросы

1) Дайте определение выборочной средней.

2) Дайте определение выборочной дисперсии.

3) Дайте определение выборочного среднего квадратического отклонения (стандарта).

4) Дайте определение обычного эмпирического момента порядка *k*.

5) Дайте определение начального эмпирического момента порядка *k*.

6) Дайте определение центрального эмпирического момента порядка k.

7) Дайте определение выборочного коэффициента вариации *V*, асимметрию $a_s$, эксцесс $e_k$.

8) При помощи какой функции записывают результаты расчета в файл?

9) Что вычисляет функция `sd()`?

# Лабораторная работа № 5
# Законы распределения случайных величин

**Цель работы**: изучить команды вычисляющие числовые характеристики вариационного ряда.

**Оборудование**: ПК, оснащенный необходимым программным обеспечением.

**Объём лабораторной работы и порядок её выполнения**. Лабораторная работа выполняется в течение двух академических часов. Изучите теоретическую часть, выполните работу и упражнения для самостоятельной работы.

## 4.1 Теоретическая часть

Рассмотрим законы распределения случайных величин, реализованных в системе R. Вспомним основные понятия, необходимые для их изучения.

**Определение.** *Функцией распределения* называют функцию $F(x)$, определяющую вероятность того, что случайная величина $X$ в результате испытания примет значение, меньшее $x$, то есть $F(x)=P(X<x)$ [7, 15, 16, 17].

### Свойства функции распределения

1) Значения функции распределения принадлежат отрезку [0; 1]: $0\leq F(x)\leq 1$.

2) F(x) − неубывающая функция, то есть $F(x_1)\geq F(x_2)$, если $x_1>x_2$.

3) Если возможные значения случайной величины принадлежат интервалу $(a;b)$, то

a) $F(x) = 0$, при $x\leq a$;

b) $F(x) =1$, при $x \geq b$.

**Определение.** *Плотностью распределения* вероятностей непрерывной случайной величины $X$ называют функцию $f(x)$ − первую производную от функции распределения $F(x)$: $f(x)=F'(x)$.

**Определение.** Плотности распределения непрерывных случайных величин называют также *законами распределения*.

В базовой установке R (пакет `stats`) реализованы следующие вероятностные распределения:

1) дискретные:
- биномиальное;
- пуассоновское;
- геометрическое;
- гипергеометрическое;
- отрицательно биномиальное;
- полиномиальное;

2) непрерывные:
- бета-распределение;

– распределение Коши;
– экспоненциальное;
– $\chi^2$-распределение;
– распределение Фишера (*f*-распределение);
– гамма-распределение;
– логистическое;
– нормальное;
– распределение Стьюдента (*t*-распределение);
– равномерное;
– распределение Вейбулла.

Кроме этого, используя дополнительные пакеты, можно реализовать еще и другие законы распределения. В этой работе будут рассмотрены не все вышеприведенные законы.

**Биномиальное распределение**

Пусть проводится $n$ испытаний, в каждом из которых событие $A$ может появиться либо не появиться. Вероятность наступления события во всех испытаниях постоянна и равна $p$ (следовательно, вероятность непоявления $q=1-p$). Случайная величина $\zeta$ описывает количество появлений события $A$ в этих испытаниях. Вероятность количества появлений события $A$ вычисляется по формуле Бернулли:

$$P_n(k)=C_n^k p^k q^{k-n}.$$

Считается, что случайная величина распределена по биномиальному закону $B(n,p)$ с параметрами $p$ и $n$. В R для биномиального распределения реализованы следующие функции:

```
dbinom(x, size, prob, log = FALSE)
pbinom(q, size, prob, lower.tail = TRUE, log.p = FALSE)
qbinom(p, size, prob, lower.tail = TRUE, log.p = FALSE)
rbinom(n, size, prob)
```

Аргументы функций:

– `x` – целочисленный неотрицательный вектор (вектор значений случайной величины $\zeta$);
– `q` – неотрицательный вектор – вектор квантилей;
– `p` – вектор вероятностей;
– `n` – длина создаваемого вектора;
– `size` – один из параметров биномиального распределения – число испытаний Бернулли;
– `prob` – второй параметр биномиального распределения – вероятность $p$;
– `log` – логический аргумент (по умолчанию его значение FALSE). Нужно ли вычислять логарифм вероятности;

– `log.p` – аналогично;

– `lower.tail` – логический аргумент. Если его значение TRUE. то используется $P(\zeta \le x)$, в противном случае используется $F(x) = P(\zeta > x)$.

Для работы с биноминальным распределением в базовой установке применяются четыре функции:

– `dbinom(x, size, prob)` – вероятность того, что случайная величина примет заданное значение. Если `x` – нецелое или отрицательное число, то будет выдано предупреждение и результатом будет ноль.

– `pbinom(q, size, prob)` – значение функции распределения в точке `q`.

– `qbinom(p, size, prob)` – значение квантиля `Qp` для заданной вероятности `p`. Квантиль `Qp` определяется как наименьшее целое значение `x` такое, что $F(x) \ge p$ (заметим, что в русской школе теории вероятностей квантиль Qp определяется как наибольшее целое значение x такое, что $F(x) < p$).

– `rbinom(n, size, prob)` – создаём вектор случайных значений (выборку), подчиняющийся заданному распределению.

**Пуассоновское распределение**

Пусть производится п независимых испытаний, в каждом из которых вероятность появления события $A$ равна $p$. Для определения вероятности k появлений события в этих испытаниях используют формулу Бернулли. Если же $n$ велико, то пользуются асимптотической формулой Лапласа. Однако эта формула непригодна, если вероятность события мала ($p<0,1$). В этих случаях ($n$ велико, $p$ мало) прибегают к асимптотической формуле Пуассона [15].

Для вывода этой нужно сделать важное допущение: произведение $np$ сохраняет постоянное значение, а именно $np=\lambda$.

Тогда, вероятность дискретной случайной величины ζ, определяющей появления события А ровно раз, вычисляется по формуле Пуассона:

$$P_n(k)=\lambda^k e^{-\lambda}/k!.$$

Распределение Пуассона реализуется в следующих случаях:

– точки попадают в неперекрещивающиеся области независимым образом;

– точки появляются поодиночке, а не парами, тройками и т.д.;

– точки распределены статистически равномерно, с некоторой плотностью $\lambda$.

Закон Пуассона иногда называют законом редких явлений – он получается при большом числе опытов $n\to\infty$ и малой вероятности события $P\to 0$.

Функции R, отвечающие за пуассоновское распределение:

```
dpois(x, lambda, log = FALSE)
ppois(q, lambda, lower.tail = TRUE, log.p = FALSE)
qpois(p, lambda, lower.tail = TRUE, log.p = FALSE)
rpois(n, lambda)
```

Аргументы функций:

`x` − неотрицательный целочисленный вектор (отрицательные значения приводят к 0),

`q` и `lambda` − неотрицательные аргументы.

**Замечание.** Использование аргумента `lower.tail = FALSE` позволяет получать более точные результаты.

### Закон равномерного распределения вероятностей

**Определение.** Распределение вероятностей называют равномерным, если на интервале, которому принадлежат все возможные значения случайной величины, плотность распределения сохраняет постоянное значение.

Плотность распределения равномерной величины будет следующей:

$$f(x) = \begin{cases} \frac{1}{b-a}, & \text{если } x \in [a; b]; \\ 0, & \text{остальные } x. \end{cases}$$

Функция распределения равномерной величины:

$$F(x) = \begin{cases} 0, & \text{если } x < a; \\ \frac{x-a}{b-a}, & \text{если } a \le x \le b; \\ 1 & \text{если } x > b. \end{cases}$$

Графики этих функций приведены на рисунке 5.1.

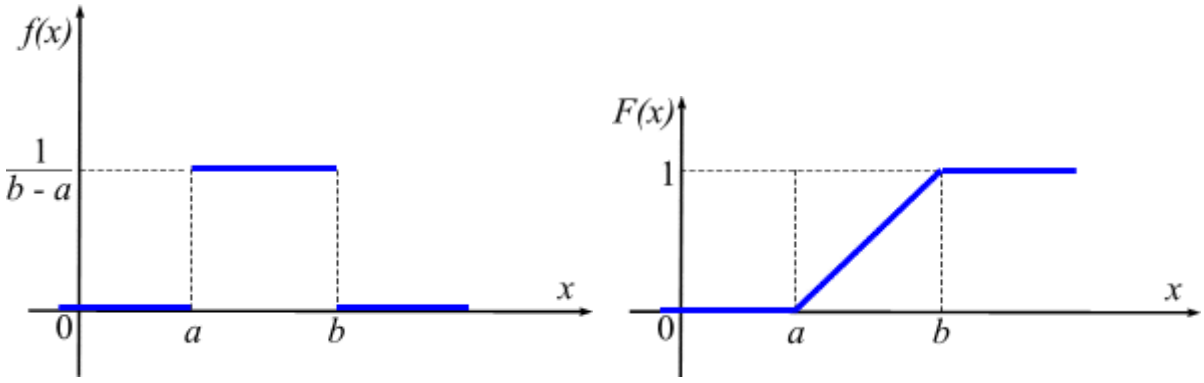



Рисунок 5.1 – Графики плотности и функции равномерного распределения

Равномерное распределение в R представлено функциями

```
unif(x, min=0, max=1, log = FALSE)
punif(q, min=0,max=1, lower.tail = TRUE, log.p = FALSE)
qunif(p, min=0,max=1, lower.tail = TRUE, log.p = FALSE)
runif(n, min=0, max=1)
```

Аргументы `min` и `max` соответствуют параметрам $a$ и $b$ распределения. Значения по умолчанию `min = 0` и `max = 1`.

### Нормальное распределение

**Определение.** Нормальным называют распределение вероятностей непрерывной случайной величины, которая описывается плотностью

$$f\ x\ = \frac{1}{\sigma\ \sqrt{2\pi}}\ e^{-\ x-a\ ^2/2\sigma^2},$$

где $a$ − математическое ожидание нормального закона, а $\sigma$ − среднее квадратичное отклонение.

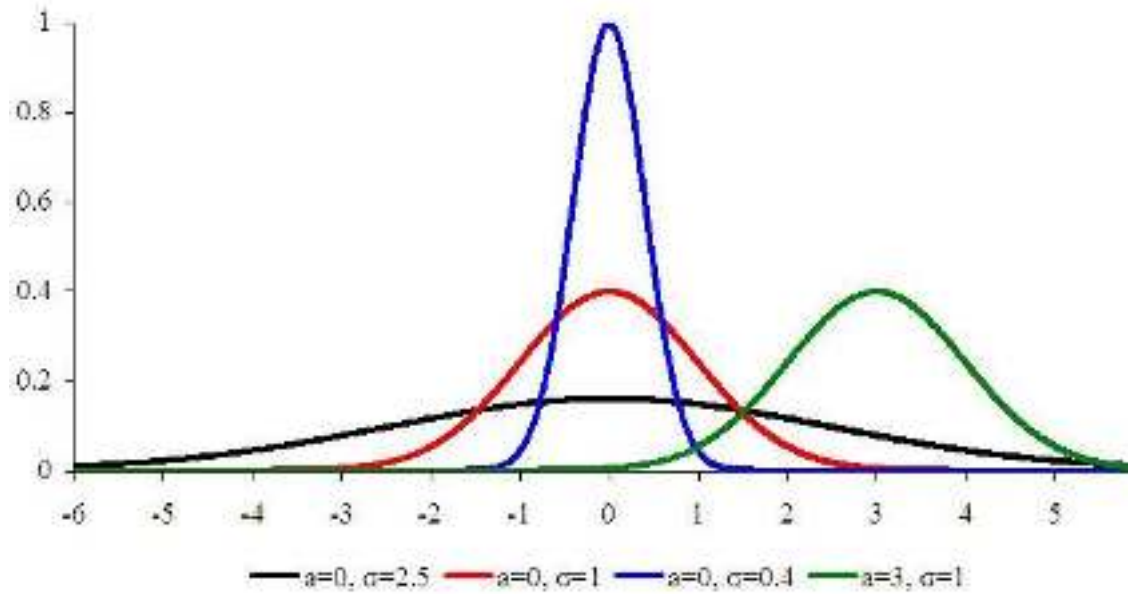



Рисунок 5.2 – Кривые плотности нормального распределения для различных значений параметров $a$ и $\sigma$

В R за нормальное распределение отвечают функции:

```
dnorm(x, mean = 0, sd = 1, log = FALSE)
pnorm(q, mean = 0,sd=1, lower.tail=TRUE, log.p = FALSE)
qnorm(p, mean = 0, sd=1, lower.tail=TRUE, log.p= FALSE)
rnorm(n, mean = 0, sd = 1)
```

Аргументы:

`x` и `q` − значения (квантили) случайной величины.

`mean` и `sd` − параметры нормального закона – математическое ожидание $a$ и среднее квадратичное отклонение $\sigma$.

`p` − вероятности.

`n` − объём создаваемой случайной выборки.

**Задача.** Найти вероятность попадания случайной величины в заданный интервал. Если случайная величина $X$ задана плотностью распределения $f(x)$, то вероятность того, что $X$ примет значение, принадлежащее интервалу (α, β), такова:

$$P\ \alpha < X < \beta\ = \int_{\alpha}^{\beta} f\ x\ dx.$$

Пусть случайная величина $X$ распределена по нормальному закону. Тогда вероятность того, что $X$ примет значение, принадлежащее интервалу (α,β), равна

$$P\ \alpha < X < \beta\ = \int_{\alpha}^{\beta} e^{-\ x-a\ ^2\ (2\sigma^2)} dx = \Phi\ \frac{\beta - a}{\sigma}\ - \Phi\ \frac{\alpha - a}{\sigma}\ .$$

**Пример 5.1.** Случайная величина X распределена по нормальному закону. Математическое ожидание и среднее квадратическое отклонение этой величины соответственно равны 30 и 10. Найти вероятность того, что X примет значение, принадлежащее интервалу (10, 50) [10].

Решение.

Согласно предыдущей формуле

$$P\ \alpha < X < \beta\ = \Phi\ \frac{50 - 30}{10}\ - \Phi\ \frac{10 - 30}{10}\ = \Phi\ 2\ - \Phi\ -2\ = 0{,}9544.$$

## Распределение $\chi^2$

**Определение.** Распределение $\chi^2$ с $n$ степенями свободы называется распределение суммы $n$ квадратов независимых стандартных нормальных величин:

$$\chi^2 = \sum_{i=1}^{n} Z_i, \quad Z_i = N\ 0{,}1\ .$$

Функция плотности распределения

$$f\ x\ = \begin{cases} \dfrac{1}{2^{n\ 2}\Gamma\ n/2}\, x^{n\ 2-1} e^{n/2} & x > 0, \\ 0, & x \le 0, \end{cases}$$

где $\Gamma(x)$ – гамма-функция:

$$\Gamma\ x\ = \int_{0}^{\infty} t^{x-1} e^{-t} dt.$$

Четыре функции в R отвечают за $\chi^2$-распределение. Функция

`dchisq(x, df, ncp=0, log = FALSE)` отвечает за плотность распределения случайной величины. Функция

`pchisq(q, df, ncp=0, lower.tail = TRUE, log.p = FALSE)` – функция распределения случайной величины. Функция

`qchisq(p, df, ncp=0, lower.tail = TRUE, log.p = FALSE)` – применяется для вычисления квантилей. Функция

`rchisq(n, df, ncp=0)` – для построение случайной выборки.

Остановимся только на аргументах функций:

`x` и `q` – неотрицательные числовые вектора.

`df` – параметр $\chi^2$-распределения – число степеней свободы.

`ncp` – параметр смещения (неотрицательное число).

## Распределение Стьюдента

Закону распределения Стьюдента с $n$ степенями свободы удовлетворяет отношение:

$$t\ n\ = \frac{Z}{\sqrt{\chi^2(n)/n}},$$

где Z – стандартная нормальная величина, независимая от $\chi^2$.

Функция плотности распределения Стьюдента имеет вид:

$$f\ x\ = \frac{\Gamma\ \frac{n+1}{2}}{\Gamma\ \frac{n}{2}\ \sqrt{\pi n}} \cdot\ 1 + \frac{x^2}{n}\ .$$

Функции в R:

```
dt(x, df, ncp, log = FALSE)
pt(q, df, ncp, lower.tail = TRUE, log.p = FALSE)
qt(p, df, ncp, lower.tail = TRUE, log.p = FALSE)
rt(n, df, ncp)
```

Аргумент `df` определяет число степеней свободы для распределения Стьюдента (положительное число, может быть `Inf`).

## Показательное распределение

**Определение.** Показательным (экспоненциальным) называют распределение вероятностей непрерывной случайной величины *X*, которое описывается плотностью:

$$f\ x\ = \begin{cases} 0, & \text{при } x < 0, \\ \lambda e^{-\lambda x}, & \text{при } x \geq 0, \end{cases}$$

где λ – постоянная положительная величина.

Функция распределения показательного закона

$$F\ x\ = \begin{cases} 0, & \text{при } x < 0, \\ 1 - e^{-\lambda x}, & \text{при } x \geq 0. \end{cases}$$

Функция плотности распределения *f*(*x*) и функция распределения *F*(*x*) показательного закона при λ = 1 приведены на рисунке 5.3.

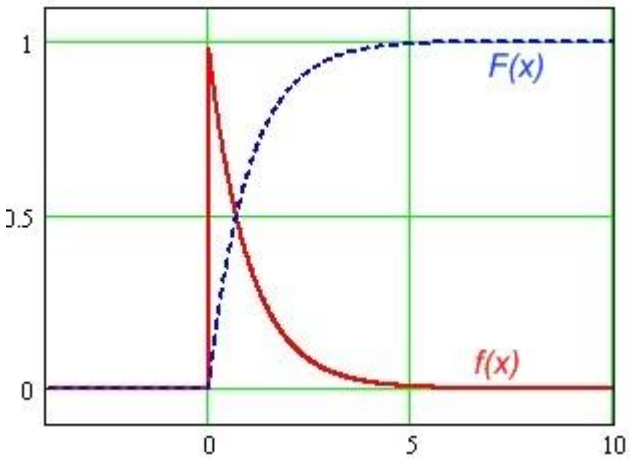



Рисунок 5.3 – Функция *f*(*x*) и функция *F*(*x*) при λ = 1

Показательное распределение реализовано в R при помощи функций

```
dexp(x, rate = 1, log = FALSE)
pexp(q, rate = 1, lower.tail = TRUE, log.p = FALSE)
```

```
qexp(p, rate = 1, lower.tail = TRUE, log.p = FALSE)
rexp(n, rate = 1)
```

Аргумент `rate` соответствует параметру $\lambda$ экспоненциального распределения.

## 5.2 Выполнение лабораторной работы

Построим графики плотности распределения и функции распределения биноминального распределения в одном графическом окне, разбив его на две части. Строить будем графики для четырех значений вероятности $p \in \{0{,}1;\ 0{,}3;\ 0{,}5;\ 0{,}7\}$. Цвет графиков для этих значений вероятности сделаем соответственно синий, красный, желтый, бирюзовый.

### *Упражнение 1*

1) Командой `p<-seq(1,7,2)/10` создайте вектор значений вероятности.

2) Командой `p` просмотрите результат вычислений.

3) Командой `par(mfrow=c(1,2))` создайте графическое окно и разбейте его на две части.

4) Командами `n <- 24; x <- seq(0, n)` зададим число опытов и вектор (переменную) `x`.

5) Командой `P<-dbinom(x,n,p[1])` создайте вектор значений теоретической плотности вероятности для $p_1$=0,1.

6) Постройте график плотности вероятности для $p_1$=0,1 командой `plot(x,P,type='l',lty='dashed',lwd=2,col='blue')`.

7) Командой `P<-dbinom(x,n,p[2])` создайте вектор значений теоретической плотности вероятности для $p_2$=0,3.

8) Постройте график плотности вероятности для $p_2$=0,3 командой `lines(x,P,type='b',col='red')`.

9) Командой `P<-dbinom(x,n,p[3])` создайте вектор значений теоретической плотности вероятности для $p_3$=0,5.

10) Постройте график плотности вероятности для $p_3$=0,5 командой `lines(x,P,type='o',col='yellow',lwd=2)`.

11) Командой `P<-dbinom(x,n,p[4])` создайте вектор значений теоретической плотности вероятности для $p_4$=0,7.

12) Постройте график плотности вероятности для $p_4$=0,7 командой `lines(x,P,type='l',col='cyan',lwd=2)`.

### *Упражнение 2*

1) Командой `P<-pbinom(x,n,p[1])` создайте вектор значений теоретической функции распределения вероятности для $p_1$=0,1.

2) Постройте график плотности вероятности для $p_1$ = 0,1 командой `plot(x,P,type='l',lty='dashed',lwd=2,col='blue')`.

3) Командой P<-pbinom(x,n,p[2]) создайте вектор значений теоретической функции распределения вероятности для $p_2 = 0{,}3$.

4) Постройте график плотности вероятности для $p_2$ = 0,3 командой `lines(x,P,type='b',col='red')`.

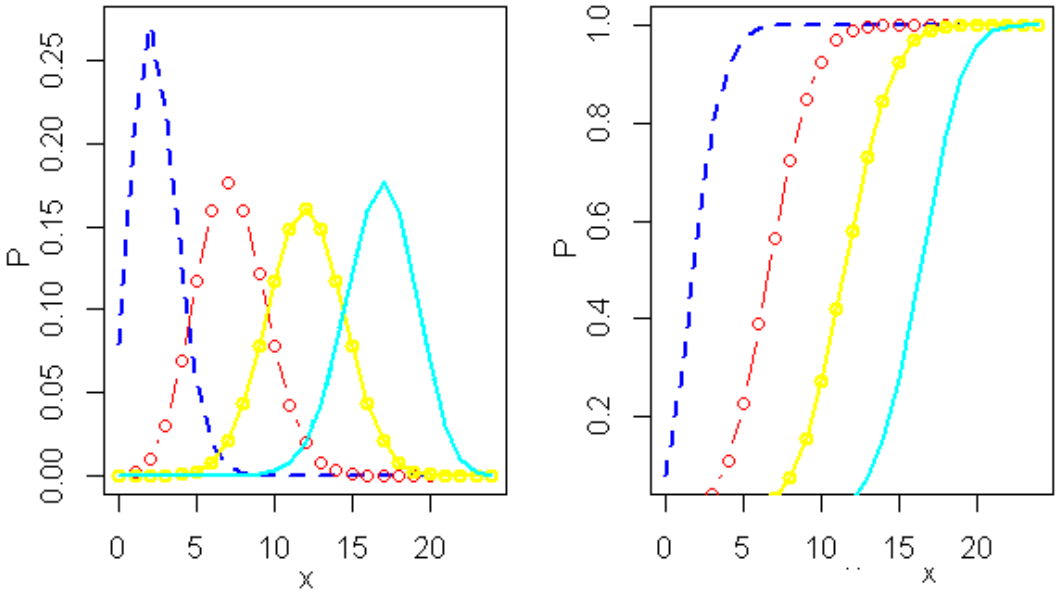
5) Командой `P<-pbinom(x,n,p[3])` создайте вектор значений теоретической функции распределения вероятности для $p_3 = 0{,}5$.

6) Постройте график плотности вероятности для $p_3$ = 0,5 командой `lines(x,P,type='o',col='yellow',lwd=2)`.

7) Командой `P<-pbinom(x,n,p[4])` создайте вектор значений теоретической функции распределения вероятности для $p_4 = 0{,}7$.

8) Постройте график плотности вероятности для $p_4$ = 0,7 командой `lines(x,P,type='l',col='cyan',lwd=2)`.

Результаты построения приведены на рисунке 5.4.




*а* *б*

Рисунок 5.4 – Графики плотности распределения (*а*)
и функции распределения (*б*)

Решим задачу. Пусть проводится десять испытаний. Вероятность появления события A в каждом их одинакова и равна 0,4. Найти $P_n(k)$ для каждого значения $k$ = 0,1,2,…,10. Следует помнить, что сумма всех вероятностей равна единице.

### *Упражнение 3*

1) Командой `size<-10` задайте количество испытаний.

2) Командой `prob<-0.4` задайте значение вероятности $p$.

3) Командой `dbinom(0:10,size,prob)` вычислите $P_n(k)$ для каждого значения $k$ = 0,1,2,…,10. Результатом работы этой команды:

```
0.00604662 0.04031078 0.12093235 0.214990848 0.25082266
 0.20065812 0.11147673 0.04246732 0.01061683 0.00157286
                    0.0001048576
```

4) Командой `sum(dbinom(0:10,size,prob))` вычислите сумму вероятностей. Она будет равна единице.

Построим выборку из 30 элементов, распределенных по биномиальному закону с вероятностью 0,3 появления события ***A*** в каждом испытании.

### *Упражнение 4*

1) Командой `prob<-0.3` задайте значение вероятности *p*.
2) Командой `size<-30` задайте количество испытаний.
3) Командой `rbinom(30,size,prob)` создайте из 30 элементов, распределенных по биномиальному закону. Результаты выполнения этой команды приведены ниже:

```
 [1]  8 11 10 11  9  9 11  4  9 11 10  9 12  9  7  5  6
      [18]  8  9 10  8 12  8  9  8 10  7  9 11  6
```

4) Командой `x.f<-ordered(rbinom(30,size,prob))` создайте категориальную переменную `x.f` полученной выборки.
5) Командой `plot(x.f,ylab="Частота",col="gray")` постройте диаграмму этой переменной (рисунок 5.5).

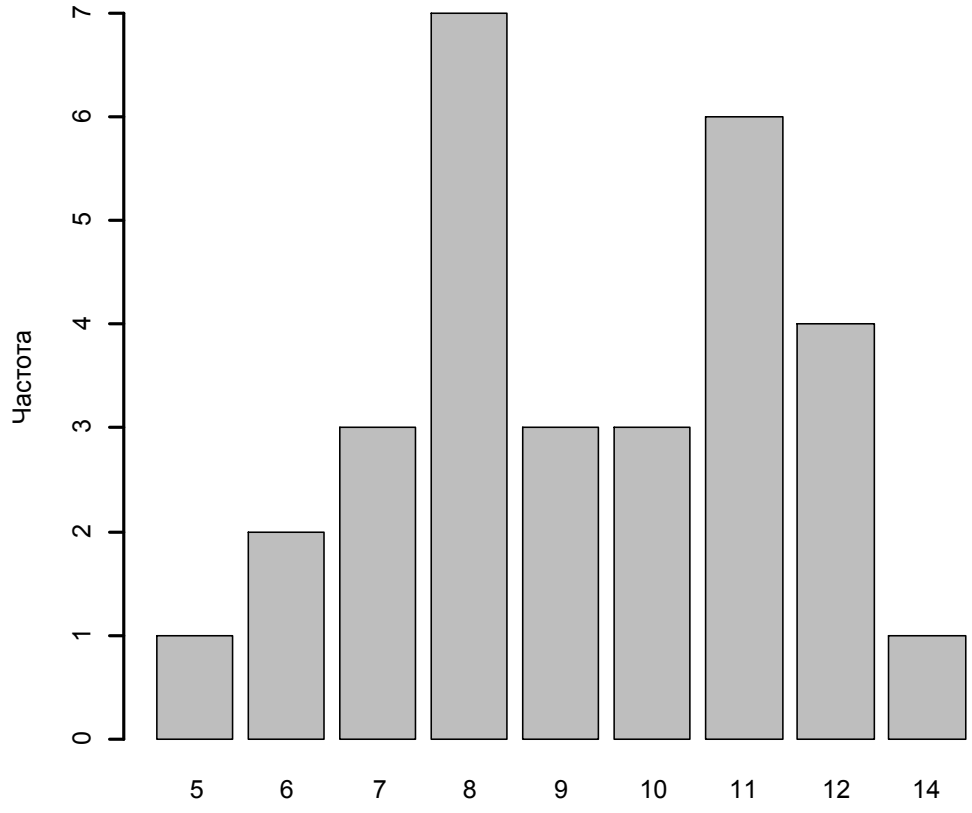



Рисунок 5.5 – Диаграмма к упражнению 4

Построим графики плотности и функции распределения пуассоновского закона.

### *Упражнение 5*

1) Командами `n<-1000;p<-0.01` введите значения вероятности и число испытаний.

2) Командой `la<-n*p` найдите значение параметра λ.

3) Командой `x<-1:200` введите вектор значений переменной.

4) Командой `f1<-dpois(x, la)` присвойте значение плотности распределения вероятности для λ=10.

5) Командой `par(mfrow=c(1,2))` создайте графическое окно и разбейте его на две части.

6) Командой `plot(x,f1,type='l',lty='dashed')` постройте её график пунктирной линией.

7) Командой `p<-0.03` измените значение вероятности.

8) Командой `la<-n*p` измените значение параметра λ.

9) Командой `f1<-dpois(x, la)` присвойте значение плотности распределения вероятности для λ=30.

10) Командой `lines(x,f1,col='red',lwd=2)` постройте её график красного цвета.

11) Командой `p<-0.06` измените значение вероятности.

12) Командой `la<-n*p` измените значение параметра λ.

13) Командой `f1<-dpois(x, la)` присвойте значение плотности распределения вероятности для λ=60.

14) Командой `lines(x,f1,type='o',col='blue',lwd=2)` постройте её график синего цвета.

15) Командой `p<-0.09` измените значение вероятности.

16) Командой `la<-n*p` измените значение параметра λ.

17) Командой `f1<-dpois(x, la)` присвойте значение плотности распределения вероятности для λ=90.

18) Командой `lines(x,f1,type='b',col='gray',lwd=2)` постройте её график зеленого цвета.

### *Упражнение 6*

1) Командой `p<-0.01` введите значения вероятности.

2) Командой `la<-n*p` измените значение параметра λ.

3) Командой `F1<-ppois(x, la)` присвойте значение функции распределения вероятности для λ=10.

4) Командой `plot(x,F1,type='l',lty='dashed')` постройте её график пунктирной линией.

5) Командой `p<-0.03` измените значение вероятности.

6) Командой `la<-n*p` измените значение параметра λ.

7) Командой `F1<-ppois(x, la)` присвойте значение функции распределения вероятности для λ=30.

8) Командой `lines(x,F1,col='red',lwd=2)` постройте её график красного цвета.

9) Командой `p<-0.06` измените значение вероятности.

10) Командой `la<-n*p` измените значение параметра λ.

11) Командой `F1<-ppois(x, la)` присвойте значение функции распределения вероятности для λ=60.

12) Командой `lines(x,F1,type='o',col='blue',lwd=2)` постройте её график синего цвета.

13) Командой `p<-0.09` измените значение вероятности.

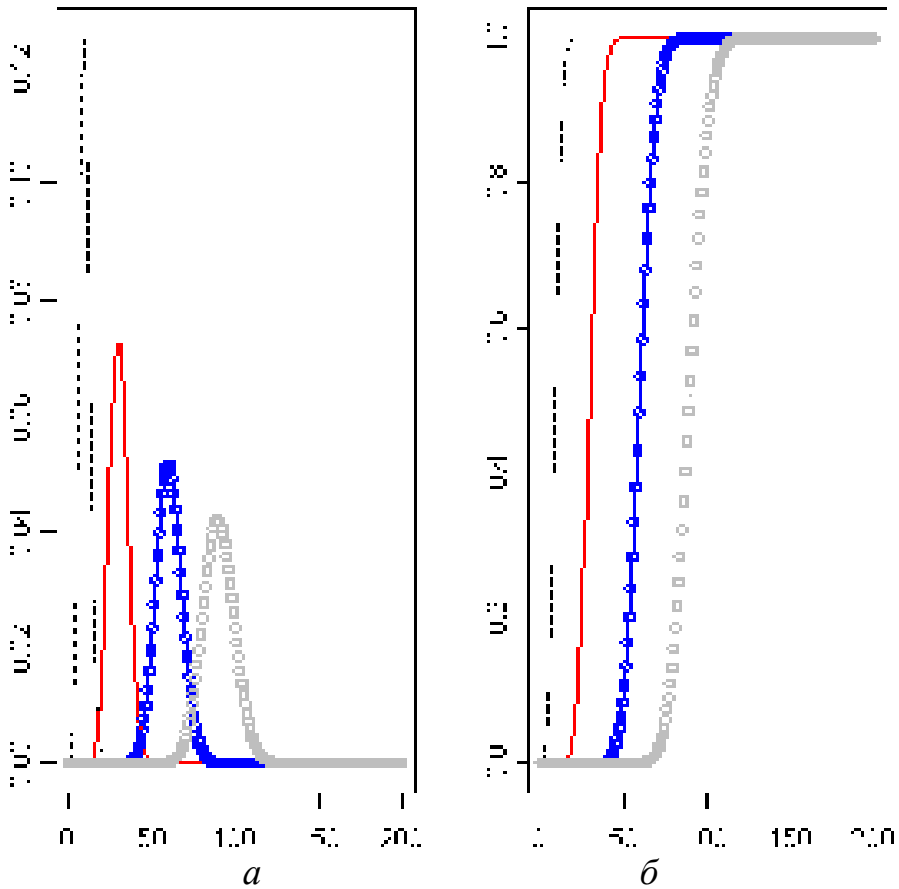
14) Командой `la<-n*p` измените значение параметра λ.

15) Командой `F1<-ppois(x, la)` присвойте значение функции распределения вероятности для λ=90.

16) Командой `lines(x,F1,type='b',col='gray',lwd=2)` постройте её график зеленого цвета.

Результаты построения приведены на рисунке 5.6.

Создайте выборку из 200 элементов при λ=10, распределенных по пуассоновскому закону.



*а* *б*

Рисунок 5.6 – Графики плотности распределения (*а*)
и функции распределения (*б*)

### *Упражнение 7*

1) Командой `x1<-rpois(200,10)` создайте из 200 элементов, распределенных по биномиальному закону.

2) Командой `x1` просмотрите полученную выборку.

Построим графики плотности распределения нормального закона, изменяя значения математического ожидания и среднего квадратического отклонения.

### *Упражнение 8*

1) Командами `a<-1;sigma<-0.9` задайте значения $a$ и $\sigma$.

2) Командой `f1<-dnorm(x,a,sigma)` присвойте значение плотности распределения вероятности.

3) Командой `op=par(mfrow=c(2,1))` создайте графическое окно и разбейте его на две части.

4) Командой
`plot(x,f1,type='l',lty='dashed',ylab='f(x)',`
`lwd=2,col='green',ylim=c(0,0.8))` постройте её график пунктирной линией зеленого цвета

5) Командой `curve(dnorm(x,mean=a,sd=0.5),col="red",` `add=TRUE,lwd=2)` постройте график плотности распределения вероятностей красного цвета для значения σ=0,5.

6) Командой `curve(dnorm(x,mean=a,sd=1.5),col="blue",` `add=TRUE,lwd=2)` постройте график плотности распределения вероятностей синего цвета для значения σ=1,5.

7) Командой `legend("topleft",c(expression(N(1,0.9)),` `expression(N(1,0.5)), expression(N(1,1.5))),`
`col=c("green","red","blue"),lty=1)` создайте легенду графика и разместите её в верхнем левом углу.

8) Разместите название диаграммы над ней командой
`title(main="Графики f(x) нормального распределения")`

9) Командой `plot(x,f1,type='l',lty='dashed',` `ylab='f(x)', lwd=2,col='green')` постройте в нижней части графического окна график плотности распределения вероятностей пунктирной линией зеленого цвета для значений $a = 1$ и $\sigma = 0{,}9$.

10) Командой `curve(dnorm(x,mean=2,sd=sigma),col="red",` `add=TRUE,lwd=2)` постройте график плотности распределения вероятностей красного цвета для значения $a$=2.

11) Командой `curve(dnorm(x,mean=3,sd=sigma),col="blue",` `add=TRUE,lwd=2)` постройте график плотности распределения вероятностей синего цвета для значения $a = 3$.

12) Командой `legend("topleft",c(expression(N(1,0.9)),`

`expression(N(2,0.9)), expression(N(3,0.9))), col=c("green","blue","red"),lty=1)` создайте легенду графика и разместите её в верхнем левом углу.

13) Разместите название диаграммы над ней командой `title(main="Графики f(x) нормального распределения")`

Результаты построения приведены на рисунке 5.7.

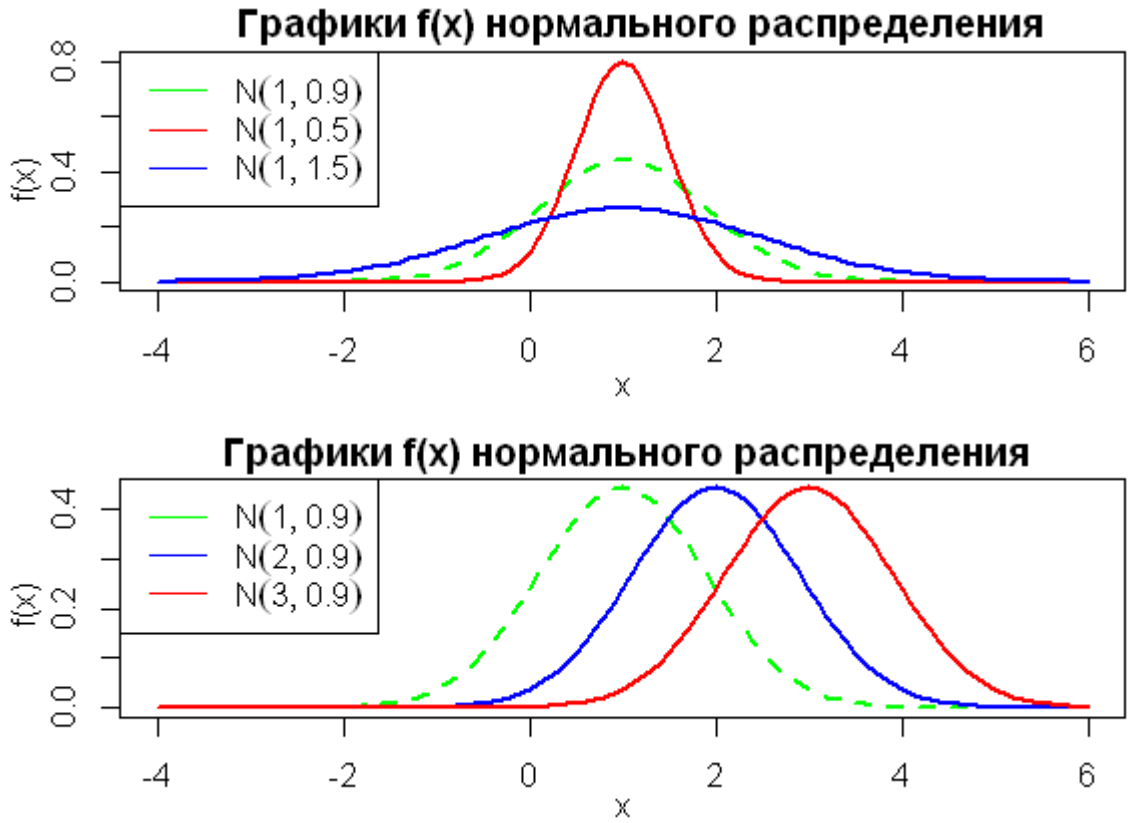



Рисунок 5.7 – Графики *f*(*x*) нормального распределения

Создайте выборку из 30 элементов, подчиняющуюся нормальному закону распределения выборки.

### *Упражнение 9*

1) Командой `x<-rnorm(30,2,0.9)` создайте выборку из 30 элементов с математическим ожиданием $a = 2$ и средним квадратическим отклонением $\sigma = 0{,}9$.

2) Командой `x` просмотрите значения выборки.

```
 [1] 1.6619083 1.6700321 1.7338903 3.2976384 1.3722155 1.6506492
2.5872828
 [8] 3.0122952 1.3051003 1.5427224 2.4712585 2.9159788 1.7739519
0.7130059
[15] 3.5382089 3.2915626 1.3606660 1.9414392 0.4164781 2.5127507
3.4511121
```

[22] 0.5264474 1.2983883 1.4229408 1.3869817 0.1700430 2.4508672 0.6213817

[29] 1.9775021 2.5336862

Решим задачу из примера 5.1, используя формулу

$$P(\alpha<X<\beta)=F(\beta)-F(\alpha).$$

### *Упражнение 10*

1) Командами `a<-30;sigma<-10` задайте значения $a$ и $\sigma$.

2) Командой `P<-pnorm(50,a,sigma)-pnorm(10,a,sigma)` вычислите значение вероятности $P(10<X<50)$.

3) Командой `P` просмотрите результат вычисления. Он будет следующим: `[1] 0.9544997`.

4) Сравните его с решением, приведенным выше в теоретической части.

Построим графики плотности и функции распределения $\chi^2$ с $n$ степенями свободы. В нижеприведенном примере (упражнение 11) $n$ зададим значения, равные 4, 6, 8, 10.

### *Упражнение 11*

1) Командой `par(mfrow=c(1,2))` создайте графическое окно и разбейте его на две части.

2) Командой `x<-0:30` присвойте значению переменной последовательность целых чисел от 0 до 30.

3) Командой `f<-dchisq(x,4)` присвойте значение плотности распределения вероятности для $n = 4$.

4) Командой `plot(x,f,type='l',lty='dashed')` постройте её график пунктирной линией.

5) Командой `curve(dchisq(x,6),col='red',add=TRUE, lwd=2)` постройте график плотности распределения вероятностей красного цвета для значения $n = 6$.

6) Командой `curve(dchisq(x,8),col='blue',add=TRUE, lwd=2)` постройте график плотности распределения вероятностей синего цвета для значения $n = 8$.

7) Постройте график плотности распределения вероятностей малинового цвета для значения $n = 10$ командой `curve(dchisq(x,10), col='magenta',add=TRUE,lwd=2)`.

8) Командой `F<-pchisq(x,4)` присвойте значение функции распределения вероятности для $n = 4$.

9) Постройте график функции распределения вероятностей пунктирной линией командой `plot(x,f,type='l',lty='dashed', ylab="F(x)",lwd=2)`.

10) Постройте график функции распределения вероятностей красного цвета для $n$ = 6 командой `curve(pchisq(x,6),col='red', add=TRUE, lwd=2).`

11) Постройте график функции распределения вероятностей синего цвета для $n$ = 8 командой `curve(pchisq(x,8),col='blue', add=TRUE, lwd=2).`

12) Постройте график функции распределения вероятностей малинового цвета для $n$ = 10 командой `curve(pchisq(x,10),col='magenta', add=TRUE,lwd = 2).`

Результаты построения приведены на рисунке 5.8.

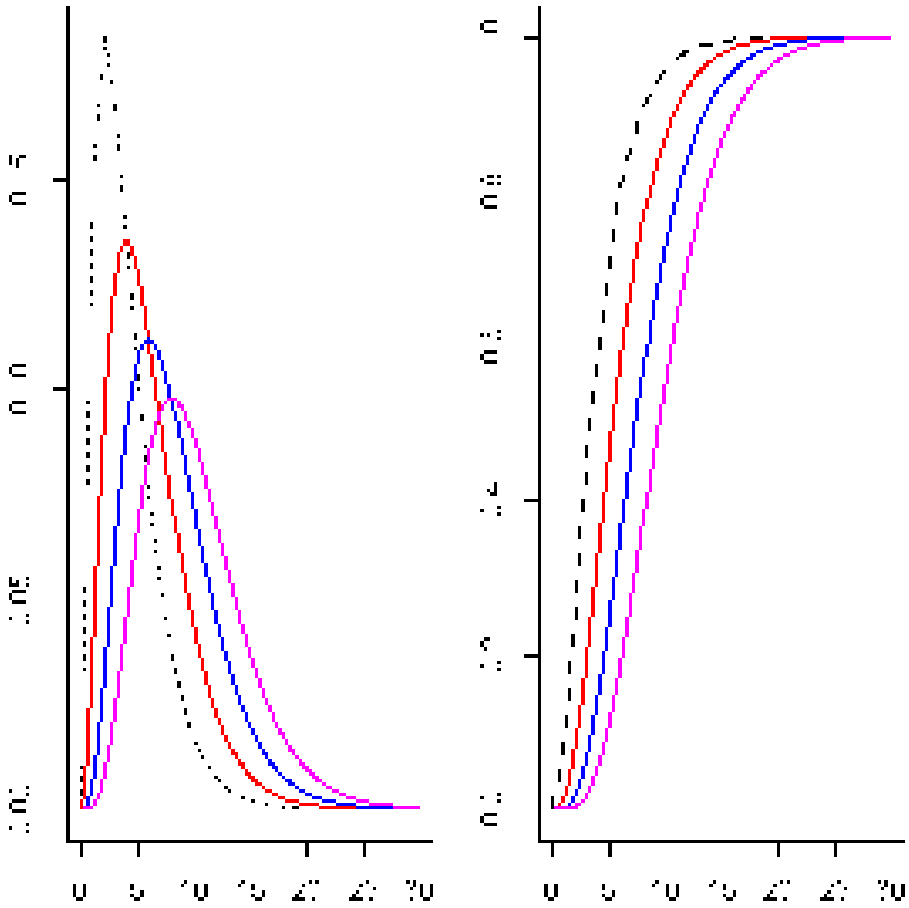


Рисунок 5.8 – Графики плотности и функции $\chi^2$ распределения

Создадим выборку из 30 элементов, распределенную по $\chi^2$ закону.

### *Упражнение 12*

1) Командой `df<-5` задайте значение степени свободы.
2) Командой `x<-rchisq(30, df)` создайте выборку.
3) Командой `x` просмотрите результат.

Результат вычислений будет следующим:

```
6.5427841  5.4116964  2.8569057  1.0691829 18.6657013
```

```
3.5055362   2.2525477   6.1522938   3.7397555   2.0304810
3.3900261   5.3442152   8.7337804   1.1397747   7.9015066
2.4036642   3.9214482   3.5990736   2.6703758   8.2848217
5.9463497   4.2875589  10.7468535   0.8791342   5.5881547
1.5100675  3.3064938  4.3204288 22.0361941  1.4783696
```

Построим графики плотности и функции распределения Стьюдента с *n* степенями свободы. В нижеприведенном примере (упражнение 12) *n* зададим значения, равные 1, 2, 4, 8.

### *Упражнение 13*

1) Командой `par(mfrow=c(1,2))` создайте графическое окно и разбейте его на две части.

2) Командой `x<-seq(-4,4,0.5)` создайте последовательность чисел.

3) Командой `f=dt(x,1)` вычислите значение плотности распределения вероятности для $n = 1$.

4) Постройте её график командой `plot(x,f,type='l', lty='dashed', lwd=2, ylab='f(x)', +ylim=c(0,0.4))`.

5) Командой `curve(dt(x,2),col='red',add=TRUE,lwd=2)` постройте график плотности распределения вероятности красного цвета для $n = 2$.

6) Командой `curve(dt(x,4),col='blue',add=TRUE,lwd=2)` постройте график плотности распределения вероятности синего цвета для $n = 4$.

7) Командой `curve(dt(x,8),col='green',add=TRUE,lwd=2)` постройте график плотности распределения вероятности зеленого цвета для $n = 8$.

8) Командой `legend("topright",c(expression(t(1)), expression(t(2)),expression(t(4)),expression(t(8))), col=c("black","red", "blue","green") ,lty=1)` создайте легенду графика и разместите её в верхнем правом углу.

9) Разместите название диаграммы над ней командой `title('Плотность f(x)')`.

10) Командой `F<-pt(x,1)` присвойте значение функции распределения вероятности для $n$=1.

11) Постройте её график пунктирной линией командой $n = 1$ `plot(x,F,type='l',lty='dashed',lwd=2,ylab='F(x)',ylim=c(0,1))`

12) Командой `curve(pt(x,2),col='red',add=TRUE,lwd=2)` постройте график функции распределения вероятностей красного цвета для $n$=2.

13) Командой `curve(pt(x,4),col='blue',add=TRUE,lwd=2)` постройте график функции распределения вероятностей синего цвета для $n = 4$.

14) Командой `curve(pt(x,8),col='green', add=TRUE, lwd = 2)` постройте график функции распределения вероятностей зелёного цвета для $n = 8$.

15) Разместите название диаграммы над ней командой `title('Функция F(x)')`

16) Командой `legend("bottomright",c(expression(t(1)), expression(t(2)), expression(t(4)), expression(t(8))), col=c("black","red","blue","green"),lty=1)` создайте легенду графика и разместите её в нижнем правом углу.

Результаты построения приведены на рисунке 5.9.

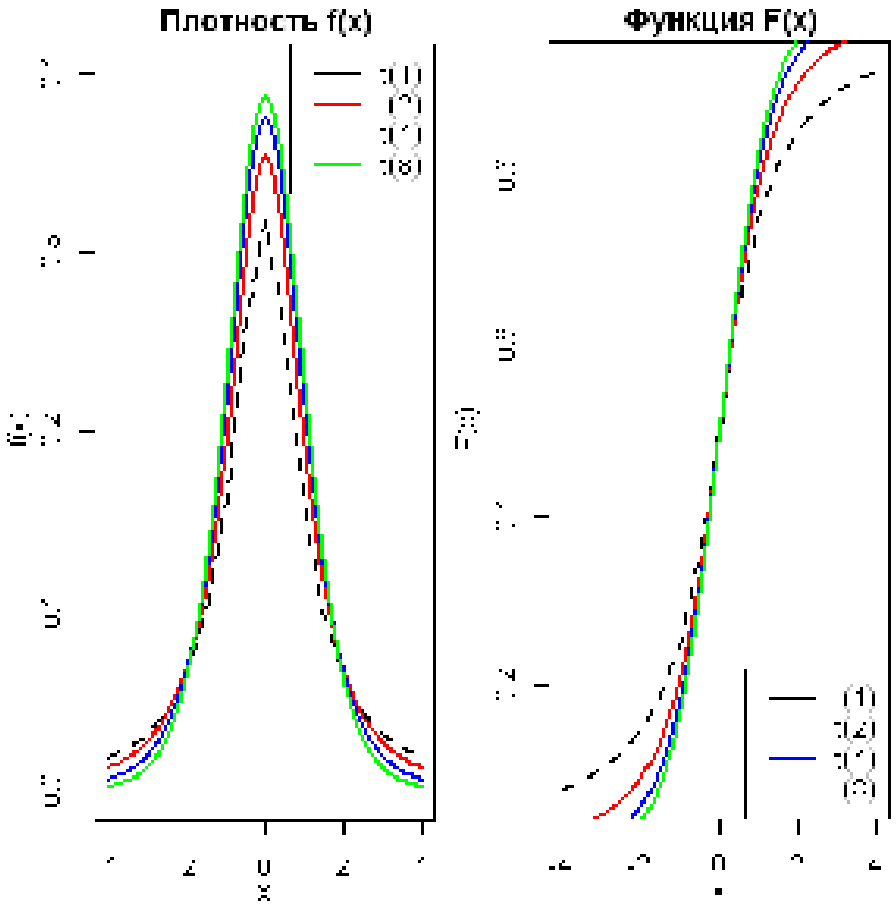



Рисунок 5.9 – Графики плотности и функции распределения Стьюдента

Создадим выборку из 30 элементов, распределенную по закону Стьюдента.

### *Упражнение 14*

1) Командой `x<-rt(30,2)` создайте выборку.
2) Командой `x` просмотрите результат.
3) Командой `sort(x)` отсортируйте выборку в порядке возрастания.

Результат вычислений будет следующим:

```
-2.33316668  -2.02672671  -2.02419504  -1.54072262
-1.29235051  -0.62827551  -0.54838382  -0.51078558
```

```
-0.45863857  -0.40917714  -0.28413702  -0.13753769
 0.09983005   0.16095910   0.22252752   0.37882176
 0.42463746   0.49282241   0.55179783   0.56190728
 0.66121616   0.67591832   0.78361437   0.91496009
 1.02297659   1.04807294   1.30111026   1.47589356
 1.55288534   2.01164644
```

Построим графики плотности распределения и функции распределения показательного распределения в одном графическом окне, разбив его на две части. Строить будем графики для четырех значений параметра $\lambda \in \{1; 1{,}5; 2; 0{,}2\}$. Цвет графиков для этих значений вероятности сделаем соответственно черный (пунктирная линия), красный, синий, зеленый.

### *Упражнение 15*

1) Командой `par(mfrow=c(1,2))` создайте графическое окно и разбейте его на две части.

2) Командой `rate<-1` присвойте параметру $\lambda = 1$.

3) Командой `f<-dexp(x,rate)` вычислите значение плотности распределения вероятности для $\lambda=1$.

4) Постройте её график командой `plot(x,f,type='l', lty='dashed', lwd=2, ylab='f(x)')`

5) Командой `rate<-1.5` присвойте параметру $\lambda=1{,}5$.

6) Командой `curve(dexp(x,rate),col='red',add=TRUE, lwd=2)` постройте график плотности распределения вероятности красного цвета.

7) Командой `rate<-2` присвойте параметру $\lambda = 2$.

8) Командой `curve(dexp(x,rate),col='blue',add=TRUE, lwd=2)` постройте график плотности распределения вероятности красного цвета.

9) Командой `rate<-0.2` присвойте параметру $\lambda=0{,}2$.

10) Командой `curve(dexp(x,rate),col='green',add=TRUE, lwd=2)` постройте график плотности распределения вероятности красного цвета.

11) Разместите название диаграммы командой `title('Плотность f(x)')`.

12) Командой `legend("topright",c(expression(exp(1)), expression(exp(1.5)),expression(exp(2)), expression(exp(0.2))), col=c("black","red","blue","green"), lty=1)` создайте легенду графика и разместите её в верхнем правом углу.

13) Командой `rate<-1` присвойте параметру $\lambda=1$.

14) Командой `F<-pexp(x,rate)` присвойте значение функции распределения вероятности.

15) Постройте её график командой `plot(x,F,type='l', lty='dashed', lwd=2, ylab='F(x)')`.

16) Командой `rate<-1.5` присвойте параметру λ = 1,5.

17) Командой `curve(dexp(x,rate),col='red',add=TRUE, lwd=2)` постройте график функции распределения вероятностей красного цвета.

18) Командой `rate<-2` присвойте параметру λ = 2.

19) Командой `curve(pexp(x,rate),col='blue',add=TRUE, lwd=2)` постройте график функции распределения вероятностей синего цвета.

20) Командой `rate<-0.2` присвойте параметру λ=0,2.

21) Командой `curve(pexp(x,rate),col='green',add=TRUE, lwd=2)` постройте график функции распределения вероятностей зеленого цвета.

22) Разместите название диаграммы командой `('Функция F(x)')`.

23) Создайте легенду графика и разместите её в нижнем правом углу командой `legend("bottomright",c(expression(exp(1)), +expression(exp(1.5)), expression(exp(2)), + expression(exp(0.2))),+col=c("black","red", "blue", "green"),lty=1)`.

Результаты построения приведены на рисунке 5.10.

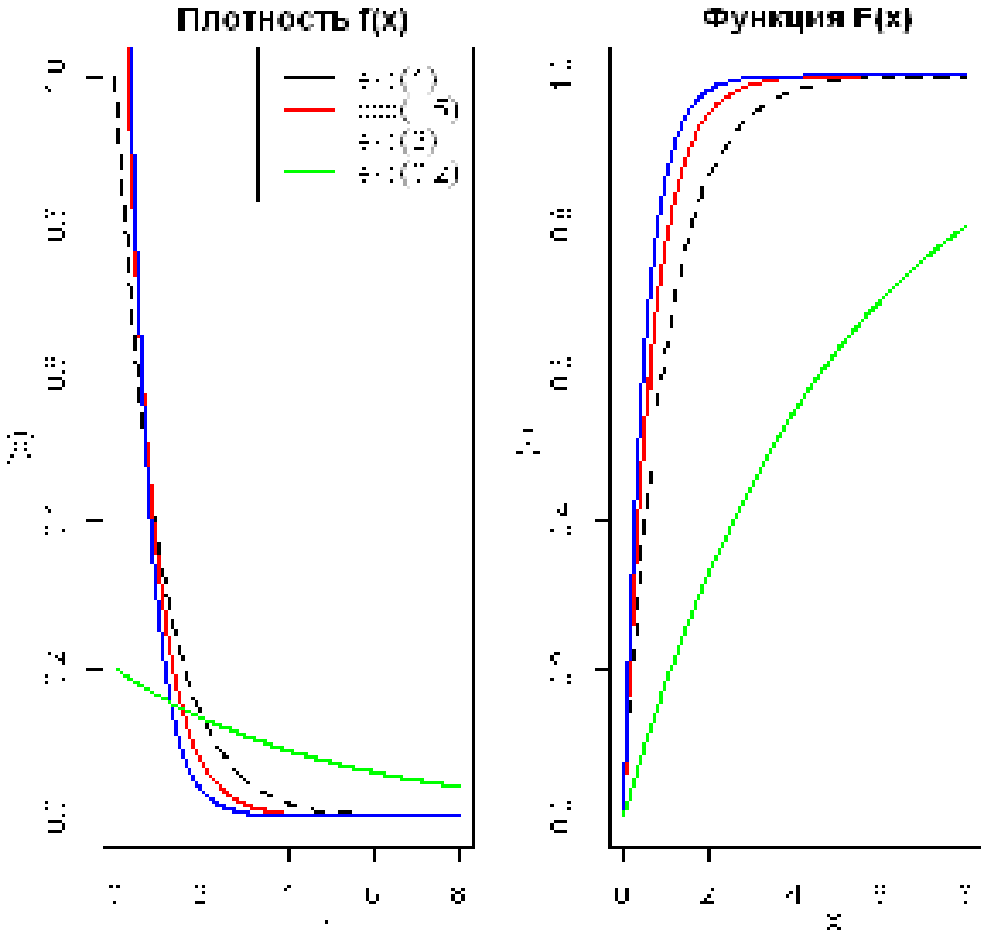



Рисунок 5.10 – Графики плотности и функции показательного распределения

Создадим выборку из 30 элементов, распределенную по показательному закону при $\lambda = 2$.

### *Упражнение 16*

1. Командой `x<-rexp(30,2)` создайте выборку, распределенную по показательному закону.

2. Командой `x` просмотрите результат.

Результат вычислений будет следующим:

```
0.3680195086  0.3564453530  0.1788408100  0.3658250541
0.3017233138  0.3577085697  0.5927761346  0.3538332481
1.0136041212  0.0264680095  0.0158238760  0.1003051945
0.3471427546  0.0009637689  0.1696047061  0.1948571068
0.2825103144 0.7492385627 0.8461298896 0.2595429101
0.3827949875  0.3355488973  0.5144723141  0.9912239350
0.1945454765  0.8566622315  1.1314320014  1.4980733766
0.3460202166 0.5936699849
```

## 5.3 Самостоятельная работа

1) Постройте в одном графическом окне графики плотности и функции распределения нормального закона для нижеуказанных значений $a$ и $\sigma$.

| | Варианты | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| $a$ | 1 | 5,2 | 2,7 | 2 | 1,5 | 1,9 | 7 | 4,3 | 10 | 9 | 21 | 17 |
| $\sigma$ | 0,5 | 1,4 | 3,1 | 1,2 | 1 | 5 | 2,9 | 2,6 | 1,2 | 2 | 5,2 | 2 |
| | Варианты | | | | | | | | | | | |
| | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 |
| $a$ | 82 | 45 | 39 | 19 | 64 | 7,1 | 15 | 34 | 58 | 55 | 78 | 61 | 81 |
| $\sigma$ | 1,5 | 2,4 | 2,3 | 5,1 | 1,8 | 1,1 | 2,1 | 3,1 | 3,5 | 2,3 | 3,5 | 4,5 | 3,4 |

2) Создайте выборку из $n$ элементов, распределенную по закону $\chi^2$ распределения с $k$ степенями свободы.

| | Варианты | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| $n$ | 25 | 35 | 40 | 40 | 40 | 35 | 35 | 35 | 30 | 30 | 50 | 50 |
| $k$ | 5 | 7 | 4 | 14 | 8 | 9 | 11 | 5 | 4 | 15 | 12 | 5 |
| | Варианты | | | | | | | | | | | |
| | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 |
| $n$ | 50 | 50 | 50 | 60 | 60 | 60 | 60 | 60 | 60 | 25 | 25 | 70 | 65 |
| $k$ | 7 | 9 | 15 | 7 | 8 | 5 | 12 | 10 | 7 | 4 | 3 | 6 | 8 |

3) Постройте в одном графическом окне графики плотности и функции распределения равномерного закона для нижеуказанных значений *a* и *b*.

| | Варианты | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| *a* | 7 | 2,1 | 6,5 | 5,7 | 6,5 | 5,9 | 1,9 | 2,8 | 3,9 | 4,8 | 6,4 | 5,2 |
| *b* | 15,1 | 5,4 | 12,4 | 13 | 14 | 13,5 | 7,8 | 16,4 | 9,2 | 11,8 | 14,4 | 32 |
| | Варианты | | | | | | | | | | | |
| | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 |
| *a* | 10 | 11 | 11 | 12 | 7,9 | 14 | 13 | 14 | 8,6 | 21 | 55 | 26 | 19 |
| *b* | 20 | 26 | 34 | 28 | 25 | 35 | 29 | 35 | 35 | 55 | 91 | 42 | 35 |

4) Создайте выборку из 50 элементов, распределенную по закону равномерного распределения по данным из предыдущей задачи.

## 5.4 Контрольные вопросы

1) Дайте определение функции распределения.

2) Дайте определение плотности распределения вероятностей.

3) Дайте определение закона распределения.

4) Какие функции в R используются для задания биномиального закона распределения?

5) Какие функции в R используются для задания пуассоновского закона распределения?

6) Дайте определение закона равномерного распределения.

7) Какие функции в R используются для задания равномерного закона распределения?

8) Дайте определение нормального закона распределения.

9) Какие функции в R используются для задания нормального закона распределения?

10) Дайте определение закона распределения Стьюдента с *n* степенями.

11) Какие функции в R используются для задания Стьюдента с *n* степенями?

12) Дайте определение закона равномерного $\chi^2$ распределения.

13) Какие функции в R используются для задания $\chi^2$ закона распределения?

14) Дайте определение закона показательного распределения.

15) Какие функции в R используются для задания закона показательного распределения?

# Лабораторная работа № 6
# Обработка данных − выделение категорий. Построение вариационного ряда

**Цель работы**: научиться преобразовывать выборку в интервальный вариационный ряд и строить полигоны, гистограммы и график эмпирического распределения.

**Оборудование**: ПК, оснащенный необходимым программным обеспечением.

**Объём лабораторной работы и порядок её выполнения**. Лабораторная работа выполняется в течение двух академических часов. Изучите теоретическую часть, выполните работу и упражнения для самостоятельной работы.

## 6.1 Теоретическая часть

При оценке характеристик выборки часто переходят к построению интервального вариационного ряда [10−13].

**Определение.** *Интервальным вариационным рядом* называется упорядоченная совокупность интервалов варьирования значений случайной величины с соответствующими частотами попаданий в каждый из них значений величины.

Значения выборки после сортировки разбивают на несколько промежутков $[c_{i-1}; c_i)$, $i=1,2,\ldots,k$. Для расчета количества промежутков k можно использовать различные формулы для объёма выборки $n$, но в данной работе будем применять формулу Стерджесса [10]:

$$k \approx 1+3{,}2lg(n).$$

Для вычисления в R по этой формуле следует применять формулу

```
k=ceiling(1+3.2*log(n)/log(10)).
```

После вычисления ширины промежутков $h=R/k$, где $R = \max(x)-\min(x)$ − выборочный размах, границы промежутков $c_i$ вычисляют по рекуррентным формулам: $c_0=\min(x)$, $c_i=c_{i-1}+h$, $i=1,2,\ldots,k$. Затем вычисляют середины интервалов $z_i=(c_i+c_{i-1})/2$.

Находят частоты $n_i$ − число вариант, попавших в $i$-й интервал. По существу, вместо выборки рассматривают статистическое распределение выборки.

**Определение.** *Статистическим распределением выборки* называют перечень вариант $z_i$ *вариационного ряда* и соответствующих им частот $n_i$ или *относительных частот* $w_i = n_i/n$.

Полученные результаты сводят в таблицу.

| $[c_{i-1}; c_i)$ | $[c_0; c_1)$ | $[c_1; c_2)$ | … | $[c_{k-1}; c_k)$ |
|---|---|---|---|---|
| $z_i$ | $z_1$ | $z_2$ | … | $z_k$ |
| $n_i$ | $n_1$ | $n_2$ | … | $n_k$ |
| $w_i$ | $w_1$ | $w_2$ | … | $w_k$ |

Подобное преобразование в R осуществляется преобразованием числовых данных в категориальные, или созданием таблиц данных. В первой лабораторной работе уже произошло ваше знакомство с одной из таких функций `factor()`.

Однако для вышеизложенной задачи воспользуемся функцией `cut()`:

```
cut(x, breaks, labels = NULL, include.lowest = FALSE,
right = TRUE, dig.lab = 3, ordered_result = FALSE, ...)
```

где `x` – числовой вектор, преобразуемый в факторы;

`breaks` – либо число, не меньшее двух, задающее количество интервалов разбиения, либо числовой вектор, задающий границы (точки) разбиения; `labels` – символьный вектор – вектор названий создаваемых категорий (по умолчанию в качестве названий категорий берутся сами интервалы разбиения «(a,b]»);

`include.lowest` – логический аргумент – нужно ли элемент $x_i$ вектора `x`, совпадающий с нижней (верхней) границей одного из интервалов разбиения, включать в этот интервал;

`right` – логический аргумент – задаёт вид интервала разбиения: (a,b] (по умолчанию) или [a,b);

`dig.lab` – числовой аргумент – число знаков после запятой в названии категорий (если названия категорий – интервалы разбиения);

`ordered_result` – логический аргумент – нужно ли упорядочить созданные категории [3, 5].

**Примечание.** Если `breaks` – число, то создаются интервалы разбиения одинаковой длины, причём нижняя граница первого интервала и верхняя граница последнего сдвигаются в сторону уменьшения (увеличения) на 0,1%, чтобы наименьшее и наибольшее значения вектора `x` попали в интервалы разбиения.

**Примечание.** Если `breaks` – числовой вектор (точки разбиения), то длина вектора labels названий категорий разбиения должна быть на единицу меньше длины вектора breaks. [3, 5].

Для визуального отображения характеристик выборки используют полигоны и гистограммы частот $n_i$ и относительных частот $w_i$ [15].

**Определение.** Полигоном частот называют ломанную, отрезки которой соединяют точки $(z_1;n_1)$, $(z_2;n_2)$, ..., $(z_k;n_k)$.

**Определение.** Полигоном относительных частот называют ломанную, отрезки которой соединяют точки $(z_1;w_1)$, $(z_2;w_2)$, ..., $(z_k;w_k)$.

**Определение.** Гистограммой частот называют ступенчатую фигуру, состоящую из прямоугольников, основаниями которых служат частичные интервалы длиной h, а высоты равны отношению $n_i/h$ (плотность частоты).

**Определение.** Гистограммой относительных частот называют ступенчатую фигуру, состоящую из прямоугольников, основаниями которых служат частичные интервалы длиной h, а высоты равны отношению $w_i/h$ (плотность относительной частоты).

Построение гистограммы происходит при помощи следующей функции:

```
hist(x, breaks = "Sturges", freq = NULL,
probability = !freq, include.lowest = TRUE,
right = TRUE, density = NULL, angle = 45, col = NULL, border = NULL, main = paste("Histogram of" , xname), xlim = range(breaks), ylim = NULL, xlab = xname, ylab, axes = TRUE, plot = TRUE, labels = FALSE,
nclass = NULL, ...)
```

Рассмотрим её аргументы:

`x` – числовой вектор (выборка), по которому строится гистограмма.

`breaks` – параметр, отвечающий за разбиение на интервалы. Возможны следующие варианты:

– числовой вектор, определяющий границы интервалов разбиения;

– число, задающее количество интервалов разбиения;

– символьная переменная, отвечающая за выбор алгоритма разбиения на интервалы;

– функция, вычисляющая количество интервалов разбиения.

`freq` и `probability` – два логических и альтернативных друг другу аргумента(не рекомендуется задавать одновременно оба этих аргумента). Если `freq = TRUE` (или, соответственно, `probability = FALSE`), то строится гистограмма частот. Если же `freq = FALSE` (или, соответственно, `probability = TRUE`) – гистограмма относительных частот. Если аргумент probability не задан и интервалы разбиения одинаковые, то по умолчанию строится гистограмма частот, т.е. `freq = TRUE`.

`include.lowest` – логический аргумент. Минимальный элемент выборки включается в первый интервал в качестве левой границы. Используется только в том случае, если `breaks` – числовой вектор. В противном случае выдаётся предупреждение.

`right` – логический аргумент. Если `right = TRUE`, то промежутки разбиения имеют вид $(a_i;a_{i+1}]$.

`density`и`angle` – числовые аргументы, отвечающие за плотность и угол наклона штриховки столбцов гистограммы.

`col` – символьный или числовой аргумент, задающий либо цвет столбцов гистограммы (если не определены аргументы density и angle), либо цвет штриховки. По умолчанию значение `col = NULL`, т.е. столбцы гистограммы не закрашены.

`border` – цвет границы столбцов гистограммы.

`main` – основной заголовок гистограммы.

`xlim` и `ylim` – границы осей гистограммы. По умолчанию для оси X в качестве границ используются границы разбиения, а границы оси У – отсутствуют.

`xlab` и `ylab` – названия осей.

`plot` – логический аргумент – отвечает за построение гистограммы. Если `plot = TRUE`, то строится гистограмма. В противном случае функция `hist()` возвращает список из интервалов разбиения и числа элементов выборки, попавших в интервалы. В последнем случае выдаётся предупреждение, если были определены графические аргументы при задании функции `hist()`.

`Labels` – логический или символьный аргумент. Если `labels=TRUE`, то над каждым столбцом гистограммы выводится число (или доля) элементов выборки, попавших в данный интервал разбиения. Если `labels` – символьный аргумент (вектор), то над столбцами выводятся элементы этого аргумента.

`nclass` – положительное целое число. Аргумент отвечает за количество интервалов разбиения

`...` – дополнительные графические аргументы, отвечающие за заголовки и оси.

Немаловажной характеристикой выборки является эмпирическая функция распределения $F(x)$ и плотность распределения $f(x)$.

**Определение.** Эмпирической функцией распределения (функцией распределения выборки) называют функцию $F(x)$, определяющую для каждого значения $x$ относительную частоту события $X<x$.

Итак, по определению $F(x)=n_x/n$, где $n_x$ – количество элементов выборки, которые меньше $x$.

В R для создания эмпирической функции распределения i($x$) применяют функцию `ecdf()`. Сама по себе функция не строит график эмпирической функции распределения, для его построения применяют функцию `plot.ecdf(x)`.

Для выполнения этой лабораторной работы нужно рассмотреть графические функции низкого уровня. Первая из них функция `rug(x)` рисует данные переменной x на оси 0x как маленькие вертикальные линии. Вторая функция `lines(x, y)` рисует ломанную по парам точек ($x_k$;$y_k$) из векторов `x` и `y`. Функция `sprintf(fmt, ...)` создает текст в C-style (можно использовать `%s`, `%.5f` и т.п.).

## 6.2 Выполнение лабораторной работы

Решим задачу построения интервального вариационного ряда нижеприведенной выборки:

97,5 98,6 99,4 99,7 100,1 100,4 100,7 101,5 97,6 98,7 99,4 99,7 100,1 100,4 101,0 101,6 98,4 99,2 99,4 99,7 100,2 100,6 101,2 102,4 98,5 99,2 99,5 99,8 100,3 100,7 101,2 102,7 98,5 99,3 99,6 99,9 100,3 100,7 101,3 102,8.

### *Упражнение 1*

1) Нижеприведенной командой введите выборку в переменную `x`:

```
x<-c(97.5,98.6,99.4,99.7,100.1,100.4,100.7,101.5,97.6,
+98.7,99.4,99.7,100.1,100.4,101.0,101.6,98.4,99.2,99.4,
+99.7,100.2,100.6,101.2,102.4,98.5,99.2,99.5,99.8,
+100.3,100.7,101.2,102.7,98.5,99.3,99.6,99.9,100.3,
+100.7,101.3,102.8)
```

2) Командой `x` просмотрите значения переменной `x`.

3) Командой `R<-max(x)-min(x)` вычислите размах выборки.

4) Командой `R` просмотрите значение `R`.

5) Командой `n<-length(x)` найдите объём выборки.

6) Командой `k<-ceiling(1+3.2*log(n)/log(10))` вычислим количество интервалов.

7) Командой `h<-R/k` вычислим ширины интервалов.

8) Командой `ci<-numeric(k+1)` создадим числовой вектор, значения элементов будут равны границам промежутков.

9) Командой `ci[1]<-min(x)` первому элементу присвоим минимальное значение выборки.

10) Командой `for (j in 1:k){ci[j+1]<-ci[j]+h}` вычислим остальные границы промежутков.

11) Командой `z<-numeric(k)` создадим числовой вектор середин промежутков.

12) Командой `for(j in 1:k) {z[j]<-(ci[j]+ci[j+1])/2}` вычислим значения середин промежутков.

13) Командой `z<-round(z,1)` округлим до первого знака после десятичной запятой.

14) Командой `lab<-as.character(z)` преобразуем числовые значения середин промежутков в строковые значения и запишем в переменную `lab`.

15) Командой `lab` просмотрите полученную переменную.

16) Командой `ci[1]<-ci[1]-0.5` сдвинем левую границу первого интервала влево.

17) Командой `ci[k+1]<-ci[k+1]+0.5` сдвинем правую границу последнего промежутка вправо.

18) Командой `razb<-cut(x,breaks=ci,labels=lab)` создадим фактор (вектор категориальных значений), соответствующий частотам попадания в i-й промежуток $[c_{i-1}; c_i)$.

19) Командой `str(razb)` просмотрите результаты подсчета частот по интервалам.

20) Создадим таблицу данных командой `razbi<-table(razb)`.

21) Командой `razbi` посмотрите полученную таблицу.

Результаты работы последней команды будут следующими:

```
razb
 97.9  98.6  99.4 100.1 100.9 101.7 102.4
    2     5    11     9     7     3     3
```

Результаты этих вычислений лучше представить визуально. Построим гистограмму частот и полигон относительных частот.

### *Упражнение 2*

1) Командой `plot(razb)` построим столбиковую диаграмму частот попадания элементов выборки в промежутки. Результаты построения приведены на рисунке 6.1. Данная диаграмма не является гистограммой частот.

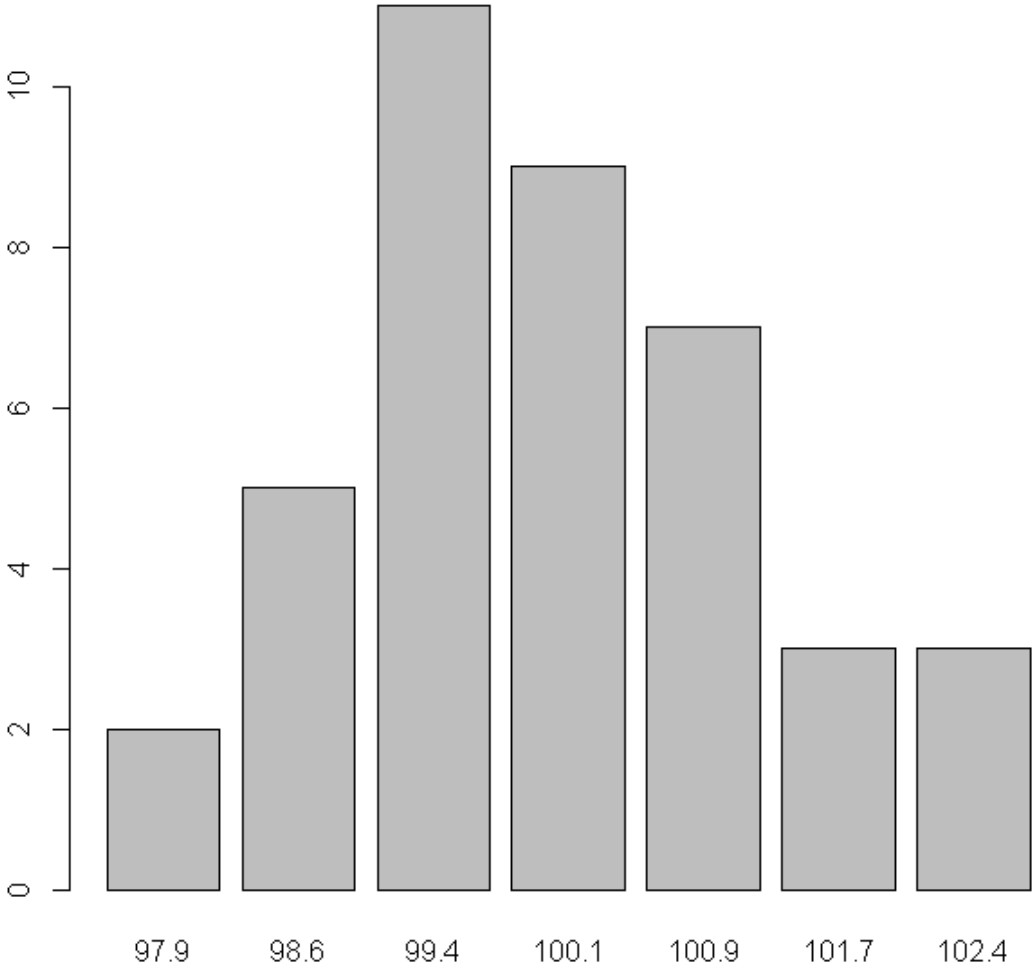



Рисунок 6.1 – Столбиковая диаграмма частот

2) Вычислите значение $n_i/h$ и присвойте эти значения переменной `ra` командой `ra<-razbi/h`.

3) Командой barplot(ra) постройте гистограмму частот. Результат построения представлен на рисунке 6.2.

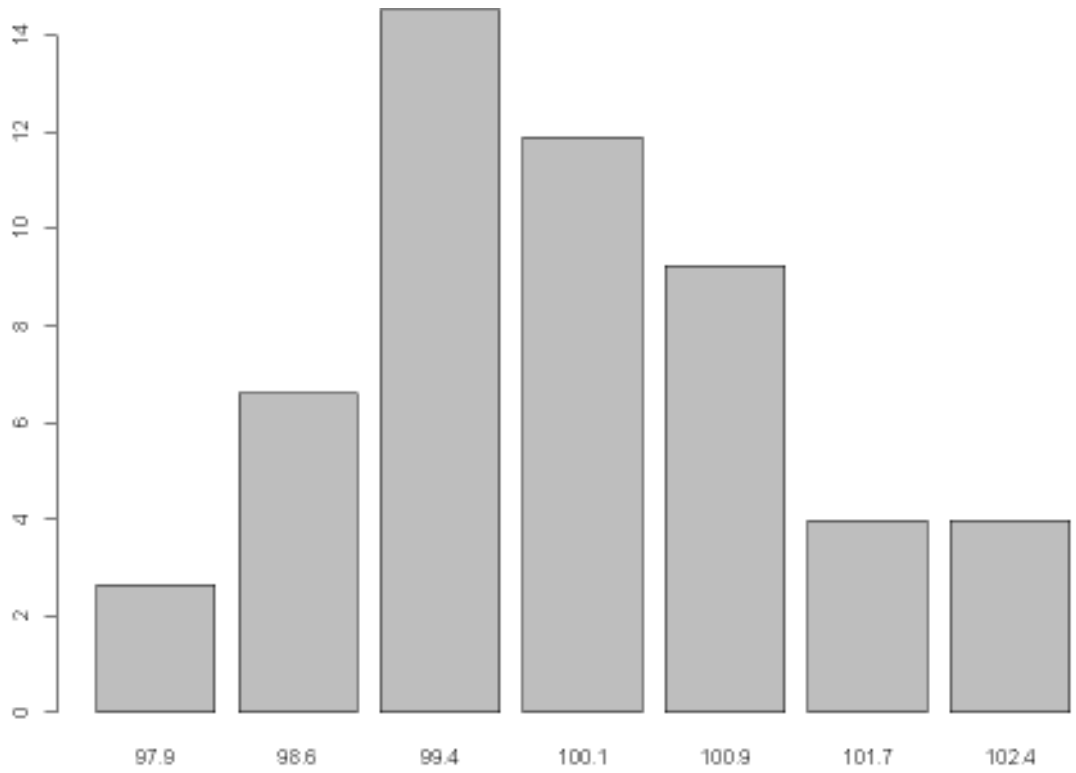



Рисунок 6.2 – Гистограмма частот

4) Командой `W<-razbi/n` вычислите относительные частоты.

5) Командой `W` просмотрите полученные результаты.

6) Командой `windows()` создайте новое графическое окно.

7) Командой `plot(W,type='l',col='blue')` постройте полигон относительных частот. Результат построения представлен на рисунке 6.3.

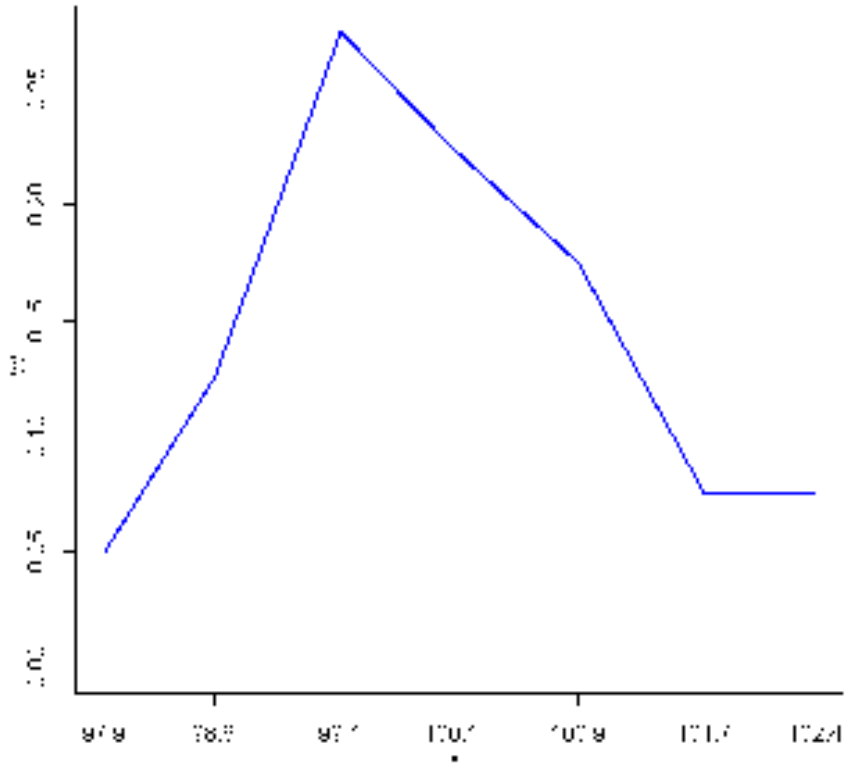


Рисунок 6.3 – Полигон относительных частот

Используем для визуализации обработки статистических данных функцию `hist()`.

### *Упражнение 3*

1) Командой `hist(x,label=TRUE,ylab='Частоты')` постройте гистограмму частот. В графическом окне видно, что выборка разбита на шесть промежутков. Ранее уже было рассчитано, что интервалов должно быть семь.

2) Просмотрите результаты разбиения путем добавления аргумента функции `plot`. Выполните в командной строке новую команду `hist(x, plot=FALSE)`. Результаты выполнения команды будут следующими:

```
$breaks
[1]  97  98  99 100 101 102 103
$counts
[1]  2  5 13 12  5  3
$density
[1] 0.050 0.125 0.325 0.300 0.125 0.075
$mids
[1]  97.5  98.5  99.5 100.5 101.5 102.5
$xname
[1] "x"
$equidist
[1] TRUE
attr(,"class")
[1] "histogram"
```

3) Изменим параметры гистограммы (её цвет и название) командой `hist(x, col='yellow', main='Гистограмма частот', ylab='частоты')`.

Результат работы этой команды представлен на рисунке 6.4.

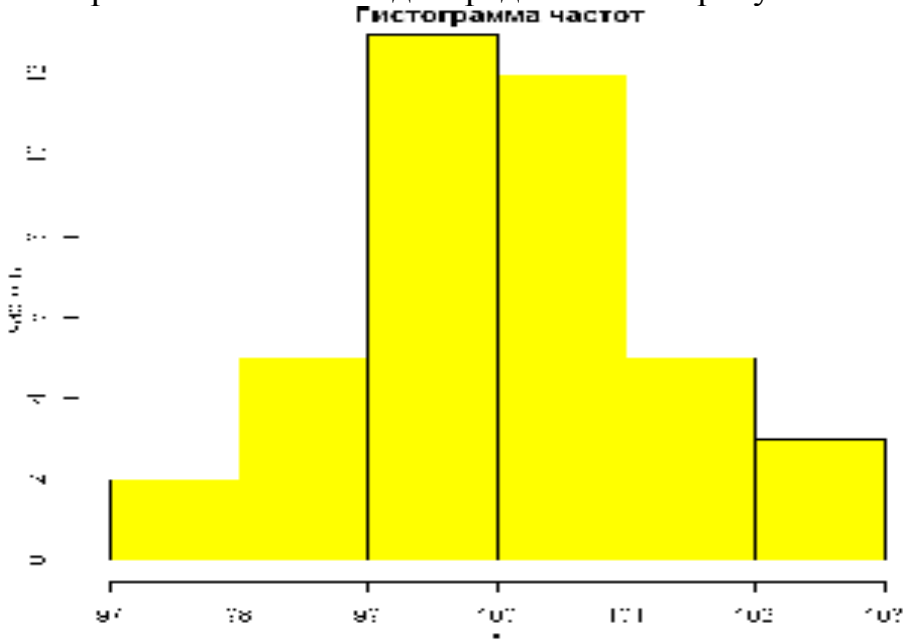



Рисунок 6.4 – Гистограмма частот (шесть интервалов)

Для изменения количества промежутков используем аргумент `breaks`. Сформируем его как числовой вектор, элементы которого являются границами промежутков.

### *Упражнение 4*

1) Для этого создайте скрипт и отправьте его на исполнение.

```
R<-round(max(x))-trunc(min(x))
n<-length(x)
k<-ceiling(1+3.2*log(n)/log(10))
h<-R/k
ci<-numeric(k+1)
ci[1]<-trunc(min(x))
for (j in 1:k){ci[j+1]<-ci[j]+h}
```

2) Сохраните его под именем `lab5_3`.

Результатом работы этого скрипта будет числовой вектор `ci`, элементы которого являются границами промежутков.

### *Упражнение 5*

1) Нижеприведенной командой постройте гистограмму, на которой выборку разбивают на семь промежутков:

```
hist(x, breaks=ci, labels = TRUE, col='orange', ylim=c(0,14), main='Гистограмма частот')
```

2) Результатов работы показаны на рисунке 6.5. Постройте гистограмму относительных частот, используя аргумент `freq`.

3) Командой `ymain<-'Гистограмма относительных частот'` создадим символьную константу. Используем её как название гистограммы.

4) Командой `windows()` создайте новое графическое окно.

5) Постройте гистограмму относительных частот командой `hist(x,breaks=ci,labels=TRUE,col='tomato',freq=FALSE, main=ymain)`.

6) Результатов работы показаны на рисунке 6.6. Сравните обе диаграммы.

7) Закройте оба графических окна.

8) Командой присвойте переменной параметры разбиения выборки на промежутки.

9) Командой `prom` просмотрите значение переменной. Результаты приведены ниже:

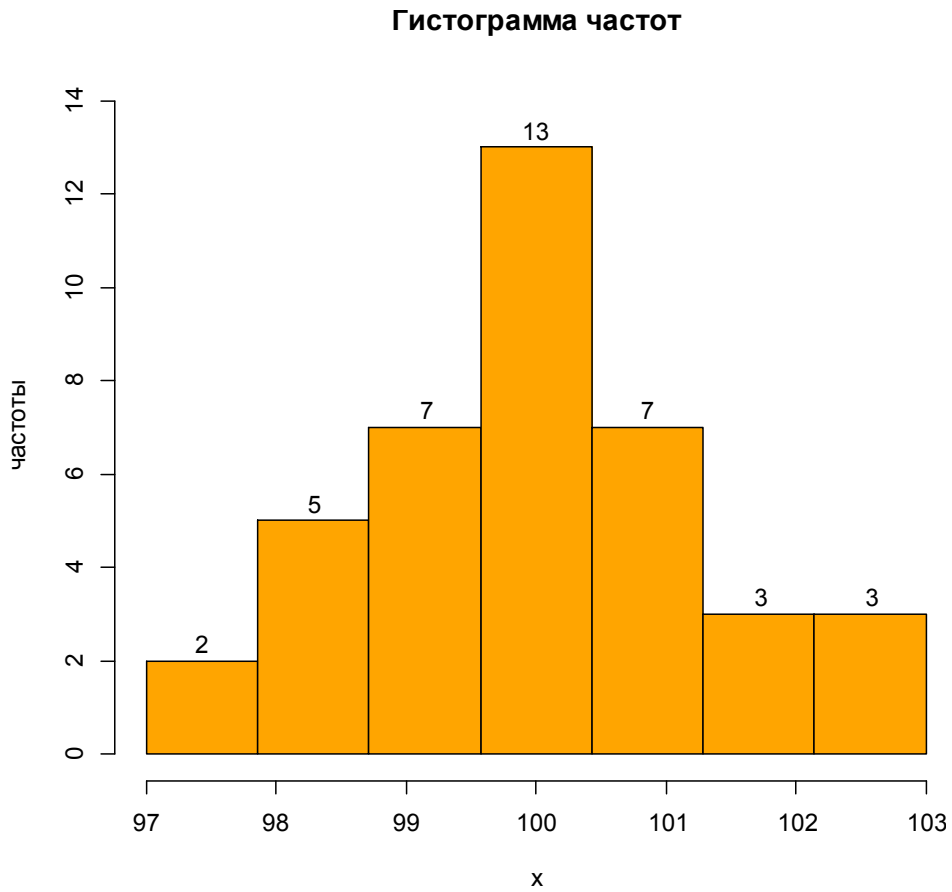



Рисунок 6.5 – Гистограмма частот (семь интервалов)

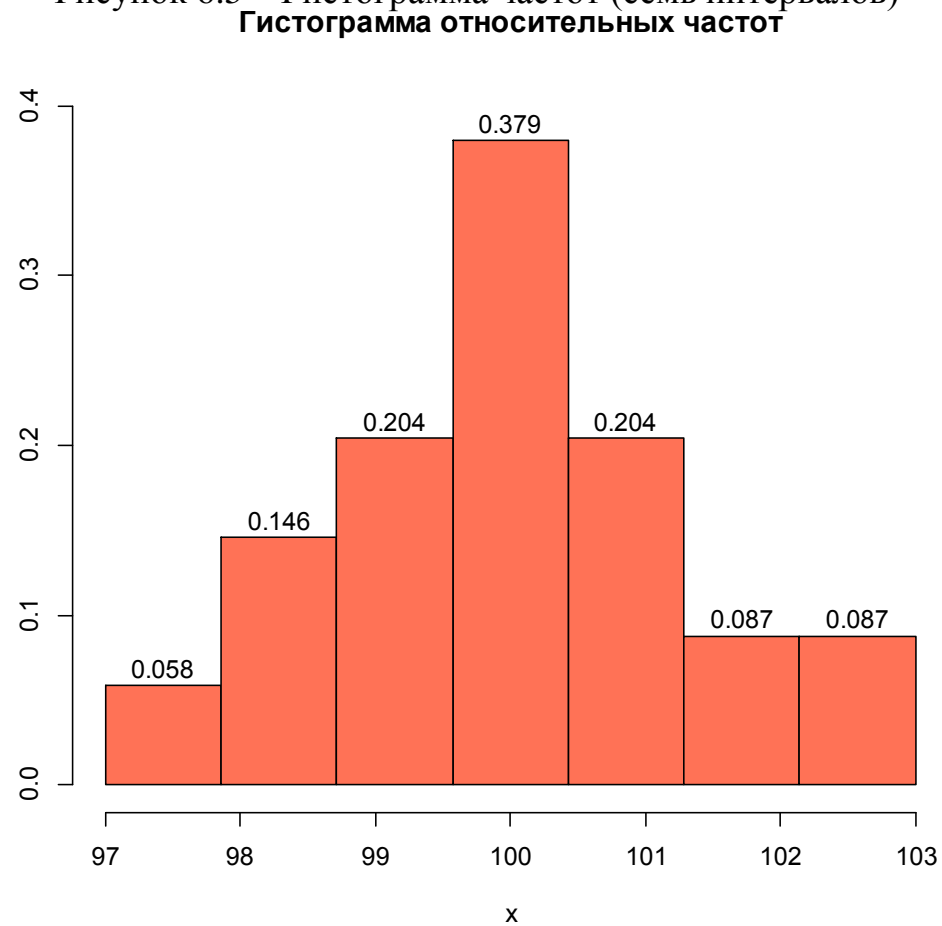



Рисунок 6.6 – Гистограмма относительных частот (семь интервалов)

```
$breaks
[1]    97.00000    97.85714    98.71429    99.57143
100.42857 101.28571 102.14286 103.00000
$counts
[1]  2  5  7 13  7  3  3
$density
[1]  0.05833333  0.14583333  0.20416667  0.37916667
0.20416667 0.08750000 0.08750000
$mids
[1]    97.42857    98.28571    99.14286  100.00000
100.85714 101.71429 102.57143
$xname
[1] "x"
$equidist
[1] TRUE
attr(,"class")
[1] "histogram"
```

Построим в том же самом графическом окне график функции плотности нормального распределения.

### *Упражнение 6*

1) Командами `a1<-mean(x);s1<-sd(x)` вычислите среднее выборочное и уточненное значение среднего квадратического отклонения.

2) Командой `x2 <- seq(a1-3*s1, a1+3*s1, len=n)` создайте вектор значений переменной `x2`.

3) Командой `range(x2)` просмотрите минимальное и максимальное значения переменной `x2`.

4) Командой `f1 <- dnorm(x2, a1, s1)` вычислите значения функции плотности распределения нормального закона.

5) Командой `lines(x2,f1,lwd=2,col='green')` постройте график функции плотности нормального распределения.

6) Командой `ltext <- sprintf("X~N(%.2f, %.2f)",a1,s1)` создайте текстовую переменную, которую поместим в легенду графика.

7) Командой `ltext` просмотрите полученное значение текстовой переменной.

8) Нижеуказанной командой поместите легенду в верхний левый край графического окна:
`legend("topleft", lty=1, legend=paste("f(x):",ltext)).`

9) Командой `rug(x)` постройте данные переменной x на оси 0x как маленькие вертикальные линии.

Сравните результат построения с рисунком 6.7.

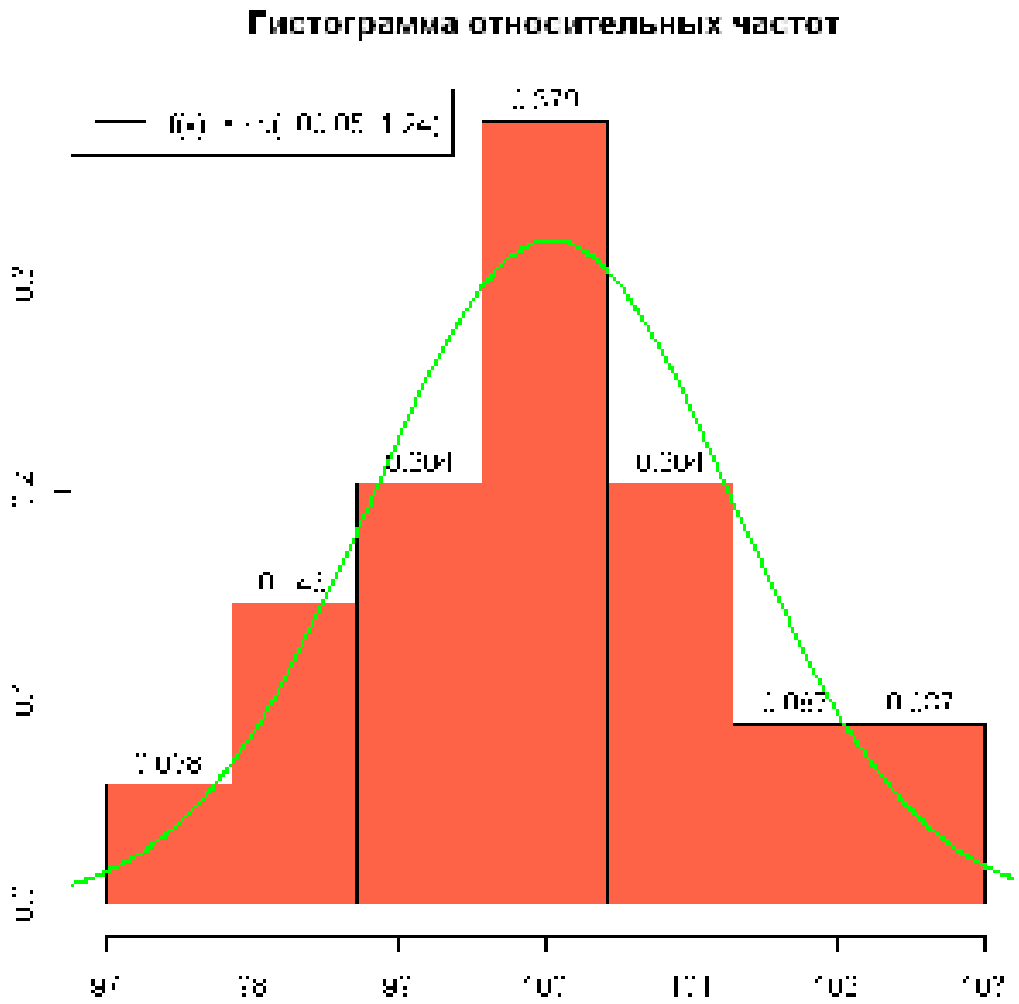



Рисунок 6.7 – Гистограмма относительных частот (семь интервалов)

Построим график эмпирической функции распределения для этой выборки.

### *Упражнение 7*

1) Командой `F1<-pnorm(x2, a1, s1)` создадим вектор значений функции распределения нормального закона.

2) Командой `ymain<-'График эмпирической функции распределения'` создайте символьную константу. Используем её как название гистограммы.

3) Командой `plot.ecdf(x)` постройте график эмпирической функции распределения $F(x)$. Постараемся убрать большую точку перед линиями.

4) Еще раз постройте график эмпирической функции командой `plot.ecdf(x,pch=".",main=ymain,xlab="",ylab="")`.

5) Командой `rug(x)` постройте данные переменной x на оси 0x как маленькие вертикальные линии.

6) Командой `lines(x2, F1, col="red", lwd=2)` постройте график функции распределения нормального закона.

7) Нижеуказанной командой поместите легенду в верхний левый край графического окна:

```
legend("topleft", lty=1, legend=paste("F(x):",ltext))
```

Результаты построения приведены на рисунке 6.8.

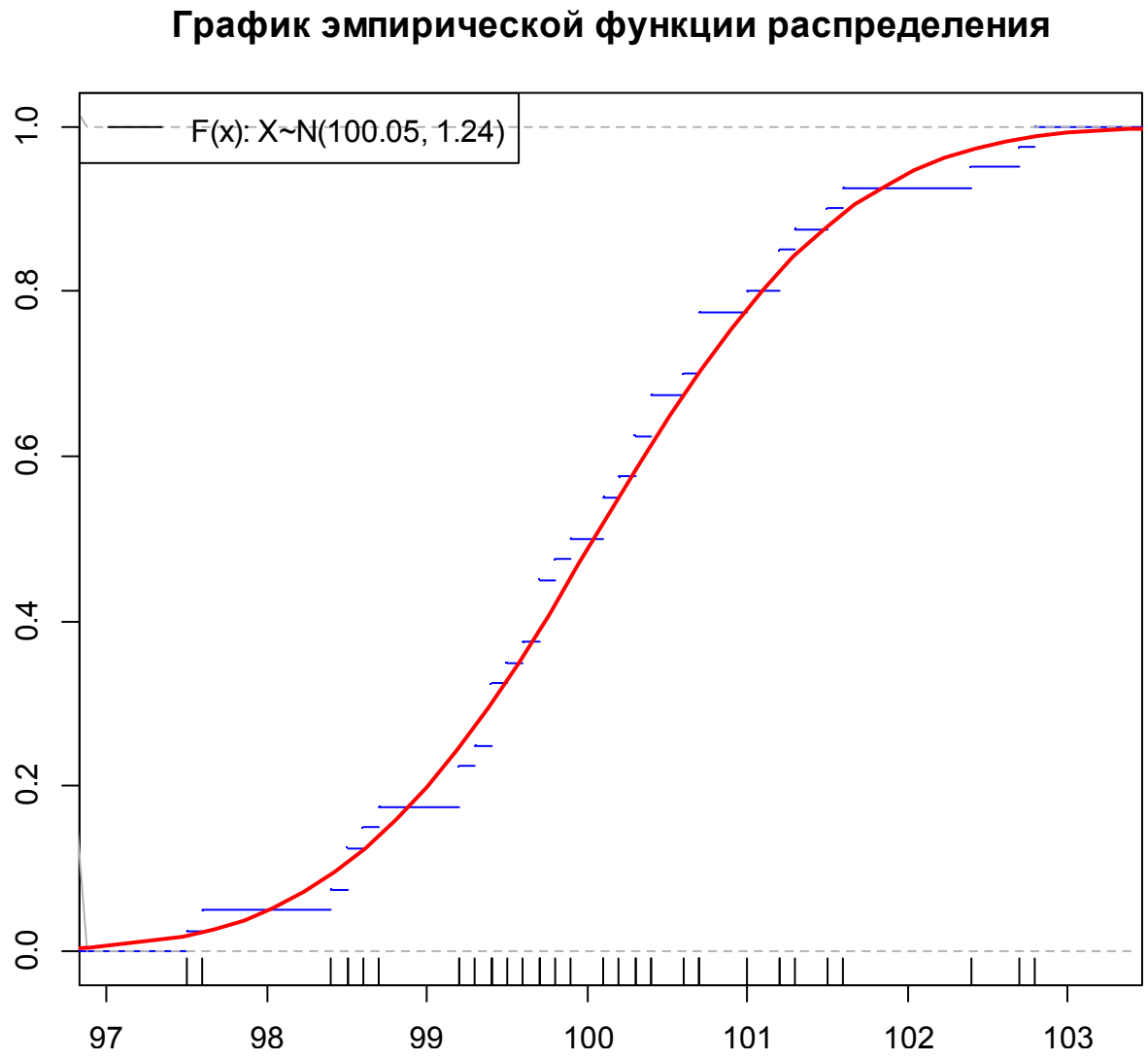



Рисунок 6.8 – График эмпирической функции распределения

### 6.3 Самостоятельная работа

1) Скопируйте в рабочую папку файл из сетевой папки **Учебная на ASHP\R** файл **l63.data** в свою рабочую папку.

2) Командой `d2=read.table("l63.data")` считайте данные из файла в переменную-таблицу данных `d2`.

3) Командой `x<-d2$V1` преобразуйте данные в вектор переменную `x`.

4) Командой `x` просмотрите её значения.

5) Для `x` постройте эмпирическую функцию распределения.

6) Постройте гистограмму частот для этой выборки.

7) Постройте гистограмму относительных частот для этой выборки.

8) Постройте полигоны частот и относительных частот для вашего варианта расчетной работы, сохраните их.

9) Постройте гистограмму относительных частот для вашего варианта расчетной работы и наложите график функции плотности распределения нормального закона, сохраните её.

10) Постройте для вашего варианта расчетной работы график эмпирической функции распределения, наложите на него график функции распределения нормального закона, сохраните его.

## 6.4 Контрольные вопросы

1) Дайте определение статистическим распределением выборки.

2) Дайте определение интервального вариационного ряда.

3) Как вычисляют количество интервалов при создании интервального вариационного ряда?

4) Каким образом в R выборка преобразуется в интервальный вариационный ряд?

5) Дайте определение полигона частот.

6) Дайте определение полигона относительных частот.

7) Дайте определение гистограммы частот.

8) Дайте определение гистограммы относительных частот.

9) При помощи какой функции в R строится гистограмма?

10) Дайте определение эмпирической функции распределения.

11) При помощи какой функции в R строится график эмпирической функции распределения?

# Лабораторная работа № 7
# Проверка статистических гипотез

**Цель работы**: научиться использовать встроенные функции системы R по проверке статистических гипотез.

**Оборудование**: ПК, оснащенный необходимым программным обеспечением.

**Объём лабораторной работы и порядок её выполнения**. Лабораторная работа выполняется в течение двух академических часов. Изучите теоретическую часть, выполните работу и упражнения для самостоятельной работы.

## 7.1 Теоретическая часть

**Определение.** ***Статистическая гипотеза*** − это некоторое предположение относительно генеральной совокупности, проверяемое по выборочным данным.

**Определение.** ***Нулевой*** (***основной***) называют выдвинутую гипотезу $H_0$.

**Определение.** ***Конкурирующей*** (***альтернативной***) называют гипотезу $H_1$, которая противоречит нулевой.

**Определение.** ***Статистическим критерием***, или ***критерием*** называют случайную величину $K$, которая служит для проверки нулевой гипотезы.

**Определение.** ***Критической областью*** называют совокупность значений критерия, при которой нулевую гипотезу отвергают.

**Определение.** ***Областью принятия гипотезы*** называют совокупность значения критерия, при которых гипотезу принимают.

**Определение.** ***Критическими точками***, или ***границами*** $k_{кр}$ называют точки, отделяющие критическую область от области принятия гипотез.

**Определение.** Вероятность допустить ошибку первого рода называется ***уровнем значимости критерия*** [17].

**Определение.** ***Мощностью критерия*** называют вероятность попадания критерия в критическую область при условии, что справедлива конкурирующая гипотеза.

**Определение.** ***Надежностью критерия*** называется вероятность 1-P($A$), то есть вероятность того, что $H_0$ будет принята, если она верна [17].

Процесс проверки гипотезы состоит из следующих этапов:

1) выбирается статистический критерий $K$;
2) по имеющейся выборке вычисляется наблюдаемое значение $K_{набл}$;
3) поскольку закон распределения $K$ известен, то по заданному уровню значимости $\alpha$ определяется критическая точка (или критические точки) $k_{кр}$, затем устанавливается вид критической области W;

4) если вычисленное значение $K_{набл}$ попадает в область принятия гипотезы, то нулевая гипотеза принимается, если в критическую область – нулевая гипотеза отвергается [17].

**Определение.** ***Критерием согласия*** называется статистический критерий проверки гипотезы о предполагаемом законе неизвестного распределения.

В R реализованы функции для различных критериев.

## Тест Шапиро-Уилка

Нулевая гипотеза $H_0$ теста Шапиро-Уилка заключается в том, что случайная величина, выборка `x` которой известна, распределена по нормальному закону. Альтернативная гипотеза $H_1$ заключается в том, что закон распределения не является нормальным.

Для проверки нулевой гипотезы по тесту Шапиро-Уилка применяется функция `shapiro.test(x)` (3<x<5000). Результатом её работы будет список со следующими значениями:

– `statistic` – значение статистики теста, которую принято обозначать буквой `W`;

– `p.value` – аппроксимация p-value для полученного значения статистики;

– `method` – строка с названием теста;

– `data.name` – имя переменной, содержащей выборку, являющейся аргументом функции shapiro.test.

## Критерий согласия $\chi^2$ Пирсона

Критерий согласия $\chi^2$ Пирсона применяется для проверки гипотезы о том, что случайная величина имеет заданное распределение, т.е. $H_0$: «Случайная величина X имеет функцию распределения $F(x)$», а также гипотезы о независимости двух признаков, т.е. $H_0$: «X не зависит от Y». Для выполнения данного статистического теста предназначена функция

```
chisq.test(x, y = NULL, correct = TRUE,
p=rep(1/length(x),length(x)), rescale.p= FALSE,
simulate.p.value = FALSE, B = 2000),
```

где

– `x` – числовой вектор, представляющий собой статистический ряд, как фактор, содержащий выборку значений случайной величины X, или матрица сопряженности признаков ;

– `y` – если `x` является фактором, то y также должен быть фактором такой же длины, содержащей выборку значений случайной величины Y, таким образом, $(x_i,y_i)$ – реализация двумерной дискретной случайной величины. По

x и y вычисляется таблица сопряженности. В остальных случаях данный параметр игнорируется;

– correct – логическое значение, которое указывает, требуется ли применять непрерывную коррекцию для матриц размерностью 2×2;

– p – вектор, содержащий вероятности попадания в интервалы разбиений значений случайных величин должен иметь такую же длину, что и x;

– rescale.p – логическое значение. Если TRUE, то p при необходимости нормируется так, чтобы сумма его компонентов была равна 1;

– simulate.p.value – логическое значение, определяющее будет ли для вычисления p-value использоваться метод Монте-Карло (simulate.p.value = TRUE), или $\chi^2$ распределение;

– B – количество испытаний в методе Монте-Карло.

Результат работы функции chisq.test будет выведен в виде списка полей:

– statistics – значение $\chi^2$ – статистики Пирсона;

– parameter – число степеней свободы распределения $\chi^2$ (равно NA, если для отыскания p-value использовался метод Монте-Карло);

– p-value – аппроксимация p-value для полученного значения статистики;

– method – символьная строка с названием используемой модификации теста, а также с указанием того, использовались ли непрерывная коррекция и метод Монте-Карло;

– data.name – строка, содержащая имя данных, подвергнутых тесту;

– observed – число точек, попавших в *i*-й интервал группировки;

– expected – теоретическое число точек (в предположении выполнения гипотезы), попадающих в *i*-й интервал группировки;

– residuals – остатки Пирсона:

$$\text{residuals} = \frac{\text{observed} - \text{expected}}{\sqrt{\text{expected}}}.$$

### Критерий Колмогорова-Смирнова

Тест Колмогорова-Смирнова для одной или двух выборок. Критерий согласия Колмогорова используется для проверки простой гипотезы о том, подчиняется ли данное эмпирическое распределение точно известной теоретической модели. Критерий однородности Смирнова предназначен для проверки гипотезы о том, подчиняются ли два эмпирических распределения одному и тому же закону.

Его синтаксис критерия в R:

```
ks.test(x, y, ...,alternative = c("two.sided", "less",
                "greater"),exact = NULL)
```

Аргументы функции:

– x – вектор, содержащий выборку.

– y – вектор, содержащий вторую выборку, или символьная строка с именем распределения.

– ... – параметры распределения.

– `alternative` – символьный аргумент, обозначающий тип альтернативной гипотезы. Принимает одно из следующих значений: `"two.sided"` (по умолчанию), `"less"` или `"greater"`.

– `exact` – `NULL` или логическое значение, обозначающее требование точного вычисления `p-value`. Не используется в двувыборочном тесте, если `alternative="less"` или `alternative = "greater"`.

Точное значение p-value не вычисляется в двувыборочном тесте, если `alternative = "less"` или `alternative = "greater"`. В одновыборочном тесте параметры гипотетического распределения должны быть известны точно, а не вычисляться по выборке x. Вариант теста Колмогорова-Смирнова с оценкой параметров не поддерживается. [5]

Объект, возвращаемый функцией `ks.test()`,– это список со следующими полями:

– `statistics` – значение статистики Колмогорова-Смирнова;

– `p.value` – аппроксимация p-value для полученного значения статистики;

– `alternative` – символьный аргумент – описание альтернативной гипотезы;

– `method` – символьный аргумент – название используемого метода;

– `data.name` – название массива данных, подвергнутых тесту [3–5].

## t-тест Стьюдента

Выполняется одно или двувыборочный тест Стьюдента. Одновыборочный t-тест предназначен для проверки равенства среднего значения выборки из нормально распределённой генеральной совокупности некоторому заданному значению в предположении, что дисперсия неизвестна.

Двувыборочный тест служит для сравнения двух средних значений выборок из нормально распределённых генеральных совокупностей в предположении, что их дисперсии равны, хотя и неизвестны.

Функции теста имеет три синтаксиса: `t.test(x, ...)`

или

```
t.test(x, y = NULL, alternative = c("two.sided",
"less", "greater"), mu = 0, paired = FALSE,
var.equal = FALSE, conf.level = 0.95, ...)
```

или `t.test(formula, data, subset, na.action, ...)`

Её аргументы:

– `x` – числовой вектор, содержащий элементы первой выборки.

– `y` – числовой вектор, содержащий элементы второй выборки.

– alternative – одно из следующих значений: "two.side" (по умолчанию), "less", "greater", обозначающих тип альтернативной гипотезы;

– conf.level – доверительный уровень для возвращаемого доверительного интервала;

– exact – либо NULL, либо логический аргумент. Отвечает за точное вычисление p-value. Не используется в двувыборочном тесте, если alternative = "less" или alternative = "greater";

– mu – математическое ожидание или разность математических ожиданий, если задано две выборки;

– paired – логическое значение, указывающее, требуется ли выполнить спаренный t-тест;

– var.equal – логическое значение, считающее, считать ли разбросы равными. Если TRUE, то вычисляется разброс для объединённой выборки;

– conf.level – доверительный уровень для возвращаемого доверительного интервала;

– formula – формула вида lhs~rhs, где lhs – числовой вектор, а rhs – фактор с двумя классами;

– data – матрица или фрейм данных, из которых берутся данные для formula;

– subset – вектор, определяющий используемое подмножество наблюдений;

– na.action – функция, которая вызывается, как только в данных встретилось значение NA.

Объект, возвращаемый функцией t.test, – это список со следующими полями:

– statistics – значение t-статистики Стьюдента;

– parameter – число степеней свободы;

– p.value – аппроксимация p-value для полученного значения статистики;

– conf.int – доверительный интервал для математического ожидания;

– estimate – оценка математического ожидания для одновыборочного теста или разности математических ожиданий для двувыборочного теста;

– null.value – предполагаемое математическое ожидание или разность предполагаемых математических ожиданий для двувыборочного теста (входной параметр mu);

– alternative – символьная строка с описанием альтернативной гипотезы;

– method – символьная строка с названием используемой модификации метода;

– data.name – строка, содержащая имя (имена) данных, подвергнутых тесту.

### F-тест Фишера

Выполняется F-тест Фишера для проверки на равенство стандартных отклонений двух нормально распределённых генеральных совокупностей. Функции теста имеет два синтаксиса:

```
var.test(x, y, ratio = 1,
alternative = c("two.sided", "less", "greater"),
conf.level = 0.95, ...)
```

или

```
var.test(formula, data, subset, na.action, ...)
```

Её аргументы:

x, y – числовые векторы, содержащие выборки из разных генеральных совокупностей, или линейные модели;

ratio – предполагаемая величина отношения стандартных отклонений в первой и второй генеральных совокупностях;

alternative – одно из следующих значений: ”two.side” (по умолчанию), ”less”, ”greater”, обозначающих тип альтернативной гипотезы;

conf.level – доверительный уровень для возвращаемого доверительного интервала;

formula – формула вида lhs~rhs, где lhs – числовой вектор, а rhs – фактор с двумя классами;

data – матрица или фрейм данных, из которых берутся данные для formula;

subset – вектор, определяющий используемое подмножество наблюдений;

na.action – функция, которая вызывается, как только в данных встретилось значение NA.

Нулевая гипотеза постулирует, что отношение стандартных отклонений генеральных совокупностей, из которых выбраны x и y соответственно, равно отношению ratio.

Объект, возвращаемый функцией var.test, – это список со следующими полями:

data – используемые выборки (названия переменных, которым присвоены значения выборок);

F – значение F-статистики Фишера;

num df – число степеней свободы.

## 7.2 Выполнение лабораторной работы

Поверим работу теста Шапиро-Уилка на двух выборках, приняв уровень значимости α = 0,05.

### *Упражнение 1*

1) Командой `x<-rnorm(1001,mean=3,sd=4)` создайте выборку.

2) Командой `shapiro.test(x)` проверьте выборку. Результаты работы функции приведены ниже.

```
Shapiro-Wilk normality test
data:  x
W = 0.9976, p-value = 0.1585
```

Значение уровня значимости (`p-value = 0.1585`) больше уровня значимости α (0,1585>0,05). Следовательно, оснований отвергать нулевую гипотезу о нормальном распределении выборки нет.

### *Упражнение 2*

1) Командой `x<-rpois(100,2)` создайте выборку, распределённую по закону Пуассона.

2) Командой `hist(x)` постройте гистограмму и убедитесь, что выборка не распределена по нормальному закону. Результат построения гистограммы приведен на рисунке 7.1.

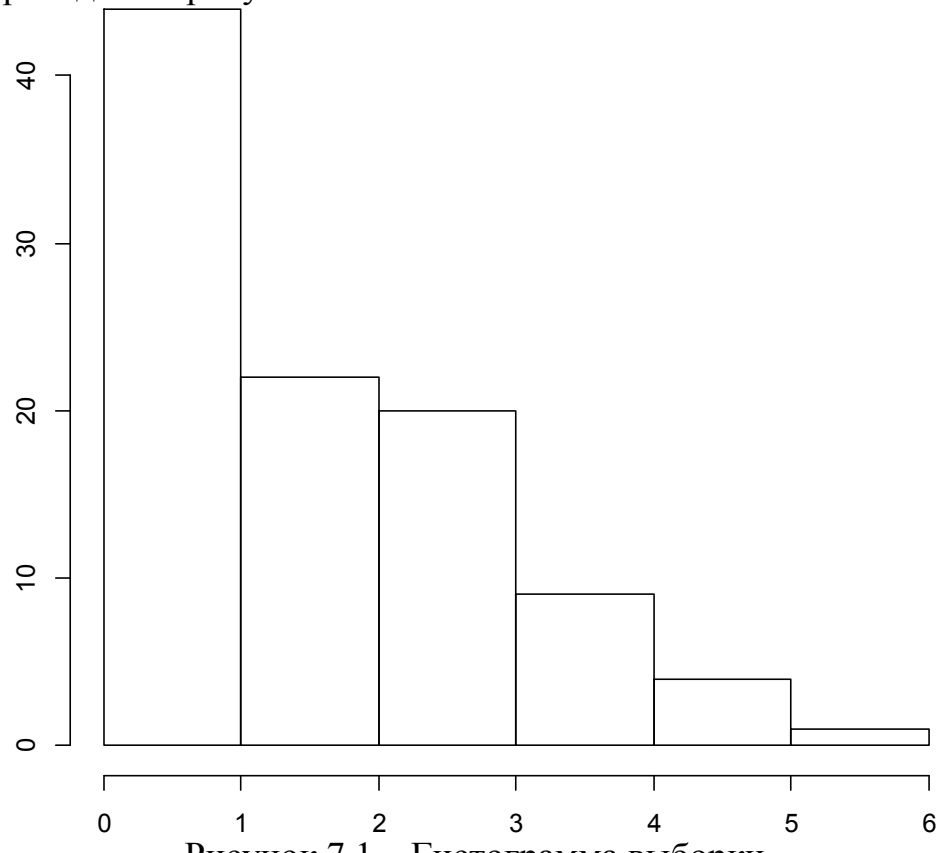



Рисунок 7.1 – Гистограмма выборки

3) Командой проверьте выборку. Результаты работы функции приведены ниже:

```
shapiro.test(x)
Shapiro-Wilk normality test
data:  x
W = 0.9203, p-value = 1.452e-05
```

Значение уровня значимости (`p-value = 1.452e-05`) меньше уровня значимости α ($1{,}452 \cdot 10^{-5} < 0{,}05$). Следовательно, выборка ***не распределена*** по нормальному закону.

Проверим гипотезу о нормальном распределении выборки при помощи критерия согласия $\chi^2$ Пирсона. Разобьём работу на два этапа. В первом загрузим выборку из файла и передадим её значения в переменную-вектор x. На втором этапе создадим скрипт, позволяющий проверить гипотезу о нормальном распределении и просмотреть визуально на диаграмме.

### *Упражнение 3*

1) Скопируйте в рабочую папку файл из сетевой папки **Учебная на ASHP\R** файл **lab7_dan3.txt** в свою рабочую папку.

2) Командой `d2<-read.table("lab7_dan3.txt")` считайте данные из файла в переменную-таблицу данных `d2`.

3) Командой `x<-d2$V1` преобразуйте данные в вектор переменную `x`.

4) Командой `x` просмотрите её значения.

### *Упражнение 4*

1) Создайте скрипт.

2) Запишите в его окно код, приведенный на рисунке 7.2. Обратите внимание на комментарии, начинающиеся с символа #. В них приводится описание команд, которые следуют после него.

3) Отправьте его на исполнение.

4) Проверьте результаты его исполнения.

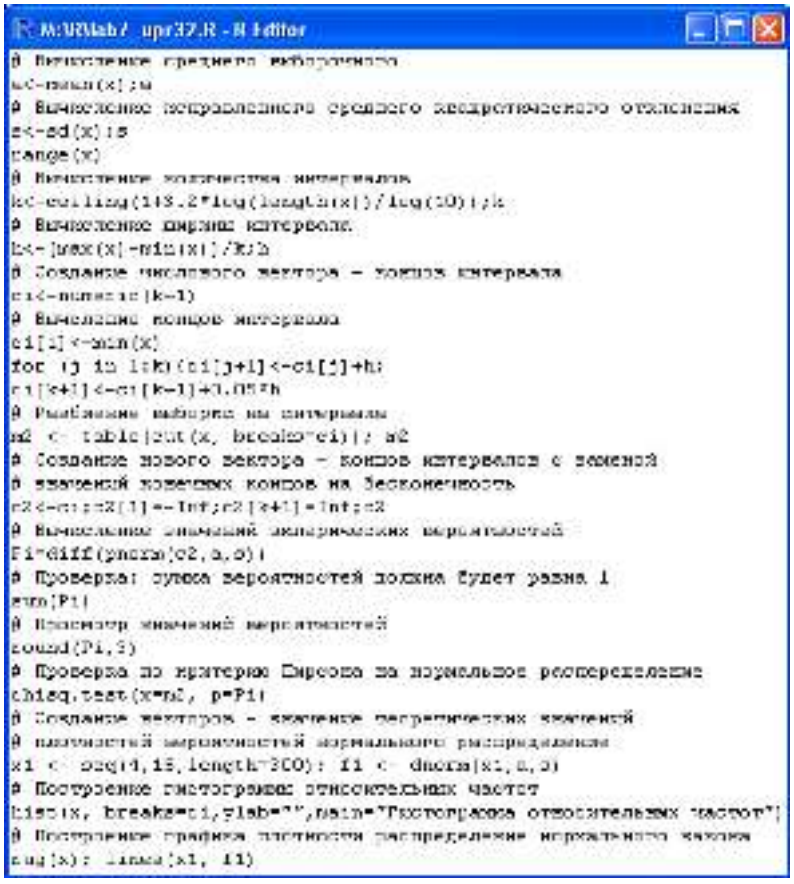
Результаты команды `chisq.test()`:

```
Chi-squared test for given probabilities
data:  m2
X-squared = 2.5495, df = 6, p-value = 0.8629
```

Результат построения диаграммы приведен на рисунке 7.3.

Значение уровня значимости (`p-value = 0.8629`) больше уровня значимости α (0,8629>0,05). Следовательно, оснований отвергать нулевую гипотезу о нормальном распределении выборки нет.

Еще один пример использования этого критерия. Проверим независимость двух случайных величин: одну возьмем распределенную по закону Стьюдента, а вторую по показательному закону.



```
# Вычисление среднего выборочного
mx<-mean(x);m
# Вычисление исправленного среднего квадратического отклонения
s<-sd(x);s
range(x)
# Вычисление количества интервалов
k<-ceiling(1+3.2*log(length(x))/log(10));k
# Вычисление ширины интервала
h<-(max(x)-min(x))/k;h
# Создание числового вектора - концов интервала
ci<-numeric(k+1)
# Вычисление концов интервала
ci[1]<-min(x)
for (j in 1:k)ci[j+1]<-ci[1]+h;
ci[k+1]<-ci[k+1]+0.05*h
# Разбиение выборки на интервалы
n2 <- table(cut(x, breaks=ci)); n2
# Создание нового вектора - концов интервалов с заменой
# крайних левого и правого концов на бесконечность
ci2<-ci;ci2[1]<-(-Inf);ci2[k+1]<-Inf;ci2
# Вычисление значений эмпирических вероятностей
Pi=diff(pnorm(ci2,m,s))
# Проверка: сумма вероятностей должна будет равна 1
sum(Pi)
# Просмотр значений вероятностей
round(Pi,3)
# Проверка по критерию Пирсона на нормальное распределение
chisq.test(x=n2, p=Pi)
# Создание векторов - значения теоретических значений
# плотностей вероятностей нормального распределения
x1 <- seq(4,15,length=200); f1 <- dnorm(x1,m,s)
# Построение гистограммы относительных частот
hist(x, breaks=ci,ylab="",main="Гистограмма относительных частот")
# Построение графика плотности распределения нормального закона
lines(x1, f1)
```

Рисунок 7.2 – Код скрипта из упражнения 4

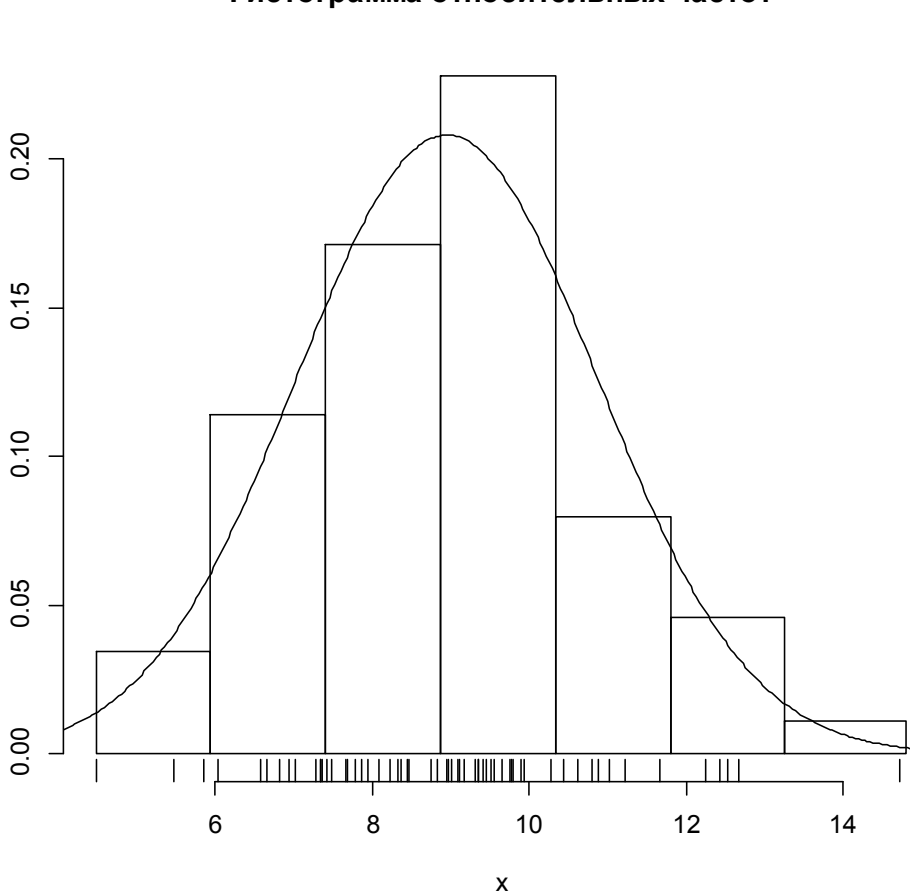



Рисунок 7.3 – Диаграмма относительных частот (упражнение 4)

### *Упражнение 5*

1) Командой `x<-rt(60,4)` создайте выборку, распределенную по закону Стьюдента с четырьмя степенями свободы.

2) Командой `y<-rexp(60,rate=1)` создайте выборку, распределенную по показательному закону с параметром $\lambda$=1.

3) Командой `chisq.test(x,y)` проверьте гипотезу о их независимости.

Результаты работы последней команды приведены ниже:

```
Pearson's Chi-squared test
data:  x and y
X-squared = 3540, df = 3481, p-value = 0.2385
```

Значение уровня значимости (`p-value = 0. 2385`) больше уровня значимости α (0,2385>0,05). Следовательно, нет оснований отвергать нулевую гипотезу о независимости двух случайных величин `x` и `y`.

Разберем t-тест Стьюдента для сравнения значения среднего выборочного двух выборок. Разберем на двух примерах. В первом примере значения среднего выборочного зададим различными, а во втором близкими.

### *Упражнение 6*

1) Командой `x<-rnorm(100,mean=5,sd=3)` создайте выборку `x` со значением математического ожидания, равного 5.

2) Командой `y<-rnorm(100,mean=7,sd=3)` создайте выборку `y` выборку `x` со значением математического ожидания, равного 7.

3) Командой `t.test(x,y)` проверьте нулевую гипотезу. Результаты работы функции приведены ниже.

```
    Welch Two Sample t-test
    data:  x and y
    t = -4.0446, df = 197.48, p-value = 7.512e-05
    alternative hypothesis: true difference in means is
not equal to 0
    95 percent confidence interval:
     -2.788346 -0.960508
    sample estimates:
    mean of x mean of y
     4.777822  6.652249
```

Значение уровня значимости (`p-value = 7.512e-05`) меньше уровня значимости α ($7{,}512\cdot10^{-5}$<0,05). Следовательно, нулевую гипотезу о равенстве среднего выборочного значения выборок, распределенных по нормальному закону, отвергаем.

### *Упражнение 7*

1) Командой `x<-rnorm(100,mean=5.6,sd=3)` создайте выборку `x` со значением математического ожидания, равного 5,6.

2) Командой `y<-rnorm(100,mean=5.7,sd=3)` создайте выборку `y` выборку `x` со значением математического ожидания, равного 5,7.

3) Командой `t.test(x,y)` проверьте нулевую гипотезу. Результаты работы функции приведены ниже.

```
    Welch Two Sample t-test
    data:  x and y
    t = -0.98844, df = 194.74, p-value = 0.3242
    alternative hypothesis: true difference in means is
not equal to 0
    95 percent confidence interval:
     -1.3030479  0.4329846
    sample estimates:
    mean of x mean of y
     5.530797  5.965828
```

Значение уровня значимости (`p-value=0.3242`) больше уровня значимости α (0,3242>0,05). Следовательно, нулевую гипотезу о равенстве среднего выборочного выборок, распределенных по нормальному закону, отвергать не будем.

Разберем F-тест Фишера для сравнения значений выборочных средних квадратических отклонений (стандартов) двух выборок, распределенных по нормальному закону. Разберем на двух примерах. В первом примере значения стандартов двух выборок, распределенных по нормальному закону, зададим различными, а во втором близкими.

### *Упражнение 8*

1) Командой `x<-rnorm(100,mean=5,sd=3)` создайте выборку `x` со стандарта, равного 3.

2) Командой `y<-rnorm(100,mean=5,sd=5)` создайте выборку `y` со стандарта, равного 3.

3) Командой `var.test(x,y)` проверьте нулевую гипотезу. Результаты работы функции приведены ниже.

```
    F test to compare two variances
    data:  x and y
    F=0.33262,  num  df=99,  denom  df=99,  p-value  =
9.633e-08
    alternative hypothesis: true ratio of variances is
not equal to 1
    95 percent confidence interval:
     0.2237997 0.4943491
```

```
sample estimates:
ratio of variances
         0.3326186
```

Значение уровня значимости (`p-value=9.633e-08`) меньше уровня значимости α ($9{,}633 \cdot 10^{-8} < 0{,}05$). Следовательно, нулевую гипотезу о равенстве стандартов выборок, распределенных по нормальному закону, отвергаем.

### *Упражнение 9*

1) Командой `x<-rnorm(100,mean=5,sd=3)` создайте выборку `x` со стандарта, равного 3.

2) Командой `y<-rnorm(100,mean=5,sd=5)` создайте выборку `y` со стандарта, равного 3.

3) Командой `var.test(x,y)` проверьте нулевую гипотезу. Результаты работы функции приведены ниже.

```
F test to compare two variances
data:  x and y
F = 0.87028, num df=99, denom df = 99, p-value =
0.4907
alternative hypothesis: true ratio of variances is
not equal to 1
95 percent confidence interval:
 0.5855638 1.2934465
sample estimates:
ratio of variances
         0.8702847
```

Значение уровня значимости (`p-value=0.4907`) больше уровня значимости α (0,4907>0,05). Следовательно, нулевую гипотезу о равенстве стандартов выборок, распределенных по нормальному закону, отвергать нет оснований.

Разберем тест Колмогорова-Смирнова для сравнения значений выборочных средних квадратических отклонений (стандартов) двух выборок, распределенных по нормальному закону.

### *Упражнение 10*

1) Командой `x<-rnorm(100,mean=5,sd=3)` создайте выборку `x`.

2) Командой `y<-rnorm(100,mean=5,sd=3.2)` создайте выборку `y`.

3) Командой `ks.test(x,y)` проверьте нулевую гипотезу. Результаты работы функции приведены ниже.

```
Two-sample Kolmogorov-Smirnov test
data:  x and y
D = 0.11, p-value = 0.5806
```

```
alternative hypothesis: two-sided
```

Значение уровня значимости (`p-value=0.5806`) больше уровня значимости α (0,5806>0,05). Следовательно, нулевую гипотезу о том, подчиняются ли два эмпирических распределения одному и тому же закону, отвергать нет оснований.

### *Упражнение 11*

1) Командой `x<-rnorm(100,mean=2,sd=3)` создайте выборку `x`.
2) Командой `y<-rnorm(100,mean=5,sd=4)` создайте выборку `y`.
3) Командой `ks.test(x,y)` проверьте нулевую гипотезу. Результаты работы функции приведены ниже.

```
Two-sample Kolmogorov-Smirnov test
data: x and y
D = 0.43, p-value = 1.866e-08
alternative hypothesis: two-sided
```

Значение уровня значимости (`p-value=1.866e-08`) меньше уровня значимости α ($1{,}866 \cdot 10^{-8}$<0,05). Следовательно, нулевую гипотезу о том, подчиняются ли два эмпирических распределения одному и тому же закону, отвергаем.

## 7.3 Самостоятельная работа

1) Проверьте при помощи теста Шапиро-Уилка выборку своего варианта, что она распределена по нормальному закону.
2) Проверьте при помощи критерия согласия $\chi^2$ Пирсона выборку своего варианта, что она распределена по нормальному закону.
3) Проверьте при помощи критерия Колмогорова-Смирнова выборку своего варианта, что она распределена по нормальному закону.
4) Создайте две выборки, распределенные по нормальному закону, имеющие различные средние выборочные и одно и то же значение стандарта.
5) Проверьте их при помощи t-теста Стьюдента.

## 7.4 Контрольные вопросы

1) Дайте определение статистической гипотезы.
2) Дайте определение нулевой гипотезы.
3) Дайте определение конкурирующей (альтернативной) гипотезы.
4) Дайте определение статистического критерия.
5) Дайте определение критической области.
6) Дайте определение области принятия гипотезы.
7) Дайте определение критической точки.
8) Дайте определение уровня значимости критерия.
9) Дайте определение мощности критерия.

10) Дайте определение надежности критерия.
11) Дайте определение критерия согласия.
12) Что проверяет тест Шапиро-Уилка?
13) Что проверяет критерий согласия $\chi^2$ Пирсона?
14) Что проверяет критерий Колмогорова-Смирнова?
15) Что проверяет t-теста Стьюдента?
16) Что проверяет F-тест Фишера?

## Литература

1. Шипунов, А. Введение в R / А. Шипунов, Е. Балдин // Linux Format. – 2008. – № 1. – С. 98–101.

2. Сайт «Советы по использованию R» – [Ресурс доступа]: http://pj.freefaculty.org/R/Rtips.html

3. Статистический анализ данных в системе R: учебное пособие / А.Г. Буховец, П.В. Москалев, В.П. Богатова, Т.Я. Бирючинская; под ред. проф. А.Г. Буховца. – Воронеж: ВГАУ, 2010. – 124 с.

4. Наглядная статистика. Используем R!: учебное пособие / А. Шипунов, Е. Балдин, П. Волкова, А. Коробейников, С. Назарова, С. Петров, В. Суфиянов. – Москва: ДМК Пресс, 2014. – 298 с.

5. Зарядов, И.С. Статистический пакет R: теория вероятностей и математическая статистика: учебно-методическое пособие / И.С. Зарядов. – Москва: Изд-во Российского университета дружбы народов, 2010. – 141 с.

6. ГОСТ Р 50779.90-2014 Статистические методы. Требования к информации о программном обеспечении, используемом при разработке нормативных документов.

7. ГОСТ Р 50779.10-2000. Статистические методы. Вероятность и основы статистики. Термины и определения.

8. ГОСТ Р 50779.21-2004. Статистические методы. Правила определения и методы расчета статистических характеристик по выборочным данным. Часть 1. Нормальное распределение.

9. ГОСТ Р 50.1.040-2002. Статистические методы. Планирование экспериментов. Термины и определения.

10. ГОСТ Р 50779.11-2000. Статистические методы. Статистическое управление качеством. Термины и определения.

11. Sturgess, H.A. The choice of classic intervals // J. Am. Statist. Assoc. – 1926. – № 3. – 47 p.

12. Шторм, Р. Теория вероятностей. Математическая статистика. Статистический контроль качества / Р. Шторм. – Москва: Мир, 1970. – 368 с.

13. Heinhold I., Gaede K.W. Ingeniur statistic. – München; Wien, Springler Verlag, 1964. – 352 s.

14. Тонева, Е. Аппроксимация распределений погрешности средств измерений / Е. Тонева // Измерительная техника. – 1981. – № 6. – С. 15–16.

15. Гмурман, В.Е. Теория вероятностей и математическая статистика / В.Е. Гмурман. – Москва: Высш. шк., 2001. – 479 с.

16. Середина, М.Н. Математический анализ и статистика. Сборник задач: учебное пособие / М.Н. Середина, Л.Ю. Шипик. – Зерноград: Азово-Черноморский инженерный институт ФГБОУ ВО Донской ГАУ, 2015. – 59 с.

17. Удинцова, Н.М. Математическая статистика: учебное пособие / Н.М. Удинцова, Л.Н. Шаповалова. – Зерноград: ФГОУ ВПО АЧГАА, 2011. – 100 с.

18. Сидняев, Н.И. Теория планирования эксперимента и анализ статистических данных / Н.И. Сидняев. – Москва: Издательство Юрайт, 2012. – 399 с.

# Приложение

## Приложение 1. Варианты задания к лабораторной работе № 1

***Вариант 1***

182 182 182 183 183 183 183 183 183 183 184 183 183 183 183 183 183 183 182 182 184 184 184 184 184 184 184 184 184 184 184 184 184 179 179 179 178 178 178 182 182 182 181 181 181 180 180 180 180 177 177 176 176 184 184 184 184 184 184

***Вариант 2***

20,63 19,93 21,08 20,43 18,64 20,73 20,55 20,01 20,43 20,07 20,49 19,83 17,74 19,16 18,54 19,81 18,72 20,30 19,18 21,19 19,15 19,32 19,37 21,60 20,85 19,40 17,83 20,36 19,63 22,88 19,02 19,95 20,68 19,63 21,12 18,47 21,43 18,75 20,20 20,10 20,64 21,32 19,77 19,46 19,61 19,23 20,77 20,86 21,13 19,38 20,07 18,81 19,93 19,22 19,73 19,07 19,96 19,13 21,20 21,03

***Вариант 3***

1,856 2,354 0,9544 1,964 1,761 2,364 1,469 2,103 2,635 2,011 2,487 1,334 1,355 0,822 2,442 2,794 1,495 1,759 1,877 2,168 1,224 2,433 1,793 2,342 1,249 1,942 2,818 2,174 1,45 1,519 2,167 1,797 1,776 2,886 2,807 2,064 2,333 1,924 2,353 2,241 1,775 1,596 1,118 2,118 2,316 2,203 1,558 2,579 1,564 1,493 1,299 1,183 2,117 2,862 2,323 1,748 1,624 1,597 2,351 1,423

***Вариант 4***

174 180 183 187 180 168 173 177 182 185 179 171 194 181 181 176 180 179 179 181 179 180 183 183 179 175 178 179 183 181 181 185 179 179 184 177 172 174 171 181 180 178 178 177 171 186 174 184 182 181 193 176 174 180 180 175 182 179 180 188

***Вариант 5***

3,712 4,709 1,909 3,930 3,521 4,727 2,939 4,205 5,271 4,022 4,975 2,669 2,711 1,644 4,884 5,588 2,990 3,517 3,754 4,336 2,448 4,866 3,586 4,683 2,499 3,883 5,637 4,348 2,901 3,039 4,335 3,594 3,552 5,771 5,613 4,127 4,666 3,847 4,706 4,482 3,550 3,191 2,235 4,235 4,631 4,406 3,116 5,157 3,128 2,986 2,598 2,366 4,234 5,724 4,647 3,497 3,249 3,193 4,701 2,846

***Вариант 6***

78 82 71 79 77 82 75 80 84 79 83 74 74 70 83 85 75 77 78 80 73 82 77 82 73 79 86 80 75 75 80 77 77 86 85 80 82 78 82 81 77 76 72 80 82 81 75 84 76 75 73 72 80 86 82 77 76 76 82 74

***Вариант 7***

49 54 57 60 54 45 49 52 56 59 53 48 66 55 55 51 54 54 53 55 53 55 57 57 53 50 53 54 57 55 55 58 54 53 58 52 48 49 48 55 54 52 53 52 47 60 50 58 56 55 65 51 49 54 54 51 56 54 54 61

***Вариант 8***

19,83 21,41 20,55 22,30 23,40 25,86 21,00 19,85 26,53 22,48 21,74 24,30 19,37 19,23 23,12 20,45 22,31 22,74 25,74 22,28 25,98 22,24 24,04 21,26 23,03 24,92 23,78 24,20 21,99 22,12 21,44 20,93 25,28 19,74 25,39 17,70 19,67 19,94 19,15 24,43 20,29 21,61 24,30 23,02 23,20 20,00 22,18 24,18 26,01 24,36 20,61 24,04 20,52 21,45 23,65 20,81 21,98 18,71 25,39 22,79

***Вариант 9***

19,02 18,85 18,96 18,79 18,99 19,05 20,22 18,58 18,93 18,92 19,00 18,17 19,38 19,16 19,14 19,30 19,58 18,08 18,70 18,80 19,23 19,54 18,86 20,25 17,72 20,00 18,96 19,45 19,76 19,04 18,35 19,81 19,86 18,57 19,02 19,59 19,35 19,90 18,81 19,29 18,76 18,65 19,32 19,49 19,05 18,77 18,62 19,04 19,37 19,48 18,81 18,92 19,82 18,72 18,10 18,28 18,90 18,87 19,24 19,03

***Вариант 10***

13,068 14,563 10,363 13,395 12,782 14,591 11,907 13,808 15,406 13,534 14,962 11,503 11,566 9,967 14,825 15,882 11,985 12,776 13,130 14,004 11,172 14,799 12,879 14,525 11,248 13,325 15,955 14,021 11,851 12,059 14,002 12,892 12,828 16,157 15,920 13,691 14,499 13,271 14,559 14,222 12,825 12,287 10,853 13,853 14,447 14,109 12,175 15,236 12,192 11,979 11,396 11,049 13,851 16,086 14,470 12,744 12,372 12,289 14,552 11,769

***Вариант 11***

11,71 13,68 14,68 15,88 13,53 10,11 11,63 12,83 14,15 15,20 13,27 11,07 17,91 13,88 14,05 12,36 13,62 13,36 13,25 13,91 13,27 13,69 14,68 14,45 13,24 12,19 13,06 13,32 14,66 14,04 13,90 15,13 13,41 13,24 14,88 12,68 11,14 11,71 11,08 13,97 13,56 12,93 13,05 12,70 10,92 15,57 11,91 14,88 14,11 13,86 17,45 12,36 11,71 13,57 13,59 12,25 14,36 13,49 13,62 16,10

***Вариант 12***

8,641 10,142 9,326 10,988 12,031 14,370 9,749 8,660 15,007 11,153 10,453 12,885 8,202 8,066 11,767 9,224 10,998 11,402 14,248 10,967 14,483 10,932 12,638 9,997 11,674 13,470 12,393 12,789 10,694 10,817 10,170 9,680 13,819 8,556 13,925 6,616 8,485 8,740 7,996 13,005 9,080 10,329 12,884 11,672 11,836 8,803 10,869 12,768 14,511 12,945 9,379 12,642 9,291 10,178 12,268 9,573 10,678 7,578 13,924 11,447

***Вариант 13***

29,418 32,332 30,635 33,859 31,170 31,124 36,087 33,437 31,282 31,321 35,575 33,806 32,873 34,998 31,259 30,846 30,372 36,082 33,091 32,767 32,285 33,641 30,795 34,164 31,367 29,054 31,040 32,121 29,498 31,621 34,336 32,501 32,507 29,298 31,277 33,949 30,852 34,545 32,151 35,212 30,971 29,090 35,355 31,449 30,954 32,574 30,439 30,455 28,558 31,544 31,807 32,951 35,616 31,206 34,502 30,611 34,454 29,384 31,754 34,287

***Вариант 14***

34,435 29,663 36,632 31,284 34,961 31,373 35,298 33,554 31,719 32,116 30,152 31,124 30,112 30,975 33,936 32,139 35,673 32,925 32,745 31,004 28,731 33,678 28,376 30,901 35,798 28,863 31,697 34,340 31,608 34,199 33,343 36,685 30,948 30,256 31,474 31,916 29,829 30,450 33,428 32,889 33,231 32,633 31,555 32,977 30,080 32,381 33,430 34,826 33,083 29,302 31,514 29,888 30,268 29,984 32,884 33,625 31,648 33,949 30,726 30,266

***Вариант 15***

9,114 9,413 8,573 9,179 9,056 9,418 8,882 9,262 9,581 9,207 9,492 8,801 8,813 8,493 9,465 9,676 8,897 9,055 9,126 9,301 8,734 9,460 9,076 9,405 8,750 9,165 9,691 9,304 8,870 8,912 9,300 9,078 9,066 9,731 9,684 9,238 9,400 9,154 9,412 9,344 9,065 8,957 8,671 9,271 9,389 9,322 8,935 9,547 8,939 8,896 8,779 8,710 9,270 9,717 9,394 9,049 8,975 8,958 9,410 8,854

***Вариант 16***

8,6035 9,2597 9,5921 9,9923 9,2084 8,0716 8,5761 8,9781 9,4171 9,7665 9,1222 8,3893 10,6714 9,3272 9,3842 8,8204 9,2402 9,1547 9,1164 9,3381 9,1220 9,2631 9,5942 9,5165 9,1120 8,7623 9,0540 9,1395 9,5871 9,3794 9,3346 9,7433 9,1701 9,1118 9,6613 8,9254 8,4137 8,6022 8,3922 9,3552 9,2208 9,0102 9,0488 8,9335 8,3385 9,8910 8,6703 9,6613 9,4032 9,3215 10,5152 8,8184 8,6032 9,2246 9,2293 8,7818 9,4850 9,1975 9,2410 10,0662

***Вариант 17***

5,6703 6,4211 6,0128 6,8439 7,3655 8,5350 6,2247 5,6800 8,8533 6,9266 6,5765 7,7927 5,4512 5,3828 7,2333 5,9618 6,8490 7,0511 8,4741 6,8334 8,5917 6,8162 7,6689 6,3486 7,1869 8,0849 7,5467 7,7447 6,6971 6,7583 6,4352 6,1902 8,2593 5,6282 8,3124 4,6582 5,5927 5,7202 5,3478 7,8526 5,8900 6,5146 7,7921 7,1860 7,2680 5,7513 6,7845 7,7340 8,6057 7,8224 6,0394 7,6709 5,9954 6,4389 7,4841 6,1367 6,6888 5,1391 8,3120 7,0735

***Вариант 18***

3,4088 4,8662 4,0175 5,6297 4,2849 4,2618 6,7436 5,4186 4,3409 4,3603 6,4876
5,6030 5,1363 6,1992 4,3294 4,1228 3,8858 6,7408 5,2456 5,0835 4,8427 5,5206
4,0975 5,7819 4,3833 3,2271 4,2200 4,7604 3,4488 4,5104 5,8678 4,9506 4,9533
3,3490 4,3383 5,6747 4,1260 5,9726 4,7754 6,3061 4,1854 3,2448 6,3776 4,4243
4,1770 4,9869 3,9195 3,9277 2,9792 4,4718 4,6034 5,1754 6,5080 4,3029 5,9509
4,0054 5,9268 3,3921 4,5769 5,8435

***Вариант 19***

21,183 18,229 22,543 19,233 21,509 19,288 21,718 20,638 19,502 19,748 18,532
19,134 18,508 19,042 20,875 19,762 21,950 20,249 20,137 19,060 17,653 20,715
17,433 18,996 22,027 17,734 19,489 21,125 19,434 21,037 20,508 22,576 19,025
18,596 19,350 19,624 18,332 18,717 20,560 20,227 20,438 20,068 19,401 20,281
18,487 19,912 20,562 21,425 20,346 18,006 19,375 18,368 18,604 18,428 20,224
20,682 19,458 20,883 18,888 18,603

***Вариант 20***

15,099 13,395 14,275 13,549 16,853 18,225 15,642 12,616 17,665 18,182 14,788
15,241 13,332 15,732 14,297 14,623 18,498 16,692 19,352 15,721 10,205 14,077
15,078 15,242 13,276 16,782 10,987 16,504 15,424 13,168 15,276 14,587 14,081
18,411 16,920 18,068 13,678 13,729 13,910 13,034 18,078 11,677 15,248 18,036
15,099 14,599 12,538 16,449 13,599 17,680 10,691 15,019 16,230 15,523 15,175
14,438 14,290 16,455 15,547 14,032

***Вариант 21***

15,159 14,751 14,995 14,594 15,077 15,213 18,022 14,085 14,941 14,900 15,109
13,109 16,013 15,477 15,427 15,813 16,493 12,902 14,382 14,631 15,657 16,397
14,754 18,106 12,034 17,508 15,012 16,182 16,921 15,199 13,529 17,042 17,163
14,072 15,139 16,505 15,942 17,260 14,636 15,788 14,534 14,263 15,879 16,276
15,216 14,552 14,183 15,185 15,988 16,247 14,633 14,899 17,062 14,438 12,943
13,373 14,869 14,783 15,668 15,164

***Вариант 22***

56,944 59,481 56,309 60,859 55,250 56,660 52,486 58,150 58,377 55,870 58,196
59,174 57,359 56,086 56,952 57,557 56,572 55,705 58,482 59,943 57,668 57,742
58,179 58,207 58,749 57,628 55,758 58,951 54,712 58,476 57,877 57,461 57,646
53,272 56,159 59,829 56,287 56,578 60,514 54,341 55,918 58,534 56,526 56,757
58,337 58,224 57,842 59,508 56,628 59,095 56,009 59,039 56,295 54,760 58,309
56,525 56,798 55,965 54,047 56,742

***Вариант 23***

37,209 36,211 42,923 42,716 37,318 45,650 37,841 39,263 46,828 44,649 43,764
43,895 37,628 39,914 45,208 39,301 43,742 39,592 49,888 46,491 43,736 41,232
41,480 40,009 40,092 38,785 46,912 43,639 44,663 50,309 46,790 38,556 42,906
39,273 44,456 42,891 46,380 38,343 39,134 38,589 47,383 43,237 38,887 39,846
33,449 43,661 44,848 47,837 43,684 44,230 43,355 41,620 49,288 49,872 31,290
43,743 44,383 43,378 45,460 43,782

***Вариант 24***

16,126 29,396 46,745 31,064 31,425 45,988 35,548 32,395 34,856 30,793 32,105
23,174 38,981 31,465 26,317 38,859 28,402 33,131 31,450 26,495 36,863 40,366
34,261 36,578 33,779 31,877 27,978 22,661 39,640 35,129 29,432 41,631 34,988
35,898 29,082 37,549 30,990 24,441 37,882 16,354 34,921 34,964 31,249 31,546
30,942 37,759 31,578 41,061 21,637 30,736 28,156 22,253 24,314 42,753 25,940
40,053 21,734 35,273 42,913 39,277

***Вариант 25***

28,025 26,475 34,193 25,987 29,866 31,681 23,712 23,708 23,837 34,414 27,626
32,583 21,548 28,383 33,014 26,834 22,472 31,963 27,744 33,626 25,371 34,363
34,955 31,634 34,875 34,142 18,781 28,560 23,306 27,579 24,638 37,881 22,428
19,854 22,810 31,051 28,003 27,451 24,052 27,050 25,361 26,675 26,485 21,189
29,123 26,522 33,965 25,676 28,090 30,991 29,523 28,913 31,931 27,744 21,960
31,100 27,307 36,081 23,342 26,118

## Приложение 2
## Варианты задания к лабораторной работе № 3

1) $y=x^2-4x+3$;

| $x$ | -1 | -0,5 | 0 | 0,5 | 1 |
|---|---|---|---|---|---|
| $y$ | 9 | 5,5 | 3 | 1,5 | 1 |

2) $y=2x^2-4x+1$;

| $x$ | -1 | 0 | 1 | 2 | 3 |
|---|---|---|---|---|---|
| $y$ | 7 | 1 | -1 | 1 | 7 |

3) $y=x^2-5x-6$;

| $x$ | 1 | 3 | 5 | 7 | 9 |
|---|---|---|---|---|---|
| $y$ | -10 | -12 | -6 | 8 | 30 |

4) $y=x^2-3x-3$;

| $x$ | -2 | -0,5 | 1 | 3,5 | 7 |
|---|---|---|---|---|---|
| $y$ | 7 | -1,25 | -5 | -1,25 | 25 |

5) $y=x^2-3x-1$;

| $x$ | 0 | 0,5 | 2 | 4,5 | 8 |
|---|---|---|---|---|---|
| $y$ | -1 | -2,25 | -3 | 5,75 | 39 |

6) $y=x^2-2x+2$;

| $x$ | -1 | 0 | 2 | 5 | 9 |
|---|---|---|---|---|---|
| $y$ | 5 | 2 | 2 | 17 | 65 |

7) $y=x^2-6x+2$;

| $x$ | -0,5 | 0 | 0,5 | 1 | 1,5 |
|---|---|---|---|---|---|
| $y$ | 5,25 | 2 | -0,75 | -3 | -4,75 |

8) $y=3x^2-3x-1$;

| $x$ | -2 | -1 | 0 | 1 | 2 |
|---|---|---|---|---|---|
| $y$ | 19 | 7 | 1 | 1 | 7 |

9) $y=2x^2-3x-1$;

| $x$ | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| $y$ | -1 | -2 | 1 | 8 | 19 |

10) $y=x^2-3x+2$;

| $x$ | -0,5 | 0 | 0,5 | 1 | 1,5 |
|---|---|---|---|---|---|
| $y$ | 3,75 | 2 | 0,75 | 0 | -0,25 |

11) $y=x^2-2x+1$;

| $x$ | -0,5 | 0 | 0,5 | 1 | 1,5 |
|---|---|---|---|---|---|
| $y$ | 2,25 | 1 | 0,25 | 0 | 0,25 |

12) $y=3x^2-2x+2$;

| $x$ | -1,5 | -1 | -0,5 | 0 | 0,5 |
|---|---|---|---|---|---|
| $y$ | 11,75 | 7 | 3,75 | 2 | 1,75 |

13) $y=x^2-3x-4$;

| $x$ | -1,5 | 0 | 2,5 | 6 | 10,5 |
|---|---|---|---|---|---|
| $y$ | 2,75 | -4 | -5,25 | 14 | 70,25 |

14) $y=x^2-7x-8$;

| $x$ | -0,5 | 1 | 2,5 | 4 | 5,5 |
|---|---|---|---|---|---|
| $y$ | -4,25 | -14 | -20 | -19 | -16,5 |

15) $y=x^2-3x+5$;

| $x$ | -1,5 | -1 | -0,5 | 0 | 0,5 |
|---|---|---|---|---|---|
| $y$ | 11,75 | 8 | 6,9 | 4 | 3,5 |

16) $y=x^2-x+1$;

| $x$ | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|
| $y$ | 3,5 | 7 | 12,5 | 21 | 31 |

17) $y=-x^2+2x+5$;

| $x$ | 0 | 0,3 | 0,6 | 0,9 | 1,2 |
|---|---|---|---|---|---|
| $y$ | 5 | 6 | 6 | 6,3 | 5,9 |

18) $y=-x^2+3x+1$;

| $x$ | -0,4 | 0 | 0,4 | 0,8 | 1,2 |
|---|---|---|---|---|---|
| $y$ | -0,36 | 1 | 2,15 | 2,8 | 3,16 |

19) $y=-x^2+3x+1$;

| $x$ | -1,5 | -1 | -0,5 | 0 | 0,5 |
|---|---|---|---|---|---|
| $y$ | -7,5 | -5 | -3 | -1 | 0,9 |

20) $y=-x^2+5x+1$;

| $x$ | 1 | 4 | 7 | 10 | 13 |
|---|---|---|---|---|---|
| $y$ | 5 | 5 | -13 | -49 | -103 |

21) $y=\ln(x+1)$;

| $x$ | 0,5 | 2 | 4,5 | 8 | 12,5 |
|---|---|---|---|---|---|
| $y$ | 0,5 | 1,1 | 1,7 | 2,4 | 2,9 |

22) $y=\ln(2x+1)$;

| $x$ | 0,5 | 2,5 | 4,5 | 6,5 | 8,5 |
|---|---|---|---|---|---|
| $y$ | 0,8 | 1,8 | 2,3 | 2,72 | 2,89 |

23) $y=e^{x+1}$;

| $x$ | 0,71 | 1,12 | 1,58 | 2,06 | 2,55 |
|---|---|---|---|---|---|
| $y$ | 5,6 | 8,3 | 13,7 | 21,4 | 35 |

24) $y=\sqrt{x+1}$;

| $x$ | -1 | 2 | 7 | 14 | 23 |
|---|---|---|---|---|---|
| $y$ | 0 | 1,8 | 2,7 | 3,9 | 5 |

25) $y=\sqrt{x^2+1}$;

| $x$ | -1 | 0 | 3 | 8 | 15 |
|---|---|---|---|---|---|
| $y$ | 1,3 | 1 | 3.1 | 8 | 15 |

**Степовой Дмитрий Владимирович**
кандидат физико-математических наук, доцент

**Жогалев Алексей Петрович**
кандидат технических наук, доцент

**Емелин Александр Аркадьевич**
кандидат технических наук, доцент

# СТАТИСТИЧЕСКИЕ МЕТОДЫ ОБРАБОТКИ ЭКСПЕРИМЕНТАЛЬНЫХ ДАННЫХ И ПЛАНИРОВАНИЕ ЭКСПЕРИМЕНТА

*Лабораторный практикум*

Практикум к изданию в авторской редакции подготовила
Гончарова Н.А.
Верстка Кудрявцева Г.С.
Дизайн обложки Вдовикина С.П.

Подписано в печать 28.09.2016 г.
Формат 60×84/16. Усл. п. л. 6,9. Тираж 30 экз. Заказ № 308.

РИО Азово-Черноморского инженерного института – филиала
ФГБОУ ВО Донской ГАУ
347740, г. Зерноград Ростовской области, ул. Советская, 15.