10. Статистические вычисления

Библиотека “stats”, поддерживает самые разнообразные статистические вычисления над массивами данных и (что наиболее эффектно) способна генерировать реализации случайных последовательностей с заданными законами распределения.

Подключение библиотеки выполняется стандартно:

```
> with(stats);
```

[anova, describe, fit, importdata, random, statevalf, statplots, transform]

После подключения пользователю становятся доступными одна функция и семь подбиблиотек, которые приведены в таблице.

ФУНКЦИЯ	ВЫПОЛНЯЕМЫЕ ПРЕОБРАЗОВАНИЯ
importdata	Импорт данных из файла
anova	Факторный анализ
describe	Вычисление статистических характеристик массивов данных
fit	Аппроксимация данных заданными зависимостями
random	Генерирование случайных величин
statevalf	Численная оценка статистических функций
statplots	Графическое представление данных
transform	Различные преобразования данных

Библиотека “stats” оперирует с данными, представленными в виде списка, элементом которого может быть:

1) число, например, **[2, 3.1, 5]**;
2) символьное выражение, имеющее единственное значение – **[sin(x), x^2]**;
3) missing – слово, обозначающее недостающее данное – **[3, missing, 2]**:
4) данное, представленное возможным диапазоном его существования – **[1..4]**;
5) Weight(<элемент>,<количество>) – нотация для обозначения элемента, повторяющегося в списке несколько раз.

10.1 Подбиблиотека describe

Эта подбиблиотека позволяет вычислять широкий спектр дискриптивных характеристик, используемых при анализе статистических данных. Формат вызова:

stats[describe, <function>](args)

или

describe[<function>](args),

где (*args*) массив данных, а *<function>* одно из ключевых слов, ука-

ФУНКЦИЯ	ВЫПОЛНЯЕМЫЕ ПРЕОБРАЗОВАНИЯ
coefficientofvariation	Коэффициент вариации
count	Число элементов в списке данных
countmissing	Количество элементов в списке данных, заданных параметром missing
covariance	Значение линейной ковариации между двумя списками
decile	Делит список данных на 10 частей
geometricmean	Геометрическое среднее
harmonicmean	Гармоническое среднее
kurtosis	Коэффициент Куртосиса
linearcorrelation	Коэффициент линейной корреляции для двух списков
mean	Среднее арифметическое
meandeviation	Среднее арифметическое абсолютных отклонений
median	Медиана распределения
mode	Мода распределения
moment,	Моменты
percentile	Нахождение элемента по проценту от доли диапазона всего списка
quadraticmean	Квадратичное среднее арифметическое
quantile	Квантиль
quartile	Квартиль
range	Размах (диапазон изменения данных в списке)
skewness	Коэффициент асимметрии
Standarddeviation	Стандартное отклонение
variance	Вычисление дисперсии

занных ниже и вычисляющих для заданного списка данных следующие характеристики:

coefficientofvariation – коэффициент вариации, как отношение среднего квадратического отклонения σ к среднему арифметическому m.

Например, определим коэффициент вариации для следующего списка:

```
> data1:=[1.,2.,3.,4.,3.,5.,2.,7.,6.,2.,5.]:
> describe[ coefficientofvariation] (data1);
```

.5014

count – число элементов в списке данных (неизвестные элементы, заданные параметром missing, не учитываются). Пример:

```
> describe[count]( [1.,2.,3.,4.,3.,5.,2.,7.,6.,
  2.,5.]);
```

11

countmissing – количество неизвестных элементов в списке данных, заданных параметром missing. Для предыдущего списка, где нет элементов, заданных ключевым словом missing, получим:

```
> describe[countmissing](data1);
```

0

covariance – значение линейной ковариации между двумя списками, вычисляемое по формуле:

$$\frac{\sum_{i=1}^{N}\left(X[i]-\overline{X}\right)\left(Y[i]-\overline{Y}\right)}{N}$$

где X[i] и Y[i] – элементы первого и второго списка с номером i, а $\overline{X}$ и $\overline{Y}$ – средние арифметические значения списков из N элементов. Формат вызова:

describe[covariance](data1, data2);

Поясним использование этой функции на примере двух массивов чисел data1 и data2.

```
> data1:=[1.,2.,3.,4.,3.,5.,2.,7.,6.,2.,5.]:
> data2:=[1,2,4,4,3.,7.,3.,8,5,3,7]:
> describe[covariance](data1,data2);
```

3.553719010

decile – делит список данных на 10 частей. Формат вызова:

describe[decile[which]](data, gap),

где *data* – статистический список, *which* – номер промежутка (число от 1 до 9), *gap* - необязательный параметр, по умолчанию *gap=false*.

Например, граница для первых шести частей списка из девяти приведенных ниже элементов равна 81.

```
> describe[decile[6]]([15,30,45,60,75,90,105,
  120,135]);
```

81

geometricmean – геометрическое среднее, вычисляемое по формуле

$$\sqrt[N]{\prod_{i=1}^{N} X[i]}$$

Для указанного ранее списка data1 геометрическое среднее будет равно:

```
> describe[geometricmean](data1);
```

3.149449927 ;

harmonicmean – гармоническое среднее, определяемое по формуле

$$H = \frac{N}{\sum_{i=1}^{N} \frac{1}{X[i]}} \quad ;$$

Например, для списка [15, 30, 45, 60, 75, 90, 105, 120, 135] геометрическое среднее определим по команде:

```
> describe[harmonicmean]([15, 30, 45, 60, 75,
  90, 105, 120, 135]);
```

340200/7129

kurtosis – коэффициент Куртосиса, используемый для оценки отклонения эмпирического распределения от нормального распределе-

ния по формуле:

$$\frac{\mu_4}{\sigma^4}$$

где – μ_4 центральный момент четвертого порядка, а σ – среднее квадратическое отклонение. Для нормального распределения коэффициент Куртосиса равен трем. Если для оцениваемого распределения коэффициент меньше трех, то его график имеет более пологую вершину, а если большее, то более остроконечную. Например, для двух распределений получим:

```
> evalf(describe[kurtosis]([1,2,3,2,1]));
```

1.846938776

```
> evalf(describe[kurtosis]([1,2,8,2,1]));
```

3.108799049

Примечание: коэффициент Кутосиса – это увеличенный на три коэффициент эксцесса (или эксцесс).

linearcorrelation – коэффициент линейной корреляции для двух списков данных с одинаковым количеством элементов (по определению коэффициент меняется от –1 до 1).

Поясним использование этой функции на примере с ранее определенными массивами чисел data1 и data2.

```
> data1:=[1.,2.,3.,4.,3.,5.,2.,7.,6.,2.,5.]:
> data2:=[1,2,4,4,3.,7.,3.,8,5,3,7]:
> evalf(describe[linearcorrelation] (data1,
data2));
```

.9128215626

mean – среднее арифметическое (начальный момент первого порядка).

Для ранее определенного списка data1, среднее арифметическое значение будет равно:

```
> describe[mean](data1);
```

3.636363636

meandeviation – среднее арифметическое абсолютных отклонений, определяемое по формуле:

$$\frac{\sum_{i=1}^{N}\left|X[i]-\overline{X}\right|}{N}$$

Приведем пример:

```
> describe[meandeviation]([2,4,6]);
```

$$\frac{4}{3}$$

median – медиана распределения. Она делит список на две равные части, например:

```
> evalf(describe[median])([1,2,3,5,6,7]);
```

4

mode – мода распределения. Возвращает последовательность из одного или нескольких элементов, имеющих наибольшую частоту. Например:

```
> q:=(describe[mode])([1,2,3,3,5,5,6,7]);
```

q := 3, 5

moment – моменты, вычисляемые по формуле:

$$M_k = \frac{\sum_{i=1}^{N}(X[i]-Q)^k}{N}$$

где X[i] – элемент списка, N – объем выборки, k – порядок момента, Q – произвольное число, относительно которого вычисляется момент. В зависимости от значения величины Q различают:

Начальные эмпирические моменты k-го порядка, когда Q=0

$$\alpha = \frac{\sum_{i=1}^{N}(X[i])^k}{N}$$

Центральные эмпирические моменты k-го порядка, когда $Q=\overline{X}$

$$\mu_k = \frac{\sum_{i=1}^{N}\left(X[i]-\overline{X}\right)^k}{N}$$

Обычные эмпирические моменты k-го порядка, когда Q произвольное число.

Формат вызова:

describe[moment[k, Q, Nconstraint]](data),

где *data* – список данных; *k* – порядок момента (целое число); *Q* – произвольное число, относительно которого вычисляется момент.

Вычислим начальные моменты 1-го и 8-го порядка относительно нуля для следующего списка [1,2,3,4,5]

```
> describe[moment[1,0]]([1,2,3,4,5]);
```

3

```
> evalf(describe[moment[8,0]] ([1,2,3,4,5]));
```

92595.80000

Вычислим центральный момент второго порядка. Обратите внимание на возможность использования статистических функций как вложенных.

```
> describe[moment[2,mean]]([1,2,3,4,5]);
```

2

percentile – нахождение элемента по проценту от доли диапазона всего списка. Формат вызова:

describe[percentile[which]](data, gap)

где *data* – статистический список данных, *which* – граница заданная величиной процента (меняется от 0 до 100), *gap* – используется, когда в списке присутствуют элементы заданные диапазоном (по умолчанию равно false).

Например, найдем границу, отделяющую 25% элементов списка:

```
> evalf(describe[percentile[25]]([2, 3, 5,
  Weight(6,2)]));
```

2.250000000

quadraticmean – квадратичное среднее арифметическое, вычисляемое по формуле:

$$\frac{\sum_{i=1}^{N}\left(X[i]\right)^2}{N}$$

```
> describe[quadraticmean]([2.0,4,5,6]);
```

4.500000000

quantile – квантиль. Формат команды:

describe[quantile[which,offset]](data, gap)

где *data* – статистический список данных, *which* – заданное значение квантиля (число между 0 и 1), *offset* – положительный или отрицательный сдвиг, прибавляемый к вычисленной позиции (по умолчанию равно 0), *gap* – используется, когда в списке присутствуют элементы, заданные диапазоном (по умолчанию равно false).

Функция quantile выводит границу области, заданной не дискретным (как в функции quartile), а произвольным параметром which.

Найдем границу отделяющую четвертую часть статистического списка:

```
> evalf(describe[quantile[1/4]]([2,3,5,6,6]));
```

2.250000000

quartile – квартиль. Формат вызова:

describe[quartile[which]](data, gap)

где *data* – статистический список данных, *which* – квартиль, одно из целых чисел: 1,2,3; *gap* – используется, когда в списке присутствуют элементы заданные диапазоном (по умолчанию равно false).

Функция quartile сначала разбивает список на четыре равные по количеству элементов части и выводит границу области, заданной параметром which.

Найдем все три значения границ квартилей для списка d.

```
> d:=[2,3,5,6,6]:
> qs:=[seq(describe[quartile[i]],i=1..3)]:
> qs(d);
```

$$\left[\frac{9}{4}, 4, \frac{23}{4}\right]$$

range – размах (диапазон изменения данных в списке).

```
> describe[range]([2,1,3,6,9,7]);
```

1 .. 9

Последняя команда показывает, что диапазон изменения элементов в списке лежит от 1 до 9.

skewness – коэффициент асимметрии, вычисляется по формуле:

$$\frac{\mu_3}{\sigma^3}$$

где μ_3 – центральный момент третьего порядка, σ – среднее квадратическое отклонение.

Если данные распределены симметрично, то коэффициент асимметрии равен нулю. Отрицательное значение присуще распределению смещенному вправо, а положительное – влево.

standarddeviation – стандартное отклонение.

```
> describe[standarddeviation]([1, 2, 3]);
```

.8164965809

variance – вычисление дисперсии (центрального момента второго порядка).

```
> describe[variance]([1, 2, 3]);
```

$$\frac{2}{3}$$

Maple позволяет создавать собственную универсальную функцию, при помощи которой далее можно вычислять несколько показателей одновременно. Например, создадим функцию **gen**, которая сразу определяет количество элементов в данных, среднее значение и дисперсию.

```
> gen:=[describe[count],describe[mean],
  describe[variance]]: gen(data1);
```

[11, 3.636363636, 3.322314050]

10.2 Подбиблиотека fit

Включает две функции: **leastsquare[vars]** и **leastmediansquare.**

Функция **leastsquare[vars]** использует метод наименьших квадратов и предназначена для нахождения корреляционных отношений и для аппроксимации статистических данных выбранными зависимостями.

Формат вызова:

stats[fit, leastsquare[vars, eqn, parms]](data)

или

fit[leastsquare[vars, eqn, parms]](data),

где *data* – список данных; *vars* – список переменных, в порядке представления данных; *eqn* – аппроксимирующее уравнение (по умолча-

нию линейное); *parms* – множество параметров, которые будут заменены на вычисленные значения.

Ниже аппроксимируются три массива. В качестве переменных выбраны **x, y, z**. Поскольку вид аппроксимирующего выражения не оговаривается, то по умолчанию система выбирает его линейным.

```
> fit[leastsquare[[xx,yy,zz]]]([[2,3,4,6,6],
[3,5,7,9,9],[ 4,6,8,11,Weight(15,2)]]);
```

$$zz = -\frac{5}{6} + \frac{11}{3}\,xx - \frac{5}{6}\,yy$$

При описании данных удобно пользоваться опцией Weight(элемент, число повторений):

```
> fit[leastsquare[[x,y,z]]]([[1,2,3,5,5],
[2,4,6,8,8], [3,5,7,10,Weight(15,2)]]);
```

$$z = 1 + \frac{13}{3}\,x - \frac{7}{6}\,y$$

Возьмем два массива **X0** и **Y0**, состоящих каждый из четырех элементов, и аппроксимируем эти данные уравнением второго порядка. На показанных ниже трех примерах проследим, как можно менять формат команды.

```
> X0:=[2,3,4,5]:   Y0:=[0,5,10,14]:
> Ur_1:= fit[leastsquare[[x,y], y=a*x^2+b*x+c,
  {a,b,c}]]([X0, Y0]);
```

$$Ur\_1 := y = -\frac{1}{4}\,x^2 + \frac{129}{20}\,x - \frac{239}{20}$$

```
> Ur_3:= fit[leastsquare[[x,y], y=a*x^2+b*x+c,
  {a,b}]]([X0, Y0]);
```

Функция **leastmediansquare** использует так называемый метод наименьших медиан квадратов отклонения. Поясним это на следующем примере.

```
> data1:=[[1,2],[2,3],[3,4],[1,1]]:
  data := convert(linalg[transpose]
```

```
   (data1),listlist):
   fit[leastmediansquare[[x,y]]](data);
> plot({op(2,"),data1},x=-2..4, y=0..5,
   style=POINT);
```

Поясним вычисления по этому методу. Выбирается две точки, например, [1,2] и [2,3]. Через них проводится прямая линия, в нашем случае y=1+x, и рассчитываются квадраты расстояния от лини до всех точек: [0,0,0,sqrt(2)/2]. Далее определяется медиана для полученного списка, которая равна нулю. Проделав подобные вычисления для всех возможных линий, находят минимальное значение. Линия, соответствующая этому минимальному значению медианы, и будет результатом.

10.3 Подбиблиотека transform

Подбиблиотека содержит богатые возможности для выполнения преобразований над данными, что видно при рассмотрении ее содержания, приведенного ниже. Формат вызова:

stats[transform, <function>](args)

или

transform[<function>](args)

где *args* – списки данных, *<function>* одно из ключевых слов (см. таблицу).

Рассмотрим некоторые возможности на примерах.

Зададим массив данных data3:

ФУНКЦИЯ	ВЫПОЛНЯЕМЫЕ ПРЕОБРАЗОВАНИЯ
apply	Замена элементов данных новыми, вычисленными по формуле
classmark	Заменяет классы данных их средними значениями
frequency	Подсчитываются веса элементов данных
deletemissing	Вычитание пустых элементов из списка данных
divideby	Деление каждого элемента на число или статистическую функцию
frequency	Частотность каждого входящего элемента в данных
moving	Вычисление средних значений последовательных порций элементов
multiapply	Преобразование по формуле данных, представленных списками
remove	Вычитание из данных числа или статистической функции
scaleweight	Умножение весов данных на число
split	Разбить данные на множество списков с соответствующими весами
standardscore	Замена элементов данных
statsort	Сортировка статистических данных
statvalue	Дать величину каждого элемента данных и элементов множества, описанные весами, как единичный элемент
tally	Подсчет повторяемости каждого элемента
tallyinto	Подсчет повторяемости каждого элемента в определенном классе

```
> data3:=[1,1,2,3,3,4,4,4,5,6,6,9]:
```

Подсчитаем веса, входящих в него элементов:

```
>  y45:=transform[tally](data3);
```

y45 := [Weight(3, 2), Weight(4, 3), 5, Weight(6, 2), 9, 2, Weight(1, 2)]

Выделим из полученного списка **y45** первый элемент:

```
> op(1,y45);
```

Weight(3, 2)

Подсчитаем, сколько элементов попадает в указанные диапазоны:

```
> y40:= transform[tallyinto](data3,
  [0..5,5..10,10..15]);
```

y40 := [Weight(0 .. 5, 8), Weight(5 .. 10, 4), Weight(10 .. 15, 0)]

После этого можно определить частотность одинаковых элементов, например так:

```
> transform[frequency](y40);
```

[8, 4, 0]

Присвоим идентификатору **n2** значение числа элементов, указанное во втором члене полученного списка **y40:**

```
> n2:=op(2,op(2,y40));
```

n2 := 4

Далее можно воспользоваться переменной **n2.** Например, вычислим такое выражение:

```
> n2*n2-1;
```

15

Вычтем из каждого элемента списка data3 среднее значение этого списка:

```
> transform[remove[mean]](data3);
```

[–3, –3, –2, –1, –1, 0, 0, 0, 1, 2, 2, 5]

Разделим элементы списка data3 на 4.

```
> transform[ divideby[2]](data3);
```

$$\left[\frac{1}{2}, \frac{1}{2}, 1, \frac{3}{2}, \frac{3}{2}, 2, 2, 2, \frac{5}{2}, 3, 3, \frac{9}{2}\right]$$

Найдем средние значения трех последовательных элементов в списке data3. Очевидно, что число элементов в новом списке будет меньше на 3.

```
> transform[moving[3]](data3);
```

$$\left[\frac{4}{3}, 2, \frac{8}{3}, \frac{10}{3}, \frac{11}{3}, 4, \frac{13}{3}, 5, \frac{17}{3}, 7\right]$$

Вычислим интегральную характеристику накопления весов элементов в данных:

```
> transform[cumulativefrequency]([5,Weight
  (1..7,10),2,2,2,2]);
```

[10, 11, 12, 13, 14]

Если есть некоторая функция, то можно выполнить преобразования над списком данных. Например, определим функцию f, возвращающую третью степень аргумента.

```
> f:=x->x^2;
```

$$f := x \to x^2$$

```
> Mass_1:=transform[apply[f]]([1,3,5]);
```

Mass_1 := [1, 9, 25]

Затем найдем разность между соответствующими элементами данных:

Дискретные распределения

КОМАНДА	РАСПРЕДЕЛЕНИЕ	ОПРЕДЕЛЕНИЕ
binomiald[n,p]	биномиальное	$\frac{n!\,p^x\,(1-p)^{(n-x)}}{x!\,(n-x)!}$
discreteuniform[a,b]	дискретное равномерное	$\frac{n!}{x!\,(n-x)!}$
empirical[list_prob]	эмпирическое	равно 0, если x<1 или x>nops(list_prob), иначе равно list_prob[x]
hypergeometric[N1, N2,n]	гипергеометрическое	$\frac{C_{N_1}^{x} C_{N_2}^{n-x}}{C_{N_1+N_2}^{n}}$
negativebinomial[n, p]	отрицательное биномиальное	$C_{n+x-1}^{n-1} p^n (1-p)^x$
poisson[mu]	Пуассона	$\frac{e^{(-\mu)}\,\mu^x}{x!}$

Непрерывные распределения

КОМАНДА	РАСПРЕДЕЛЕНИЕ	ОПРЕДЕЛЕНИЕ
beta[nu1, nu2]	бета	$\frac{x^{(\nu 1 - 1)} (1 - x)^{(\nu 2 - 1)}}{B(\nu 1, \nu 2)}$
cauchy[a, b]	Коши	$\frac{1}{\pi b \left(1 + \frac{x - a}{b}\right)^2}$ (b>0)
chisquare[nu]	хи–квадрат	$\frac{x^{\left(\frac{1}{2}\nu - 1\right)} e^{\left(-\frac{1}{2}x\right)}}{2^{\left(\frac{1}{2}\nu\right)} \Gamma\left(\frac{1}{2}\nu\right)}$ (x>0, ν>0)
exponential[alpha, a]	экспоненциальное	$\alpha e^{(-\alpha (x - a))}$ при x≥a и равно 0 при x<a
fratio[nu1, nu2]	Фишера (F – распределение)	$\frac{\Gamma\left(\frac{1}{2}\nu 1 + \frac{1}{2}\nu 2\right)\left(\frac{\nu 1}{\nu 2}\right)^{\left(\frac{1}{2}\nu 1\right)} x^{\left(\frac{1}{2}\nu 1 - 1\right)}}{\Gamma\left(\frac{1}{2}\nu 1\right)\Gamma\left(\frac{1}{2}\nu 2\right)\left(1 + \frac{\nu 1 f}{\nu 2}\right)^{\left(\frac{1}{2}\nu 1 + \frac{1}{2}\nu 2\right)}}$ x≥0, ν1>0, ν2>0
gamma[a, b]	гамма	$\frac{x^{(a - 1)} e^{\left(-\frac{x}{b}\right)}}{\Gamma(a)\, b^a}$ x>0, a>0, b>0

Непрерывные распределения (продолжение)

КОМАНДА	РАСПРЕДЕЛЕНИЕ	ОПРЕДЕЛЕНИЕ
laplaced[a, b]	Лапласа	$\frac{1}{2}\frac{e^{\left(-\frac{\lvert x-a\rvert}{b}\right)}}{b}$ b>0
Logistic[a, b]	логистическое	$\frac{x-a}{b^2\left(1+e^{\left(-\frac{x-a}{b}\right)}\right)^2}$ b>0
lognormal[mu, sigma]	логарифмически нормальное	$\frac{1}{2}\frac{\sqrt{2}\,e^{\left(-\frac{1}{2}\frac{(\ln(x)-\mu)^2}{\sigma^2}\right)}}{x\sqrt{\pi}\,\sigma}$ x>0
normald[mu, sigma]	нормальное (Гаусса)	$\frac{1}{2}\frac{e^{\left(-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2}\right)}\sqrt{2}}{\sqrt{\pi\,\sigma^2}}$
studentst[nu]	Стьюдента	$\frac{\Gamma\left(\frac{1}{2}\nu+\frac{1}{2}\right)}{\Gamma\left(\frac{1}{2}\nu\right)\sqrt{\nu\,\pi}\left(1+\frac{t^2}{\nu}\right)^{\left(\frac{1}{2}\nu+\frac{1}{2}\right)}}$
uniform[a, b]	равномерное непрерывное	$\frac{1}{b-a}$ при a≤x≤b, и равно 0 в противном сл учае
weibull[a, b]	Вейбула	$\frac{a\,x^{(a-1)}\,e^{\left(-\left(\frac{x}{b}\right)^a\right)}}{b^a}$ x>0, a>0, b>0

```
> Itog:=transform[multiapply[(x,y)-> x-y]]
  ([[12,40,70], Mass_1]);
```

Itog := [11, 31, 45]

Зададим список элементов data_0, рассортируем его, определим среднее значение и оценим разброс данных:

```
> data_0:=[1,3,5,3]: transform[statsort]
  (data_0); describe[mean](data_0);
  describe[standarddeviation](data_0);
```

[1,3,3,5]

$$3$$

$$\sqrt{2}$$

```
>transform[statvalue]([Weight(3,10),2,2,1,1,1]);
```

[3, 2, 2, 1, 1, 1]

10.4 Подбиблиотека random

Содержит мощные средства генерирования случайных чисел из генеральной совокупности с заданным законом распределения. Формат вызова:

stats[random, distribution](quantity, uniform, method)

или

random[distribution](quantity, uniform, method)

где *distribution* – описание закона распределения получаемых чисел; *quantity* – опция (по умолчанию равно 1) положительное целое число, определяющее сколько случайных чисел нужно получить, или ключевое слово ‘generator’; *uniform* – опция (по умолчанию равно ‘default’) генерирующая числа равномерного распределения; *method* – одно из следующих ключевых слов ‘auto’, ‘inverse’ или ‘builtin’ (по молчанию = ‘auto’).

Библиотека включает следующие шесть дискретных и тринадцать

непрерывных распределений (приведены в таблицах).

Рассмотрим возможности подбиблиотеки random на примерах.

Сгенерируем четыре случайных числа из генеральной совокупности, описываемой нормальным законом распределения:

```
> stats[random, normald](4);
```

.7433, .9421, –.8268, –.8957

Аналогично выше выполненному, создадим массив из 53 элементов, но с заданными параметрами распределения: со средним значением – 7 и среднеквадратическим отклонением – 1.5

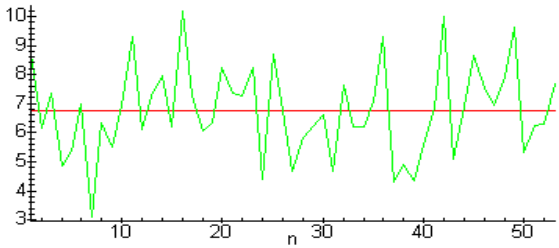
```
> k:=53:
> mas_1:=stats[random, normald[7,1.5]](k):
```

Оценим среднее значение в полученном нами массиве и построим график, на котором покажем последовательность элементов, соединенных линиями, а также линию, показывающую среднее значение.

```
> describe[mean]([mas_1]);
```

7.076

```
> plot({describe[mean]([mas_1]), [[n,op(n,
[mas_1])]$n=1..k]}, n=1..k, style=line);
```



Генерируется четыре числа из гамма распределения:

```
> random[gamma[3]](4);
```

1.514910905, 1.972198351, 5.137418697, 3.631030278

Создадим команду назначения идентификатора с именем gen1 как генератора и получим по ней четыре чисела с точностью представления по умолчанию.

```
> gen1:=random[gamma[3]]('generator'):
'gen1()'$4;
```

4.322417178, 1.556014201, 1.085076020, 7.213556406

Создадим под именем gen2 новый генератор и получим восемь случайных чисел с точность представления до 4:

```
> gen2:=random[gamma[3]]('generator[4]'):
> 'gen2()'$8;
```

4.458, 5.016, 3.087, 5.379, 2.430, 1.485, .6150, 3.078

10.5 Подбиблиотека statevalf

Формат вызова:

stats[statevalf, <function>, <distribution>](args)

или

statevalf[<function>, <distribution>](args),

где *<function>* – одно из следующих выражений.

Для непрерывных функций распределения:

cdf – функция распределения вероятности (интегральная функция);
icdf – обратная функция от интегральной функции;
pdf – функция плотности вероятности (дифференциальная функция).

А для дискретных функций распределения:

dcdf – дискретная функция распределения вероятности;
icdcf – обратная функция;
pf – функция плотности вероятности.

Ниже приведены примеры, поясняющие использование подбиблиотеки statevalf.

Для компактности выводимых результатов, установим точность представления чисел до четырех значащих цифр.

```
> Digits:=4:
```

Определим значение функции плотности распределения Пуассона в точке 2:

```
> stats[statevalf,pf,poisson[3]](2);
```

.2241

Определим значение обратной функции от интегральной функции нормального распределения в точке 0.7:

```
> stats[statevalf,icdf ,normald](0.7);
```

.5244

Определим значение функции плотности вероятности нормального распределения в точке 0.7:

```
> stats[statevalf,pdf,normald](.5);
```

.3519

Для пояснения сказанного построим функцию.

```
> plot(stats[statevalf,pdf,normald](x),x=-3..3);
```

Получим восемь случайных чисел для гамма распределения и оценим их среднее значение:

```
> y3:=stats[random,gamma[3,1]](8);
```

y3 := 1.382, 3.726, 3.936, 3.930, 2.006, 5.439, 1.305, .5604

```
> describe[mean]([y3]);
```

2.786

Определим значение обратной функции от интегральной функции вероятности вейбуловского распределения в точке 0.5:

```
> stats[statevalf,icdf,weibull[3, 2]](0.5);
```

1.770

Получим пять случайных чисел для бета распределения:

```
> stats[random,beta[1,2]](5);
```

.4621, .4244, .2967, .4706, .3968

Получим четыре случайных числа для экспоненциального распре-

деления и оценим их среднее значение:

```
> t1:=stats[random,exponential[3]](4);
  describe[mean]([t1]);
```

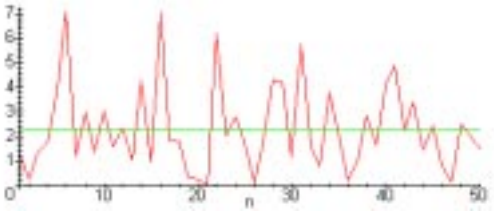
t1 := .01257, .1229, .2387, .04893 .1058

Сгенерируем числа из ^{2}– распределения с точностью до третей значащей цифры при помощи команды из трех операторов: в первом – зададим алгоритм, во втором – сформируем генератор, а в третьем – определим значение:

```
> A:=stats[random,chisquare[2]] (generator[5]):
  t4:='A()'$50: k:=50:
```

Теперь построим реализацию из случайной последовательности чисел, определяемые вейбуловским распределением с заданными параметрами. Для лучшего представления графика зададим ключ соединения точек линиями (style=line).

```
> plot({describe[mean]([t4]),[[n,op(n,[t4])]
$n=1..k]}, n=1..k,style=line);
```



10.6 Подбиблиотека statplots

Предназначена для визуализации статистических данных

Формат вызова:

stats[statplots, <function>](args)

или

statplots[<function>](args)

В качестве *<function>* выбирается один из следующих видов графического представления данных: **boxplot, histogram, notchedbox, quantile, quantile2, scatter1d, scatter2d, symmetry.** Рассмотрим эти функции подробнее.

boxplot – строит рамки, которые иногда называют “ящики с уса-

ми". Рамка состоит из следующих элементов:

1) прямоугольник с линией внутри
   a) внутренняя линия, показывающая медиану;
   b) нижняя сторона , показывающая первый квартиль;
   c) верхняя сторона, показывающая третий квартиль;
2) две линии ("усы"), отходящие от рамки - определяют допустимую область отклонения данных;
3) крайние линии определяют начало области, в которой лежат данные, не входящие в доверительную область.

Функция может иметь параметры, например:

boxplot [9] - указана координата на уровне которой располагается график; **boxplot** [9,1/2] - добавлен параметр ширины прямоугольника.

Такие рамки полезны для сравнения нескольких выборок данных. В качестве примера, можно сравнить изменение дневной температуры месяца с данными по году, отображая два boxplot (бок о бок). Другое возможное использование – изображать его проекцию как одномерный итог, помещенный на край чертежа.

Заметим, что диапазоны данных представляются средним значением, (например 10..12 имеет величину 11), а данные типа mising игнорируются.

notchedbox - эта функция подобна вышеописанной **boxplot** за исключением того, что форма рамки изменяется, информируя о характере изменения квантилей.

scatter2d - эта функция показывает разброс точек в двумерных координатах. Формат вызова:

stats[statplots, scatter2d](xdata, ydata)

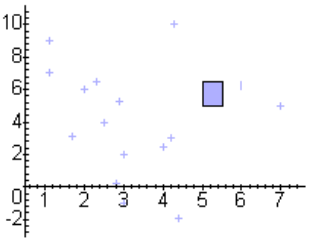
или

statplots[scatter2d](xdata, ydata)

Классы против классов вычерчиваются как прямоугольники, классы против точек вычерчиваются как линии, а точки против точек вычерчиваются как точки.

```
> data1:=[3, 2.5, 2.8, 4.2, 2.9, 4.3, 1.7, 4.4,
  3,4, 2, 1.1, 2.3, 1.1, 7, 5..11/2, 6]:
> data2:=[2, 4, .18, 3, 5.3, 10, 3.1, -1.9, -
```

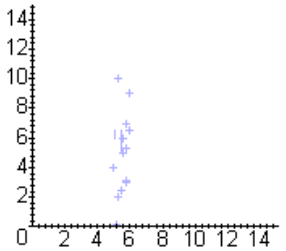
```
  1,2.5, 6, 9, 6.5, 7, 5, 5..13/2, 6..13/2]:
> statplots[scatter2d](data1,data2);
```



scatter1d - эта функция показывает одномерный разброс данных. Если выбран стиль 'projected' (используется по умолчанию), то точки данных вычерчиваются по линии с их координатами. Это позволяет видеть концентрацию данных, но не обнаруживать присутствие регулярности.

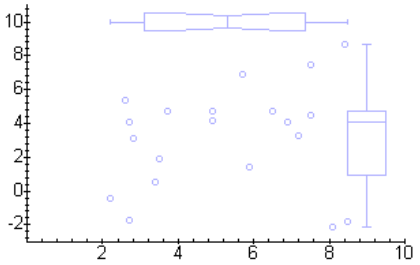
Если выбран стиль jittered, тогда точки с одинаковыми значениями разбрасываются вдоль вертикальной линии. Это дает визуализацию плотности распределения точек. Диапазоны вычерчиваются как линии. Данные типа mising игнорируются. Этот тип графика эффектно применять совместно с двумерным графиком. Для **scatter1d, boxplot** и **notchedbox** применение опции xyexchange изменяет ориентацию рамки на 90 градусов. Опция xshift используется для позиционирования выводимых точек, сдвигая их по оси X на указанную величину. Опция xscale изменяет шкалу x координаты двухмерного графика, умножая ее на указанную величину.

```
> plots[display](statplots[xshift[5]](statplots
[xyexchange] ( statplots[scatter1d[jittered]]
(data2))), view = [0..15, 0..15], axes=frame);
```



Еще раз проиллюстрируем возможности подбиблиотеки **statplot** на двух массивах данных **Xdata** и **Ydata.**

```
> Xdata := [7.5, 4.9, 8.4, 6.5, 7.5, 2.7, 7.2,
  5.9, 6.9, 3.4, 8.5, 5.7, 2.7, 8.1, 3.7, 2.6,
  3.5, 2.2, 2.8, 4.9]:
> Ydata:= [4.5, 4.7, 8.7,4.7, 7.5,-1.7, 3.3,
  1.4, 4.1, .5, -1.8, 6.9, 4.1,-2.1, 4.7, 5.4,
  1.9, -.4, 3.1,4.2]:
> plots[display]({statplots[scatter2d](Xdata,
  Ydata),statplots[boxplot[9]](Ydata),
  statplots[xyexchange](statplots[notchedbox[10]]
  (Xdata))},view =[0..10,-3..11], axes=FRAME,
  symbol=circle);
```



Функция **histogram** - строит гистограммы по статистическим данным. Формат вызова:

stats[statplots, histogram[scale]](data)

или

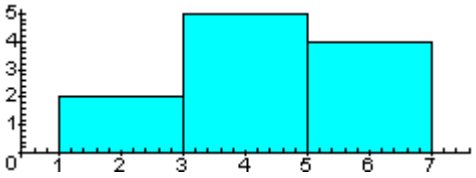
statplots[histogram[scale]](data)

где data - статистический список; scale - число или опция.

Диапазоны данных вычерчиваются как рамки с областью, пропорциональной их весу.

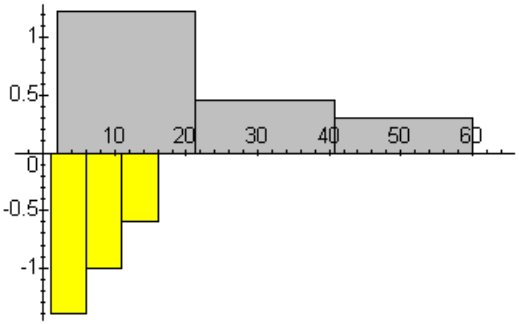
```
> data1:=[ Weight(1..3, 4), Weight(3..5, 10),
  Weight(5..7, 8)]: histogram(data1,
```

```
colour=cyan);
```



Если данные не содержат диапазонов, **histogram** сгруппирует их по зонам (от 5 до 20 зон). По молчанию общая площадь всех зон будет равна 1. Коэффициент масштабирования управляет общей суммой значений зон. Например, построим гистограммы с коэффициентами масштабирования 2 и -3 (знак минус переворачивает гистограмму).

```
> with(stats[statplots]):
> histogram[2]([2,31,33,7,4,50,11,40,16,17,15,
  60,10], colour=gray):
  histogram[-3]([2,1,5,7,4,5,6,11,7,15,5,10,8,5,
  16],colour=yellow):plots[display]({","""});
```

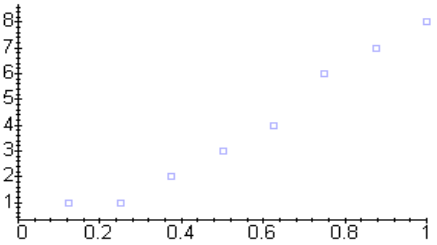


Функция **quantile** – сортирует данные в возрастающем порядке и строит график кумулятивной суммы на отрезке от 0 до 1.

Рассмотрим следующий пример:

```
> statplots[quantile]([6,3,7,4,8,1,2,1],
```

```
symbol=box);
```



Функция **quantile2** – строит график в координатах quantile-quantile. Это хорошее средство для сравнения двух списков данных, например, для сравнения максимальных ежедневных температур в двух городах.

Формат вызова:

stats[statplots, quantile2](data1,data2)

или

statplots[quantile2](data1,data2)

где data1, data2 - статистические списки (могут быть неодинаковыми).

Перед построением данные сортируются в возрастающем порядке.

График quantile-quantile для двух идентичных списков данных даст точки, лежащие на прямой y=x.

Данные типа “диапазон” вычерчиваются как прямоугольники.

Применим сказанное для ранее определенных двух списков Xdata и Ydata.

```
> statplots[quantile2](Xdata,Ydata);
```