Казанский (Приволжский) Федеральный Университет, Институт Вычислительной Математики и Информационных Технологий

Демонстрационные программына языке R

Для студентов ИВМиИТ

Ирина Григорьева
1.1.2016

Оглавление

Центральная предельная теорема

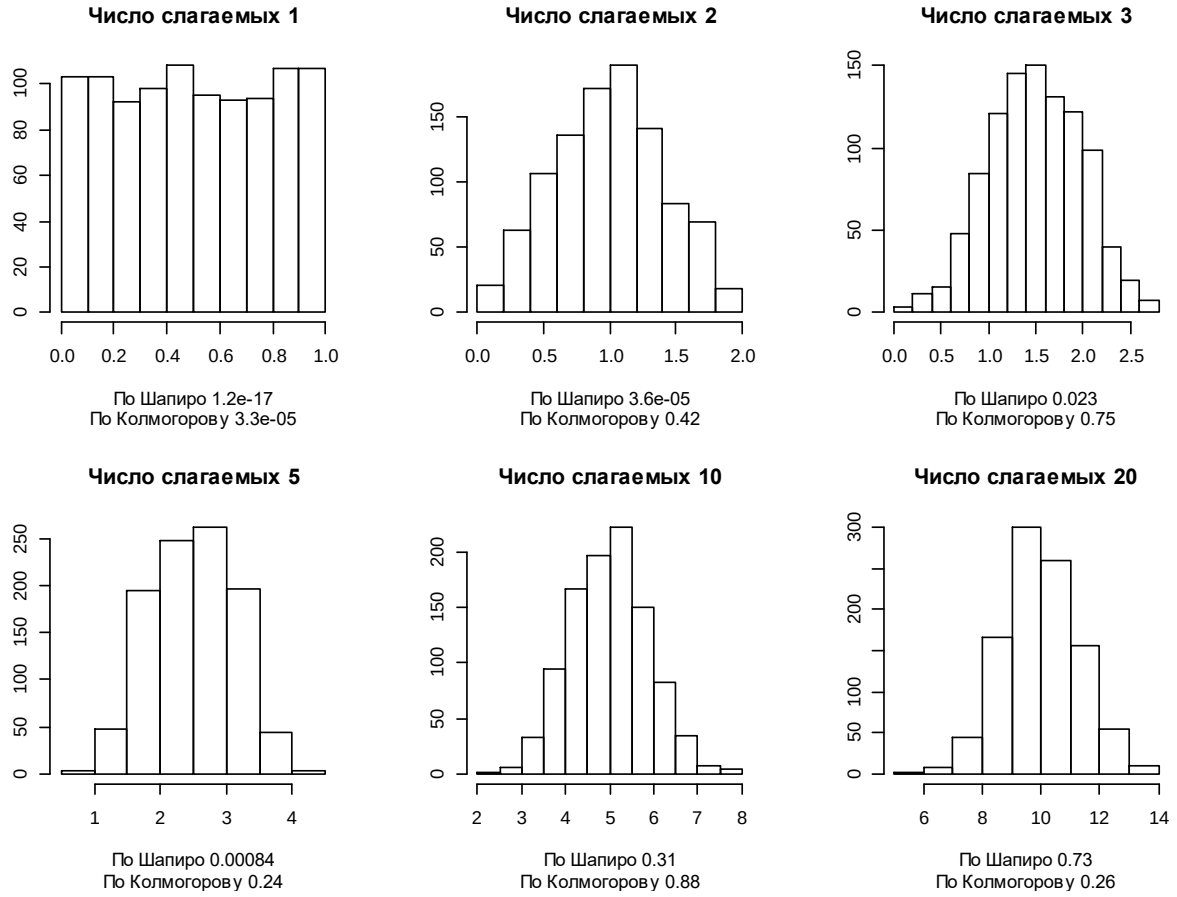
Общая постановка задачи

Центральные предельные теоремы – класс теорем в теории вероятностей, утверждающих, что сумма достаточно большого количества независимых случайных величин, имеющих примерно одинаковые масштабы (ни одно из слагаемых не доминирует, не вносит в сумму определяющего вклада), имеет распределение, близкое к нормальному.

Задача состоит в том, чтобы продемонстрировать этот факт статистически.

Данные для решения

Строятся независимые выборки из равномерного распределения, одинакового объёма. Рассматриваются суммы значений 1, 2, 3, … выборок. На экран выводятся гистограммы сумм. Кроме того, нормальность суммарной выборки проверяется на нормальность с помощью критерия Шапиро-Уилкса.



Сравнение оценок методом Монте-Карло

Общая постановка задачи

Пусть данные описываются одним или несколькими показателями. Одна из первых задач анализа – исследовать закон их распределения. В частности, найти оценки параметров. Студент должен понимать, что точное значение θ параметра – константа, но его оценка T – случайная величина. Чтобы сделать этот факт наглядным, можно смоделировать процесс создания выборок и вычисления по ним оценок. Таким образом, мы получим «выборку выборок», на основе которой строится выборка из с.в. T.

Для одного и того же параметра θ можно ввести несколько различных оценок. Они могут быть смещенными/несмещенными, более или менее эффективными. Для изучения этих свойств предлагается создать большой набор выборок из с.в., подсчитать несколько оценок и вывести на экран как значения этих оценок, так и их характеристики.

Часто характеристики величины T можно подсчитать точно. Но можно, в свою очередь, найти их выборочные оценки: $\bar{T}$ вместо ET и s_T^2 вместо $D_T = \sigma_T^2$. Именно они и вычисляются в предлагаемой программе.

Кроме того, строится линия регрессии для каждой пары оценок (точнее, их отклонений от истинного значения). Коэффициент наклона показывает, какая из оценок эффективней.

Данные для решения

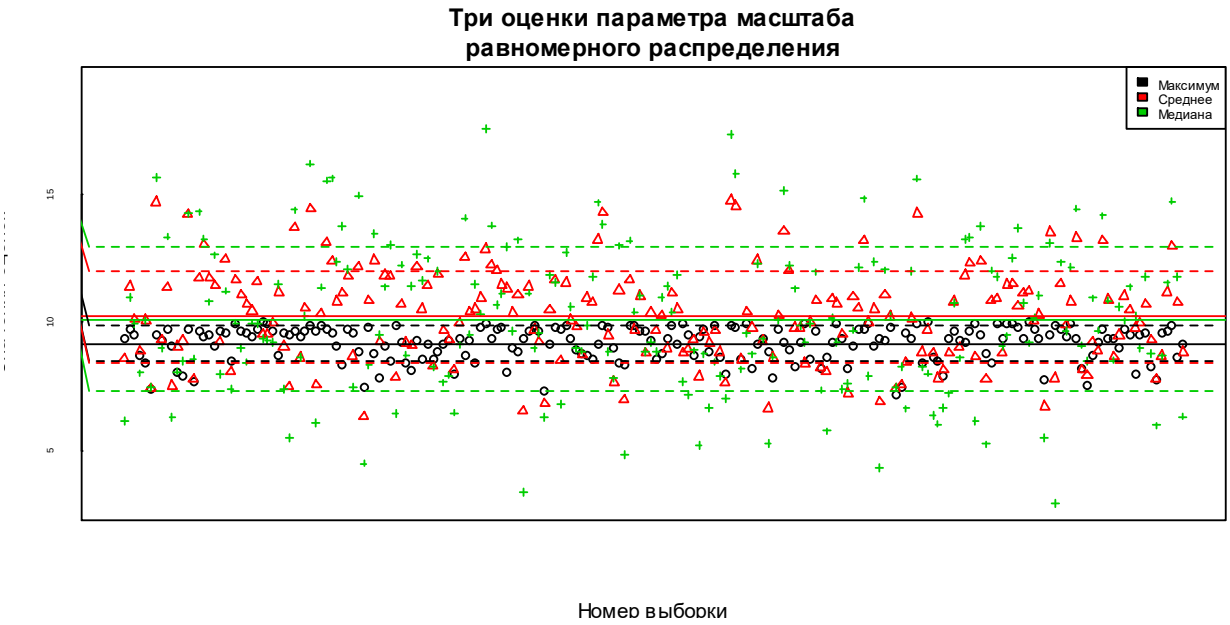
В качестве с.в. берется величина, равномерно распределенная в промежутке $[0; a]$. Здесь a – параметр масштаба, оценку которого необходимо построить. Сравниваются три оценки:

a_1 – максимум выборки;

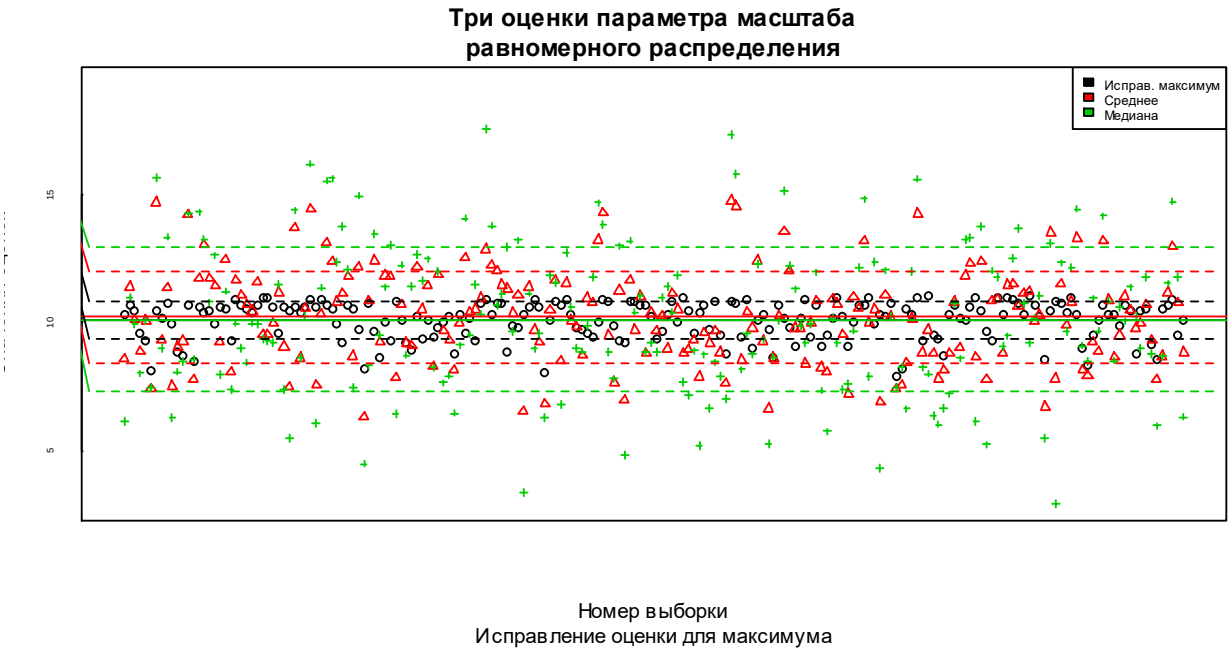
a_2 – удвоенное среднее;

a_3 – удвоенная медиана.

На рисунке видно, что самой эффективной является оценка «максимум», далее – среднее и наименее эффективная – медиана.

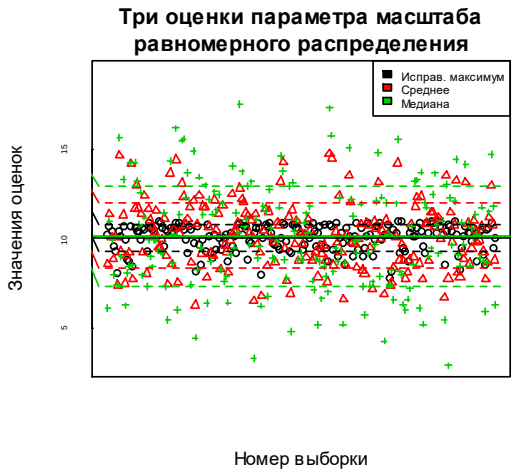


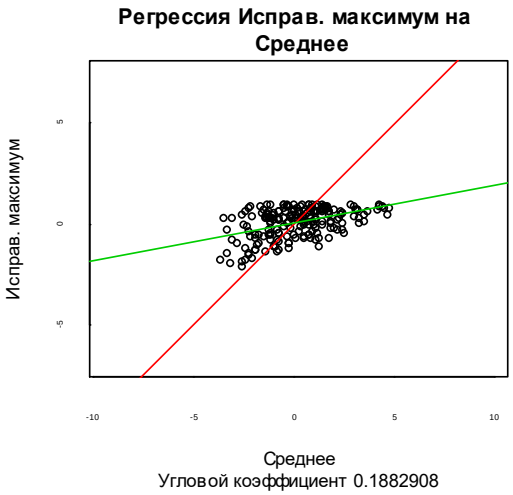
Однако максимум – смещен вниз. Это можно исправить, корректировочный коэффициент $(n + 1)/n$. Результат виден на втором графике.

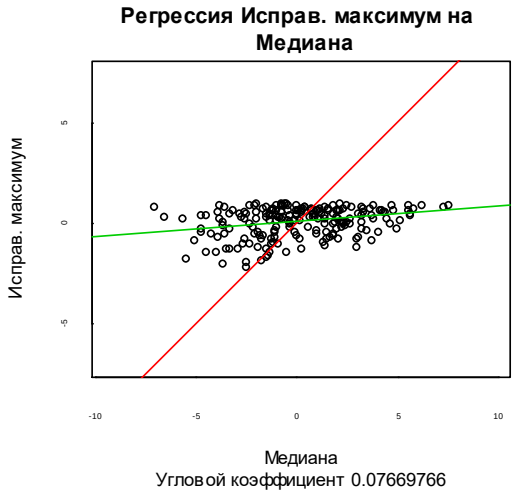


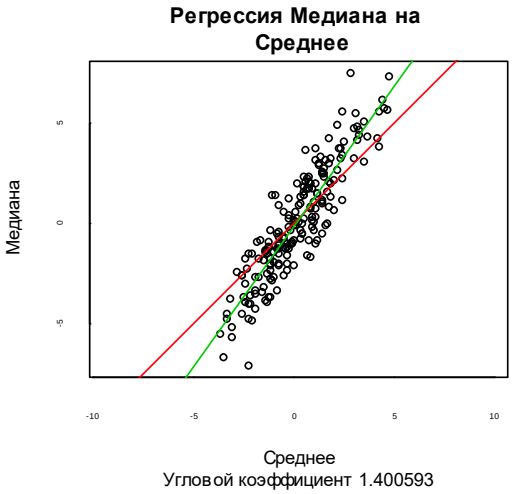
Исправление оценки для максимума

Ещё один способ сравнить оценки – построить линии регрессии. На рисунке красная прямая имеет уравнение $y = x$. Видно, что исправленный максимум гораздо меньше отклоняется от истинного значения, чем две другие оценки. С другой стороны, медиана даёт несколько худшую оценку, чем среднее.


Три оценки параметра масштаба равномерного распределения
Исправ. максимум
Среднее
Медиана
Значения оценок
Номер выборки


Регрессия Исправ. максимум на Среднее
Исправ. максимум
Среднее
Угловой коэффициент 0.1882908


Регрессия Исправ. максимум на Медиана
Исправ. максимум
Медиана
Угловой коэффициент 0.07669766


Регрессия Медиана на Среднее
Медиана
Среднее
Угловой коэффициент 1.400593

Состоятельность оценки

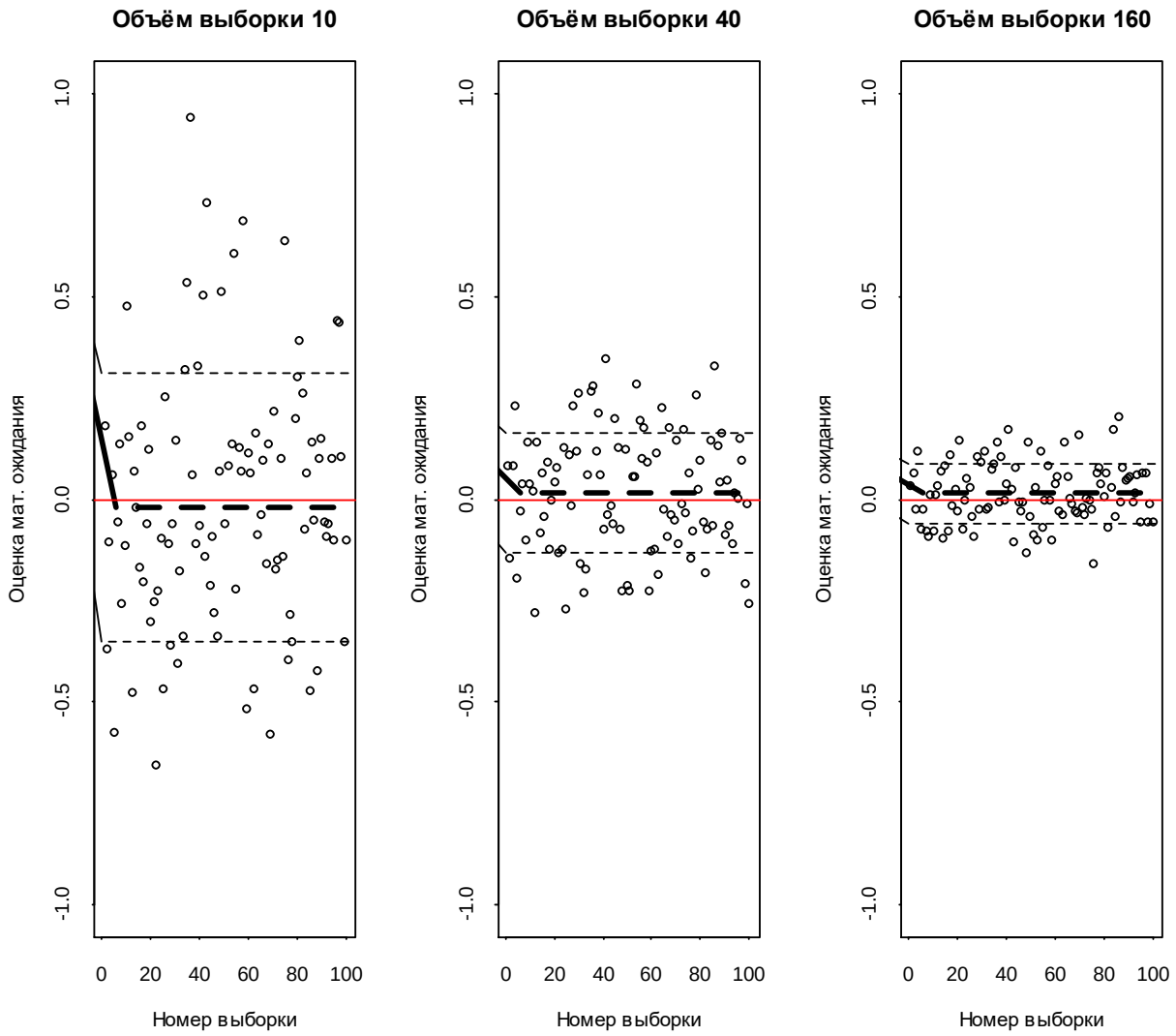
Общая постановка задачи

Предыдущая программа демонстрирует свойства несмещенности и эффективности оценок. Аналогично можно показать состоятельность оценки. Для этого можно сравнить значения оценки T, полученной по выборкам разного объёма. Создается выборка выборок, причем значение параметра вычисляется только по части выборки.

Данные для решения

В качестве с.в. берется нормально распределённая величина с фиксированными значениями параметров. Вычисляется выборочное среднее для выборок размером 10, 40 и 160 значений. На экран выводятся как значения этих оценок, так и их среднее. Красным цветом отмечено истинное значение математического ожидания.

Для наглядности также показан промежуток $\bar{T} \pm s(T)$, где $T = \bar{x}$ – оценка математического ожидания. Видно, что при увеличении объёма выборки в 4 раза разброс значений уменьшается примерно в 2 раза.



Доверительные интервалы

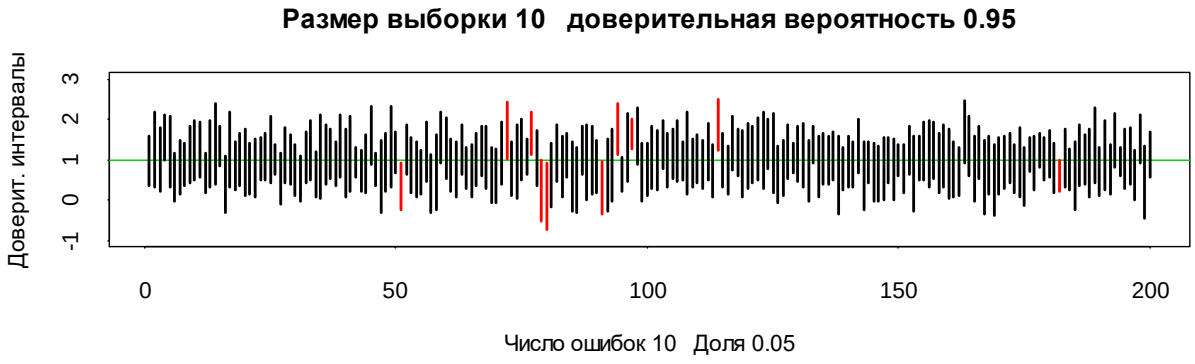
Общая постановка задачи

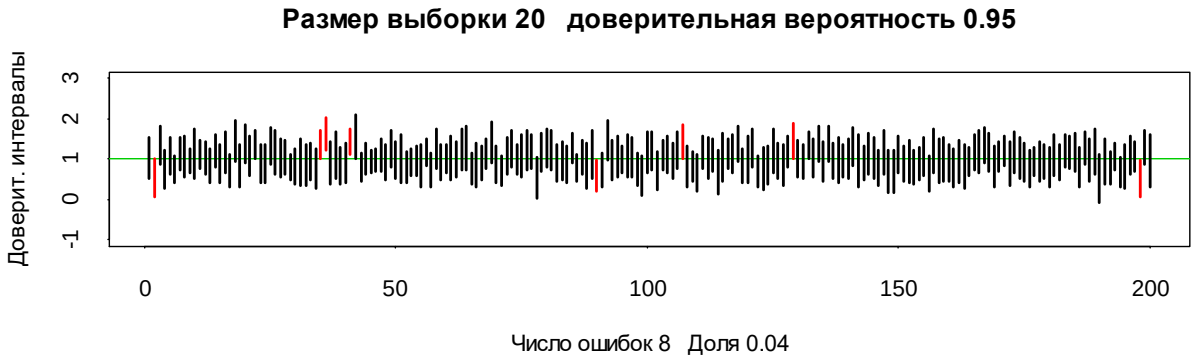
Доверительный интервал – это интервальная оценка параметра θ, то есть интервал, который накрывает точное значение с доверительной вероятностью γ. Вероятность ошибки обозначим α = 1 – γ. Иногда студенты говорят, что γ – «вероятность того, что истинное значение θ попадает в интервал». Это высказывание не совсем корректно, так как θ – константа, а вот концы доверительного интервала вычисляются по выборке и являются случайными величинами.

Чтобы показать это наглядно, создаём выборку из выборок и для каждой рассчитываем доверительный интервал.

Данные для решения

Рассматривается нормально распределенная случайная величина. В качестве параметра берём математическое ожидание μ. Строим доверительные интервалы: ошибочные отмечены на графике красным цветом. То же исследование повторяем для другого размера выборки. Видно, что для бóльшей выборки доверительный интервал уже (при той же доверительной вероятности).





Распределение значений p-value

Общая постановка задачи

Рассмотрим задачу проверки гипотезы о среднем. Результатом проверки можно считать величину α – критического уровня значимости или p-value. По идее этот коэффициент равен вероятности ошибки 1 рода. То есть примерно в α случаях тест покажет, что p-value < α. Это значит, что величина p-value распределена равномерно.

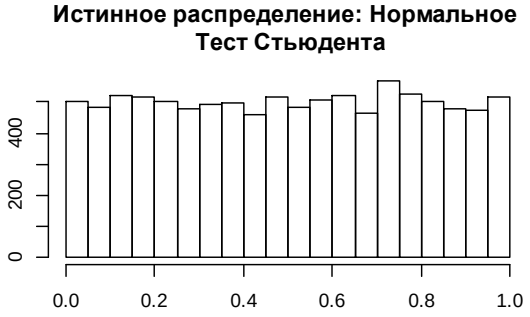
Тест Стьюдента, как известно, предназначен для исследования нормально распределенной с.в. Для с.в. с неизвестной функцией распределения лучше использовать непараметрический критерий, например, Вилкоксона.

Интересный результат получается, если применить критерий Колмогорова для проверки гипотезы о распределении с.в. Этот критерий хорошо сравнивает неизвестное распределение с некоторым конкретным. Если же распределение задано с точностью до некоторых параметров, то их приходится подбирать по выборке. В этом случае качество критерия очень падает.

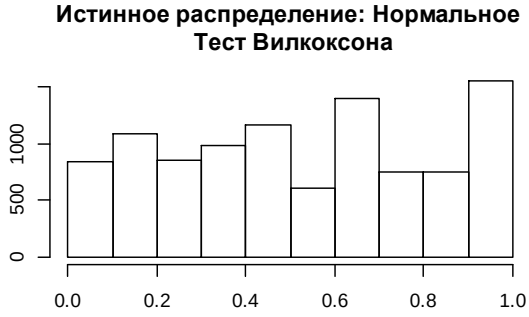
Данные для решения

Создается выборка из выборок из нормально распределенной величины со средним 0. К каждой из них применяется критерий Стьюдента. Набор значений p-value рассматривается как выборка из случайной величины. Для неё строится гистограмма. Кроме того, подсчитывается доля тех выборок, для которых p-value оказалось меньше α.

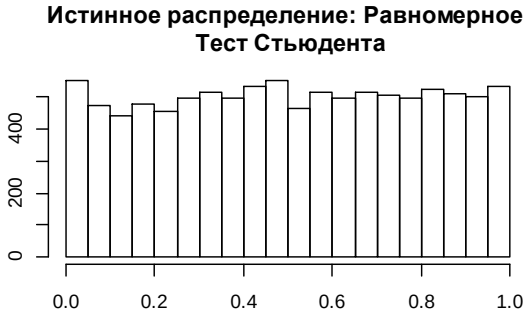
Для проверки равномерности распределения величины p-value применяется критерий Колмогорова.



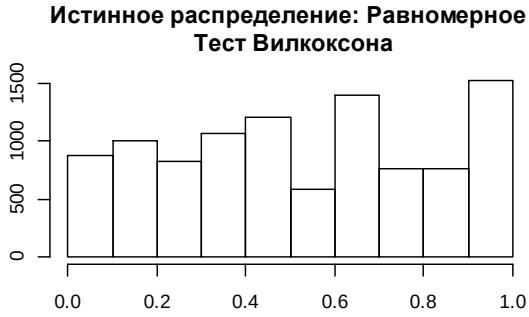
p-value меньше 0.05 для 0.0503 выборок
Равномерность p-value по Колмогорову на уровне 0.64



p-value меньше 0.05 для 0.0489 выборок
Равномерность p-value по Колмогорову на уровне 0



p-value меньше 0.05 для 0.0547 выборок
Равномерность p-value по Колмогорову на уровне 0.082



p-value меньше 0.05 для 0.0494 выборок
Равномерность p-value по Колмогорову на уровне 0

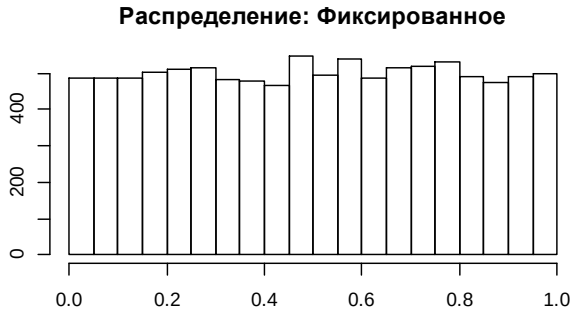
Те же действия повторяются для критерия Вилкоксона, а также для равномерно распределённой случайной величины.

Второе исследование касается проверки равномерности с помощью критерия Колмогорова. Создаётся выборка выборок из нормального (0; 1) распределения. Для каждой выборки делается две проверки. Первая – на принадлежность $N(0; 1)$, вторая – на принадлежность $N(\bar{x}, s)$.

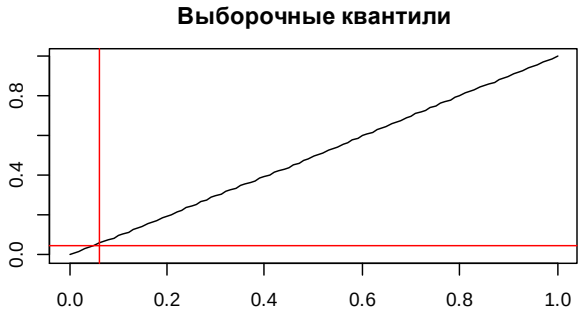
В каждом случае p-value рассматривается как с.в. Строится её гистограмма, а также выборочные квантили с шагом 5%.

Как мы видим, использование выборочных характеристик вместо точных сильно меняют распределение с.в. p-value. На правом графике показаны выборочные квантили распределения p-value. Горизонтальная прямая соответствует уровню $\alpha = 0{,}05$. На верхнем графике это значение соответствует p-value = 0,05. На нижнем графике соответствующее значение α гораздо больше.

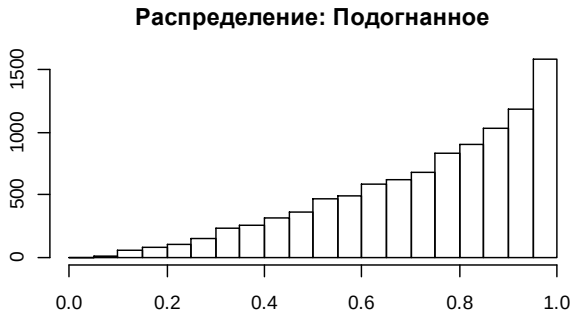
Это значит, что вероятность ошибки первого рода, равная 0,05, достигается при p-value ≈ 30%



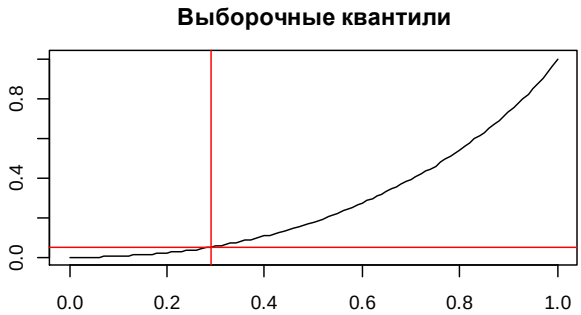
Равномерность p-value на уровне 0.41



p-value меньше 0.05 для 0.0488 выборок
Менее 0.05 выборок при p-value 0.06



Равномерность p-value на уровне 0



p-value меньше 0.05 для 2e-04 выборок
Менее 0.05 выборок при p-value 0.29

Проверка зависимости с.в.

Общая постановка задачи

Для проверки зависимости двух случайных величин можно использовать разные методы. Например, коэффициент корреляции ρ. Он хорошо «улавливает» линейную зависимость между величинами… Если две величины распределены нормально, то независимость в вероятностном смысле совпадает с некоррелированностью, то есть с тем, что ρ = 0.

Выборочная оценка *r* для ρ, конечно, не будет совпадать с нулём. Однако можно проверить статистическую гипотезу H_0: ρ = 0 (в частности, найти критический уровень значимости, p-value).

Заметим, что квадратичная зависимость, даже почти жестко функциональная, не даст большого коэффициента корреляции.

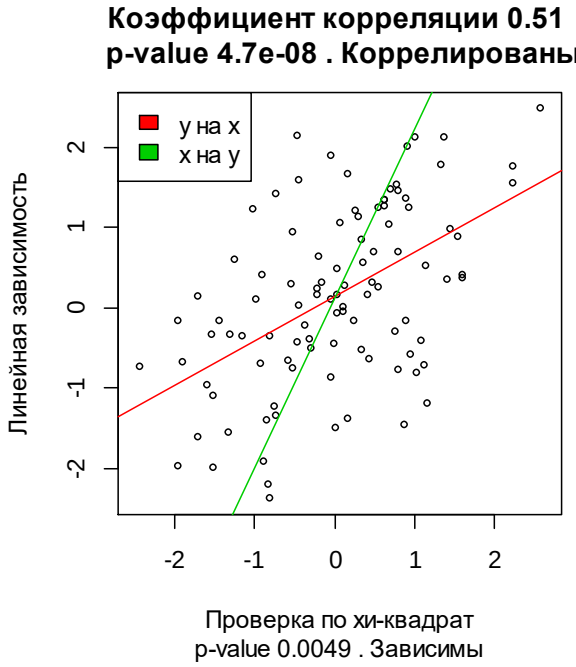
Второй способ проверки зависимости – с помощью критерия хи-квадрат.

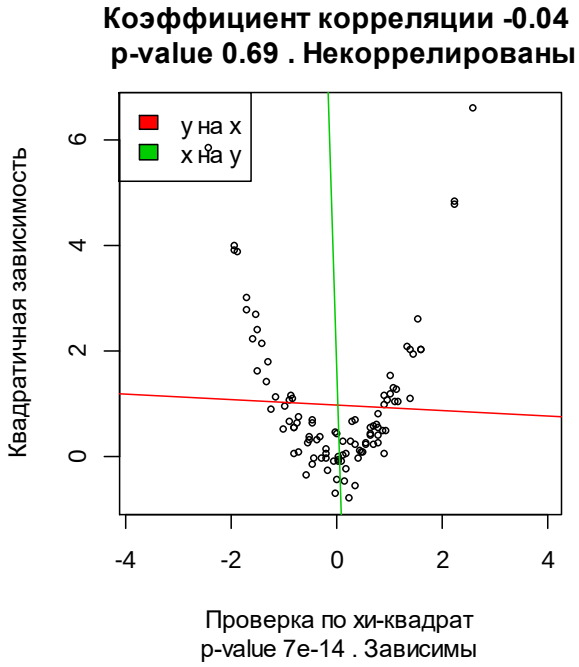
Данные для решения

Рассматриваются две случайные величины. Одна, ξ, распределена нормально, а другая получается из первой по формуле типа η = *f*(ξ) + ε, где ε – нормально распределенная величина. В качестве *f* берётся квадратичная или линейная функция.

Вычисляется коэффициент корреляции и проверяется гипотеза H_0: ρ = 0. Кроме того, строятся линии линейной регрессии, как *x* на *y*, так и *y* на *x*.

Для обоих случаев независимость проверяется с помощью критерия хи-квадрат. Результаты представлены на рисунках.





Результаты проверки могут быть самыми разными. Рекомендуется применить программу несколько раз.

Поиск зависимости между величинами

Общая постановка задачи

В предыдущем пункте мы исследовали, зависимы ли некоторые случайные величины. Если ответ положительный, можно поискать конкретный вид зависимости. Для этого можно использовать ту же команду, которая строит линейные модели. Например, если мы подозреваем, что зависимость – квадратичная функция от x и y, то ее можно рассматривать как линейную от величин x, y, x^2, y^2, xy. Аналогично степенную зависимость $z = cx^a y^b$ легко свести к линейной с помощью логарифмирования: $\ln z = \ln c + a \cdot \ln x + b \cdot \ln y$. Команда *lm*() ищет коэффициенты линейной комбинации так, чтобы они минимизировали сумму квадратов отклонений величин $\ln z_i$ от $\ln c + a \cdot \ln x_i + b \cdot \ln y_i$. Конечно, в такой постановке мы не минимизируем разницу между z_i и $cx_i^a y_i^b$. Тем не менее, какую-то формулу зависимости мы получаем. Разницу этих величин можно посчитать и оценить непосредственно.

Данные для решения

Дан файл данных с расширением *.csv*. В нем три колонки, x, y, z, в первых двух значения заданы датчиком случайных чисел и меняются от 1 до 10. Значения из третьей колонки подсчитываются по двум первым с помощью какой-то формулы вида $z = f(x, y)$. Причем известно, что эта формула в некотором смысле «простая», например, с целыми коэффициентами.

После расчётов значения x, y, z округляются до одного знака после запятой. Это дает погрешность не более 0,05 (относительную погрешность порядка 0,5 – 5 процентов). Распределение этой погрешности, конечно, не является нормальным, однако можно этим пренебречь (в силу малости отклонений).

В программе реализована проверка трёх видов зависимостей.

$z = c_1 + c_2x + c_3y + c_4x^2 + c_5y^2 + c_6xy$;

$z^2 = c_1 + c_2x + c_3y + c_4x^2 + c_5y^2 + c_6xy$;

$z = cx^a y^b$.

Каждая из них проводится независимо, пользователю предоставляется выбор варианта:

```
Поиск зависимости
 квадратичной (z) 1
 квадратичной (z^2) 2
 степенной 3
```

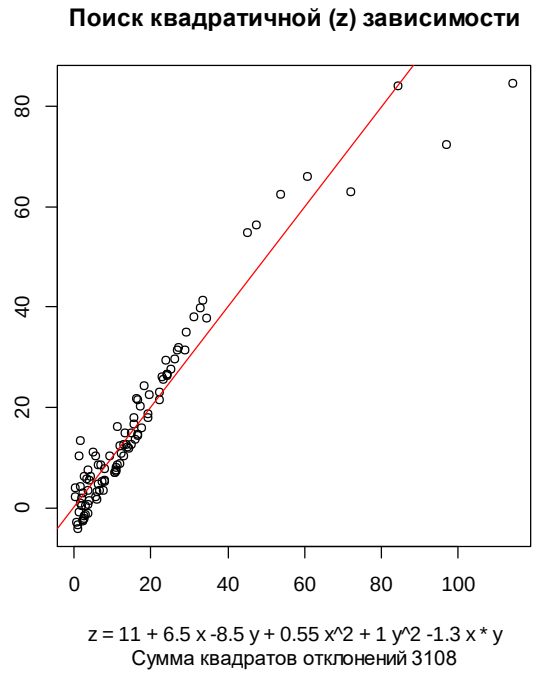
Результатом применения программы являются сведения о линейной модели. Например, если выбран вариант (1), получаем:

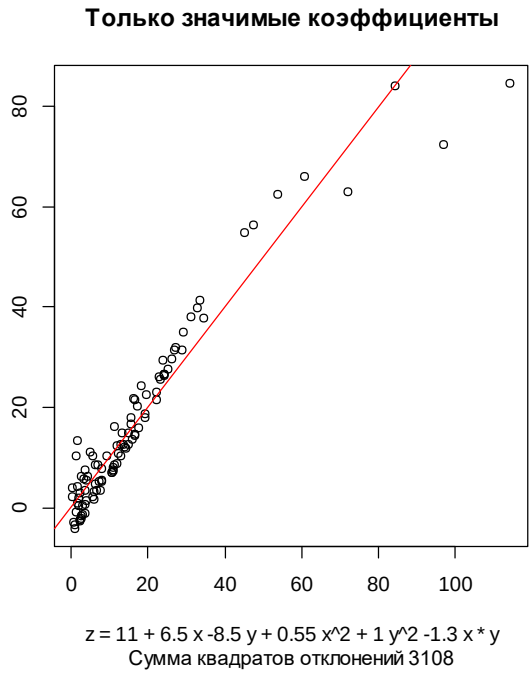
```
Поиск квадратичной (z) зависимости

Call:
lm(formula = formula[[proba]])

Residuals:
     Min       1Q   Median       3Q      Max
-11.6808  -3.0510   0.1203   2.7435  29.7913
```

```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 10.58468    4.71075   2.247    0.027 *
x            6.48298    1.31809   4.918 3.70e-06 ***
y           -8.48364    1.21430  -6.986 4.00e-10 ***
I(x^2)       0.55032    0.11211   4.909 3.85e-06 ***
I(y^2)       1.03698    0.10061  10.307  < 2e-16 ***
I(x * y)    -1.29048    0.09553 -13.509  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 5.75 on 94 degrees of freedom
Multiple R-squared:  0.9191,    Adjusted R-squared:  0.9148
F-statistic: 213.6 on 5 and 94 DF,  p-value: < 2.2e-16
```

Мы видим, что все коэффициенты c_i в линейном выражении значимы (помечены от одной до трёх звёздочек)

Кроме того, на экран выводятся точки с координатами (z_i, $t_i = f(x_i, y_i)$). Если зависимость найдена правильно, то эти точки должны лежать на прямой $z = t$. Она изображена на графике красным цветом:



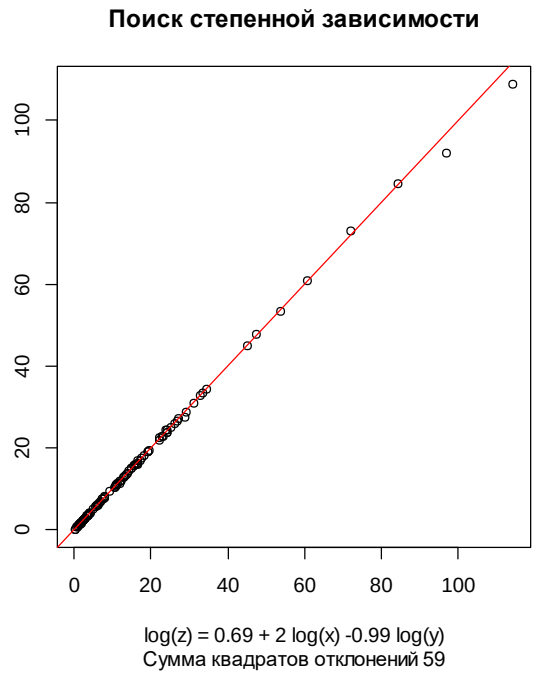


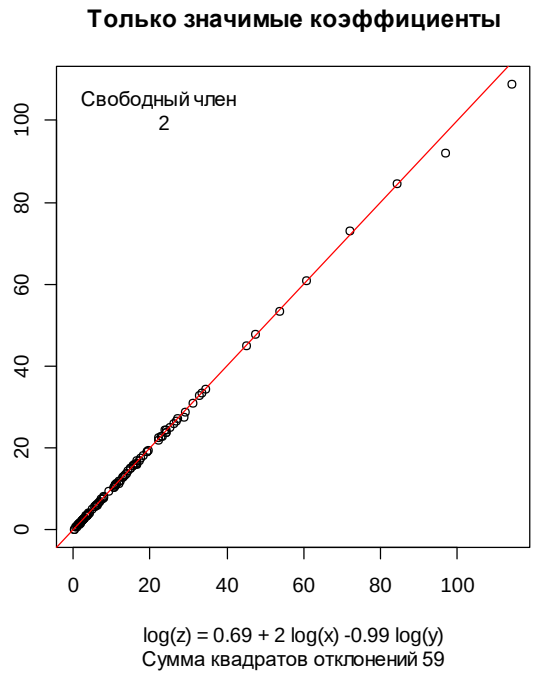
Как мы видим, совпадение не очень хорошее, что видно и по величине суммарного квадратичного отклонения. Такой же расчет по третьей формуле даёт

```
Поиск степенной зависимости

Call:
lm(formula = formula[[proba]])

Residuals:
     Min       1Q   Median       3Q      Max
-0.05990 -0.01244 -0.00106  0.00738  0.12899

Coefficients:
```

```
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.688483   0.009761   70.54   <2e-16 ***
log(x)       1.992792   0.004466  446.20   <2e-16 ***
log(y)      -0.990315   0.004514 -219.39   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.02507 on 97 degrees of freedom
Multiple R-squared:  0.9996,    Adjusted R-squared:  0.9996
F-statistic: 1.136e+05 on 2 and 97 DF,  p-value: < 2.2e-16
```

Графически результат выглядит так:





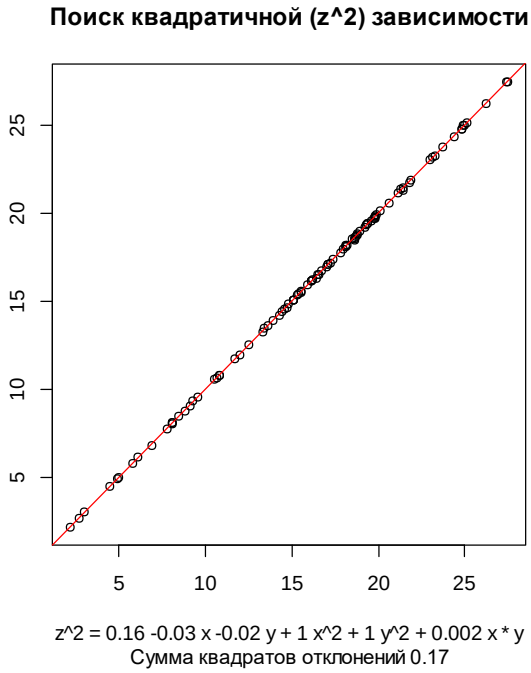
Как мы видим, и визуально, и по значению суммы квадратов отклонений, эта зависимость гораздо точнее. На втором графике указан также свободный член c, логарифм которого равен c_1. Итак, по-видимому, была «задумана» формула $z = \frac{2x^2}{y}$.

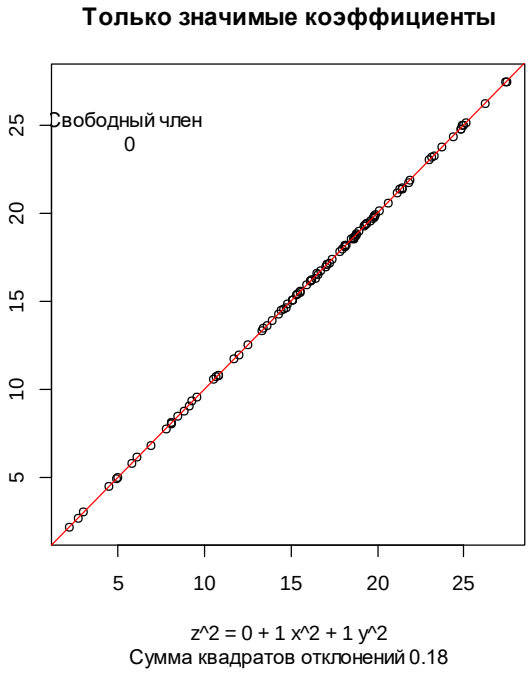
Продемонстрируем ещё случай зависимости второго типа (для другого файла данных).

```
Поиск квадратичной (z^2) зависимости

Call:
lm(formula = formula[[proba]])

Residuals:
    Min      1Q  Median      3Q     Max
-3.9446 -1.1516  0.0198  0.9223  3.4160

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.161111   0.783859   0.206    0.838
x           -0.030232   0.118727  -0.255    0.800
y           -0.020180   0.121471  -0.166    0.868
```

```
I(x^2)        0.999959   0.005350 186.907   <2e-16 ***
I(y^2)        1.000337   0.005496 182.026   <2e-16 ***
I(x * y)      0.001963   0.004390   0.447    0.656
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.543 on 93 degrees of freedom
Multiple R-squared:  0.9999,    Adjusted R-squared:  0.9999
F-statistic: 2.659e+05 on 5 and 93 DF,  p-value: < 2.2e-16
```

Как мы видим, значимы только два коэффициента, программа оставляет их и обнуляет незначимые (правый график):





Мы видим, что запрограммирована была формула $z^2 = x^2 + y^2$.

Приложение

Скрипт «Демонстрационный пример к ЦПТ»

```
#------------------
# Блок функций
#-----------------
cpt<-function(k) {
apply(x[1:k,],2,sum) ->y
hist(y, main=paste("Число слагаемых",k), ylab="",
xlab = paste("По Шапиро", format(shapiro.test(y)$p.value,dig=2)),
sub = paste("По Колмогорову",
format(ks.test(y,"pnorm",k/2,sqrt(k/12))$p.value,dig=2)))
} конец функции

#--------------------
# Основная программа
#--------------------
n<-20; nvib <- 5000
x<-runif(n*nvib)
dim(x)<-c(n,nvib)

par(mfrow=c(2,3)) ->o.p

k<- 1; y <- x[1,]
hist(y, main=paste("Число слагаемых",k), ylab="",
xlab=paste("По Шапиро", format(shapiro.test(y)$p.value,dig=2)),
sub=paste("По Колмогорову", for-
mat(ks.test(y,"pnorm",k/2,sqrt(k/12))$p.value,dig=2)))

for(k in c(2,3,5,10,20)) {cpt(k)}

par(o.p)
```

Скрипт «Сравнение оценок»

```
# Блок функций
#---------------------------------------
# Вывод данных по выборкам
#---------------------
viv_otsen<- function(names) {
plot(a0,xlim = c(1,nvib), ylim = c(min(a),max(a)*1.1),
xlab = "Номер выборки", ylab = "Значения оценок", col = 3, type =
"n",xaxt = "n")
title("Три оценки параметра масштаба
равномерного распределения")
legend("topright",fill = 1:3,legend = names,col = 1:3,cex = 0.6)

for (i in 1:3) { points(a[i,],col = i,pch = i,cex = (5-i)/4) }

abline(h = apply(a,1,mean),col = 1:3)
abline(h = apply(a,1,mean)+apply(a,1,sd),lty = 2,col = 1:3)
abline(h = apply(a,1,mean)-apply(a,1,sd),lty = 2,col = 1:3)

} # конец функции вывода оценок

#---------------------
```

```
# Вывод линий регрессии для пар оценок
#---------------------
regr<-function(k1,k2) {
x<-a[k1,]-a0; y<-a[k2,]-a0
lm(y ~ x)$coefficients->lmc

plot(x, y, xlim = c(min(a)-a0, max(a)-a0),ylim = c(min(a)-
a0,max(a)-a0),
xlab = names[k1], ylab = names[k2], asp = 1,
main = paste("Регрессия",names[k2],"на",names[k1]),
sub = paste("Угловой коэффициент",format(lmc[2])))
abline(0,1, col = 2)
abline(lmc, col = 3)
#title(sub = "Красная прямая y = x")
} # конец функции regr
#---------------------------------------
# Основная программа
#---------------------
# Задание параметров выборок
#---------------------

nvib<-200; n<- 10; a0<-10
a<-rep(0,3*nvib)
dim(a)<- c(3,nvib)

runif(n*nvib,0,a0)-> x
dim(x)<- c(n,nvib)

dev.size("px")
par(mfrow = c(2,1), mar = c(5, 3, 3, 1) + 0.1,
  cex = 0.7, cex.axis = 0.5, pty = "m", tck = 1, tcl = 0)->old.p

#---------------------
# Вычисление оценок

apply(x,2,max)-> a[1,]
apply(x,2,mean)*2-> a[2,]
apply(x,2,median)*2-> a[3,]

names<-c("Исправ. максимум","Среднее","Медиана")
viv_otsen(names)

#---------------------
# Исправление оценок

a[1,]*(n + 1)/n-> a[1,]

names<-c("Исправ. максимум","Среднее","Медиана")
viv_otsen(names)
title(sub = "Исправление оценки для максимума")

#Sys.sleep(10)

#---------------------
# Вывод линий регрессии для пар оценок
```

```
#pdf("Оценки Равномерн.pdf")

par(mfrow = c(2,2),mar = c(5, 5, 3, 1) + 0.1, cex = 0.7, ask = TRUE)

viv_otsen(names); regr(2,1); regr(3,1); regr(2,3)

par(old.p);par(ask = FALSE)
#dev.off()
```

Скрипт «Состоятельность выборочного среднего»

```
nvib<-100; n <- 160; sd<- 1
rnorm(n*nvib,0,sd)-> x
dim(x)<- c(n,nvib)

vivod<-function(tek_razm){
apply(x[1:tek_razm,],2,mean)-> sred

plot(sred,main = paste("Объём выборки",tek_razm),ylim = c(-sd,sd),
   xlab = "Номер выборки", ylab = "Оценка мат. ожидания")
abline(h = mean(sred),lty = 2,lwd = 3)
abline(h = mean(sred) + sd(sred),lty = 2,lwd = 1)
abline(h = mean(sred) - sd(sred),lty = 2,lwd = 1)

abline(h = 0,col = 2)
}

par(mfrow = c(1,3))->old.p
vivod(10); vivod(40); vivod(160)
par(old.p)
```

Скрипт «Доверительный интервал»

```
#--------------------------
# Блок функций
#--------------------------
dover <- function(n){
x<-rnorm(n*N,mu); dim(x)<-c(n,N)
apply(x,2,t.test,mu=mu, conf.level = gamma)->tt

cir<-cil<-rep(0,N)
for (i in 1:N) {
tt[[i]]$conf.int[1]->cil[i]
tt[[i]]$conf.int[2]->cir[i]
}

err <- sum((cil>mu)+(cir<mu))
plot(1:N,rep(0,N),ylim=c(mu-2,mu+2),type="n",xlab=paste("Число
ошибок",err,"  Доля",format(err/N,dig=3)),
main =paste("Размер выборки",n,"  доверительная вероятность",gamma),
ylab="Доверит. интервалы")

abline(h=mu, col = 3)
```

```
for (i in 1:N){lines(c(i,i),
c(cil[i],cir[i]),lwd=2,col=(cil[i]>mu)+(cir[i]<mu)+1) }
} # конец функции dover

mu<-1; N<-200; gamma=0.95

par(mfrow=c(2,1))->op
dover(10); dover(20)
par(op)
```

Скрипт «Распределение значений p-value», часть1

```
#-----------------------------------
# блок функций
#-------------------------
Hist_p <-function(k) {
pv1<-pv2<-NULL

for (i in 1:N) {
x<-funk[[k]](n)
pv1<-c(pv1,t.test(x,mu=mus[k])$p.value)
pv2<-c(pv2,wilcox.test(x,mu=mus[k])$p.value)
}
hist(pv1, ylab="",
    main=paste("Истинное",fnames[k],"\n","Тест",tnames[1]),
    sub=paste("Равномерность p-value по Колмогорову на уровне",
                format(ks.test(pv1,"punif")$p.value,dig=2)),
    xlab=paste("p-value
меньше",alfa,"для",sum(pv1<alfa)/N,"выборок"))

hist(pv2, breaks=10, ylab="",
    main=paste("Истинное",fnames[k],"\n","Тест",tnames[2]),
    sub=paste("Равномерность p-value по Колмогорову на уровне",
                format(ks.test(pv2,"punif")$p.value,dig=2)),
    xlab=paste("p-value
меньше",alfa,"для",sum(pv2<alfa)/N,"выборок"))

} # конец функции Hist_p

#-----------------------------------
# основная программа
#-------------------------
n<-10; N<-10000; alfa<-0.05

funk<-c(rnorm,runif)
mus<-c(0, 0.5)
fnames<-c("Нормальное","Равномерное")
tnames<-c("Стьюдента","Вилкоксона")

par(mfrow=c(2,2))->o.p
Hist_p(1)
Hist_p(2)
par(o.p)
```

Скрипт «Распределение значений p-value», часть2

```
#-----------------------------------
# блок функций
#-------------------------
Hist_p <-function(pris) {
pv<-NULL

for (i in 1:N) {
x<-rnorm(n)
pv<-c(pv,ks.test(x,"pnorm",pris*mean(x))$p.value)
}

hist(pv, ylab="",
     main=paste("Распределение:",tnames[pris+1]),
     xlab=paste("Равномерность p-value на уровне",
                format(ks.test(pv,"punif")$p.value,dig=2)))
c(0,as.numeric(cumsum(table(cut(pv,100))))/N) -> quant

alfa1<-max(which(quant<0.05))/100
plot(0:100/100,quant,type="l",
     main="Выборочные квантили", ylab="",ylim=c(0,1),
     xlab=paste("p-value меньше",alfa,"для",sum(pv<alfa)/N,"выбо-
рок"),
     sub=paste(alfa,"значений для",alfa1,"выборок")
)
abline(h=0.05,col=2)
abline(v=alfa1,col=2)

} # конец функции Hist_p

#-----------------------------------
# основная программа
#-------------------------
n<-10; N<-20000; alfa<-0.05

funk<-c(rnorm,runif)
mus<-c(0, 0.5)
tnames<-c("Фиксированное","Подогнанное")

par(mfrow=c(2,2),mar=c(5,3,3,1)+0.1)->o.p
Hist_p(0)
Hist_p(1)
par(o.p)
```

Скрипт «Проверка независимости величин»

```
#-----------------------------------
# Блок функций
#--------------------------------
korel<-function(funk){
cor.test(x,y)$p.value->cc
m<-round(sqrt(N/10))+1
chisq.test(cut(x,m),cut(y,m))$p.value -> chi
```

```
plot(x,y, cex=0.7,asp=1,xlab="Проверка по хи-квадрат", ylab=funk)

title(main = paste("Коэффициент корреляции",for-
mat(cor(x,y),dig=2),"\n",
     "p-value",format(cc,dig=2),if(cc<0.05){".
Коррелированы"}else{". Некоррелированы"}),
     sub = paste("p-value",format(chi,dig=2),if(chi<0.05){". Зави-
симы"}else{". Независимы"}))

lm(y~x)->xy

abline(xy$coefficients,col=2)
lm(x~y)->yx
yx$coefficients->cc;
abline(c(-cc[1],1)/cc[2],col=3)

legend("topleft",fill = 2:3,legend =c("y на x","x на y"),col = 2:3)
} # конец функции korel

#---------------------------------
#  Основная программа
#-----------------------------
par(mfrow=c(1,2))->op
N<-100;
x<-rnorm(N)

k=0.5; b=0
y<-rnorm(N)+k*x+b
korel("Линейная зависимость")
y<-rnorm(N,0,0.3)+x*x
korel("Квадратичная зависимость")

par(op)
```

Скрипт «Поиск зависимости между величинами»

```
#-----------------------
# Запись формулы зависимости
#-----------------------
podpis <- function(cc){
attr(lmxyz$terms,"term.labels")->term
res<-paste(attr(attr(lmxyz$terms, "factors"), "dimnames")[[1]][1],"=",
format(cc[1],dig=2))

for(i in 1:length(term)){
if (substr(term[i],1,1)=="I") {term[i]<-sub-
str(term[i],3,nchar(term[i])-1)}
if (cc[i+1] != 0)  {if (cc[i+1]>0) {res<-paste(res,"+")}
       res<-paste(res,format(cc[i+1],dig=2),term[i])}
}
result<-res
} # конец функции podpis
f<-function(x){switch(proba,x,sqrt(x),exp(x))}
#-------------------------------
#  Основная программа
#-----------------------------
```

```
# задание имен и формул
#---------------------------
file <-c("Три показателя.csv", "Зависимость.csv")
tip<-c("квадратичной (z)","квадратичной (z^2)","степенной")
formula<-
c(z~x+y+I(x^2)+I(y^2)+I(x*y),z^2~x+y+I(x^2)+I(y^2)+I(x*y),log(z)~log(x
)+log(y))

cat("Поиск зависимости","\n",paste(tip,1:3,"\n"))
readline()->proba; as.numeric(proba)->proba

nfile<-1
#-----------------------
# ввод данных

read.table(file[nfile],header=TRUE,sep=";",dec=",")->xyz

x<-xyz[,1]
y<-xyz[,2]
z<-xyz[,3]

cat("Поиск",tip[proba],"зависимости","\n")
lm(formula[[proba]])->lmxyz
print(summary(lmxyz))

if (proba<3) {rbind(1,x,y,x^2,y^2,x*y)->xy } else
rbind(1,log(x),log(y))->xy
#------------------
# вывод графика зависимости
#-----------------
par(mfrow=c(1,2))->o.p

cc<-coefficients(lmxyz)
res<-podpis(cc)
plot(z, f(cc %*% xy), ylab="", xlab=res,
main=paste("Поиск",tip[proba], "зависимости"))
abline(0,1,col=2)
title(sub=paste("Сумма квадратов отклонений",format(sum((z-f(cc %*%
xy))^2),dig=2)))

cc[summary(lmxyz)$coefficients[,4]>0.05]<-0
res<-podpis(cc)
plot(z,f(cc %*% xy), ylab="", xlab=res, main=paste("Только значимые
коэффициенты"))
abline(0,1,col=2)
title(sub=paste("Сумма квадратов отклонений",format(sum((z-f(cc %*%
xy))^2),dig=2)))

text(max(z)*0.2,max(z)*0.9,paste("Свободный
член","\n",format(f(cc[1]),dig=2)))

par(o.p)
```