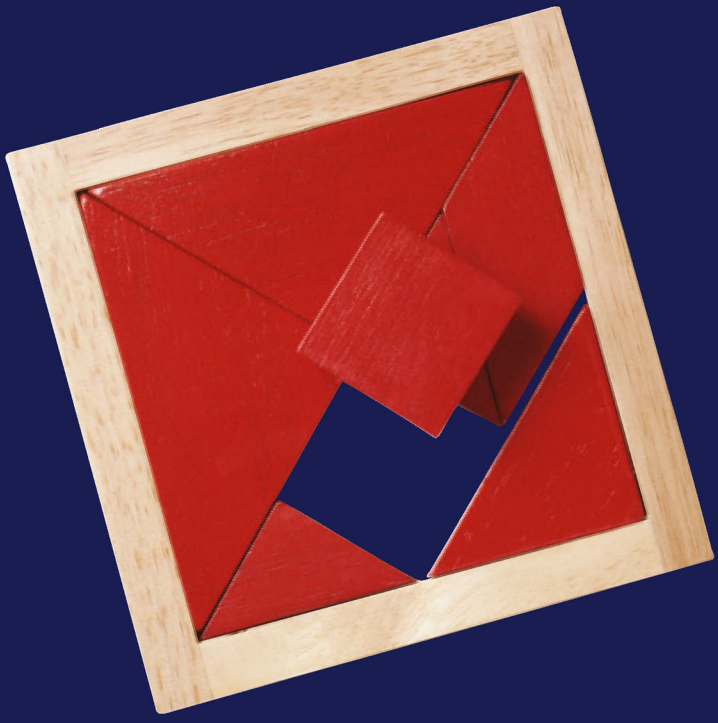
А. Б. Шипунов, Е. М. Балдин, П. А. Волкова,
А. И. Коробейников, С. А. Назарова,
С. В. Петров, В. Г. Суфиянов

Наглядная статистика

Используем R!



ДМК ПРЕСС
ИЗДАТЕЛЬСТВО

УДК 311:004.9R
ББК 60.6с515
Ш63

Ш63 А.Б. Шипунов, Е.М. Балдин, П.А. Волкова, А.И. Коробейников,
С.А.Назарова, С.В. Петров, В.Г. Суфиянов
Наглядная статистика. Используем R! – М.: ДМК Пресс, 2012. – 298 с.: ил.

ISBN 978-5-94074-828-1

Если вам необходима статистическая обработка данных для курсовой, диплома, статьи или диссертации; вы хотите лучше понимать результаты тех статистических методов, которые применяете; вы устали от того, что программы анализа данных не способны выполнить нестандартные задачи; вам необходимо перегруппировать ваши данные, но жаль тратить на это часы ручного труда; вам нужно освоить самые современные методы, еще не нашедшие отражения в большинстве статистических пакетов, то эта книга – для вас!

Изложение построено на базе самого современного программного обеспечения – статистической среды R, которая принадлежит к числу наиболее динамически развивающихся программ в своем классе.

Освоив R, вы сможете: полностью автоматизировать свою работу; запускать статистическую обработку прямо из текста документа; получать графики высокого качества и сохранить их в переносимых графических форматах; в любой момент повторить ваш анализ (например, если поменялись требования к иллюстрациям или исходные данные); использовать сотни «библиотек»-плагинов, разработанных для R; применять самые современные методы; разрабатывать собственные программы анализа данных: от коротких «макросов» до полноценных пакетов, реализующих новейшие алгоритмы; и, естественно, проводить любой стандартный анализ данных, получая при этом графики любой степени сложности.

УДК 311:004.9R
ББК 60.6с515

Все права защищены. Любая часть этой книги не может быть воспроизведена в какой бы то ни было форме и какими бы то ни было средствами без письменного разрешения владельцев авторских прав.

Материал, изложенный в данной книге, многократно проверен. Но поскольку вероятность технических ошибок все равно существует, издательство не может гарантировать абсолютную точность и правильность приводимых сведений. В связи с этим издательство не несет ответственности за возможные ошибки, связанные с использованием книги.

ISBN 978-5-94074-828-1

© А.Б. Шипунов и др., 2012
© Оформление, издание, ДМК Пресс, 2012

Оглавление

Предисловие

Эта книга написана для тех, кто хочет научиться обрабатывать данные. Такая задача возникает очень часто, особенно тогда, когда нужно выяснить ранее неизвестный факт. Например: есть ли эффект от нового лекарства? Или: различаются ли рейтинги двух политиков? Или: как будет меняться курс доллара на следующей неделе?

Многие люди думают, что этот неизвестный факт можно выяснить, если просто немного подумать над данными. К сожалению, часто это совершенно не так. Например, по опросу 262 человек, выходящих с избирательных участков, выяснилось, что 52% проголосовало за кандидата А, а 48% — за кандидата Б (естественно, мы упрощаем ситуацию). Значит ли это, что кандидат А победил? Подумав, многие сначала скажут «Да», а через некоторое время, возможно, «Кто его знает». Но есть простой (с точки зрения современных компьютерных программ) «тест пропорций», который позволяет не только ответить на вопрос (в данном случае «Нет»), но и вычислить, сколько надо было опросить человек, чтобы можно было бы ответить на такой вопрос. В описанном случае это примерно 5000 человек (см. объяснение в конце главы про одномерные данные)!

В общем, если бы люди знали, что можно сделать методами анализа данных, ошибок и неясностей в нашей жизни стало бы гораздо меньше. К сожалению, ситуация в этой области далека от благополучия. Тем из нас, кто заканчивал институты, часто читали курс «Теория вероятностей и математическая статистика», однако кроме ужаса и/или тоски от длинных математических формул, набитых греческими буквами, большинство ничего из этих курсов не помнит. А ведь на теории вероятностей основаны большинство методов анализа данных! С другой стороны, ведь совсем не обязательно знать радиофизику для того, чтобы слушать любимую радиостанцию по радиоприемнику. Значит, для того чтобы анализировать данные в практических целях, не обязательно свободно владеть математической статистикой и теорией вероятностей. Эту проблему давно уже почувствовали многие английские и американские авторы — названиями типа «Статистика без слез» пестрят книжные полки магазинов, посвященные книгам по анализу данных.

Тут, правда, следует быть осторожным как авторам, так и читателям таких книг: многие методы анализа данных имеют, если можно так

выразиться, двойное дно. Их (эти методы) можно применять, глубоко не вникая в сущность используемой там математики, получать результаты и обсуждать эти результаты в отчетах. Однако в один далеко не прекрасный день может выясниться, что данный метод совершенно не подходил для ваших данных, и поэтому полученные результаты и результатами-то назвать нельзя... В общем, будьте бдительны, внимательно читайте про все *ограничения* методов анализа, а при чтении примеров досконально сравнивайте их со своими данными.

Про примеры: мы постарались привести как можно больше примеров, как простых, так и сложных, и по возможности из разных областей жизни, поскольку читателями этой книги могут быть люди самых разных профессий. Еще мы попробовали снизить объем теоретического материала, потому что мы знаем — очень многие учатся только на примерах. Поскольку книга посвящена такой компьютерной программе, которая «работает на текстовом коде», логично было поместить эти самые коды в текстовый файл, а сам файл сделать общедоступным. Так мы и поступили — приведенные в книге примеры можно найти на веб-странице по адресу `http://ashipunov.info/shipunov/software/r/`. Там же находятся разные полезные ссылки и те файлы данных, которые не поставляются вместе с программой.

О структуре книги: первая глава, по сути, целиком теоретическая. Если лень читать общие рассуждения, можно сразу переходить ко второй главе. Однако в первой главе есть много такой информации, которая позволит в будущем не «наступать на грабли». В общем, решайте сами. Во второй главе самые важные — разделы, начиная с «Как скачать и установить R», в которых объясняется, как работать с программой R. Если не усвоить этих разделов, все остальное чтение будет почти бесполезным. Советуем внимательно прочитать и обязательно *проработать все примеры* из этого раздела. Последующие главы составляют ядро книги, там рассказывается про самые распространенные методы анализа данных. Глава «Статистическая разведка», в которой обсуждается общий порядок статистического анализа, подытоживает книгу; в ней еще раз рассказывается про методы, обсуждавшиеся в предыдущих главах. В приложениях к книге содержится много полезной информации: там рассказано о графических интерфейсах к R, приведен простой практический пример работы, описаны основы программирования в R, приведены выдержки из перевода официальной документации. По сути, каждое приложение — это отдельный небольшой справочник, который можно использовать более или менее независимо от остальной книги.

Конечно, множество статистических методов, в том числе и довольно популярных, в книгу не вошли. Мы почти не касаемся статистических моделей, ничего не пишем о контрастах, не рассказываем о стандартных распределениях (за исключением нормального), не объясняем,

как делать многофакторный и блочный дисперсионный анализ, планировать эксперимент, эффектах, кривых выживания, байесовых методах, факторном анализе, геостатистике и т. д., и т. п. Наша цель — научить основам статистического анализа. А если читатель хорошо освоит основы, то любой продвинутый метод он сможет одолеть без особого труда, опираясь на литературу, встроенную справку и Интернет.

Несколько технических замечаний: все десятичные дроби в книге представлены в виде чисел с разделителем-точкой (типа `10.4`), а не запятой (типа `10,4`). Это сделано потому, что программа R по умолчанию «понимает» только первый вариант дробей. И еще: многие приведенные в книге примеры можно (и нужно!) повторить самостоятельно. Такие примеры напечатаны `машинописным шрифтом` и начинаются со значка «больше» — «`>`». Если пример не умещается на одной строке, все последующие его строки начинаются со знака «плюс» — «`+`» (не набирайте эти знаки, когда будете выполнять примеры!). Если в книге идет речь о загрузке файлов данных, то предполагается, что все они находятся в поддиректории `data` в текущей директории. Если вы будете скачивать файлы данных с упомянутого выше сайта, не забудьте создать эту поддиректорию и скопировать туда файлы данных.

Глава 4

Великое в малом: одномерные данные

Теперь, наконец, можно обратиться к статистике. Начнем с самых элементарных приемов анализа — вычисления общих характеристик одной-единственной выборки.

4.1. Как оценивать общую тенденцию

У любой выборки есть две самые общие характеристики: *центр* (центральная тенденция) и *разброс* (размах). В качестве центра чаще всего используются *среднее* и *медиана*, а в качестве разброса — *стандартное отклонение* и *квартили*. Среднее отличается от медианы прежде всего тем, что оно хорошо работает в основном тогда, когда распределение данных близко к нормальному (мы еще поговорим об этом ниже). Медиана не так зависит от характеристик распределения, как говорят статистики, она более *робастна* (устойчива). Понять разницу легче всего на таком примере. Возьмем опять наших гипотетических сотрудников. Вот их зарплаты (в тыс. руб.):

```
> salary <- c(21, 19, 27, 11, 102, 25, 21)
```

Разница в зарплатах обусловлена, в частности, тем, что Саша — экспедитор, а Катя — глава фирмы.

```
> mean(salary); median(salary)
[1] 32.28571
[1] 21
```

Получается, что из-за высокой Катиной зарплаты среднее гораздо хуже отражает «типичную», центральную зарплату, чем медиана. Отчего же так получается? Дело в том, что медиана вычисляется совершенно иначе, чем среднее.

Медиана — это значение, которое отсекает половину упорядоченной выборки. Для того чтобы лучше это показать, вернемся к тем двум векторам, на примере которых в предыдущей главе было показано, как присваиваются ранги:

```
> a1 <- c(1,2,3,4,4,5,7,7,7,9,15,17)
> a2 <- c(1,2,3,4,5,7,7,7,9,15,17)
> median(a1)
[1] 6
> median(a2)
[1] 7
```

В векторе `a1` всего двенадцать значений, то есть четное число. В этом случае медиана — среднее между двумя центральными числами. У вектора `a2` все проще, там одиннадцать значений, поэтому для медианы просто берется середина.

Кроме медианы, для оценки свойств выборки очень полезны *квартили*, то есть те значения, которые отсекают соответственно 0%, 25%, 50%, 75% и 100% от всего распределения данных. Если вы читали предыдущий абзац внимательно, то, наверное, уже поняли, что медиана — это просто третий квартиль (50%). Первый и пятый квартили — это соответственно минимум и максимум, а второй и четвертый квартили используют для робастного вычисления разброса (см. ниже). Можно понятие «квартиль» расширить и ввести специальный термин для значения, отсекающего любой процент упорядоченного распределения (не обязательно по четвертям),— это называется «*квантиль*». Квантили используются, например, при анализе данных на нормальность (см. ниже).

Для характеристики разброса часто используют и параметрическую величину — *стандартное отклонение*. Широко известно «правило трех сигм», которое утверждает, что если средние значения двух выборок различаются больше чем на тройное стандартное отклонение, то эти выборки разные, то есть взяты из разных генеральных совокупностей. Это правило очень удобно, но, к сожалению, подразумевает, что обе выборки должны подчиняться нормальному распределению. Для вычисления стандартного отклонения в R предусмотрена функция `sd()`.

Кроме среднего и медианы, есть еще одна центральная характеристика распределения, так называемая *мода*, самое часто встречающееся в выборке значение. Мода применяется редко и в основном для номинальных данных. Вот как посчитать ее в R (мы использовали для подсчета переменную `sex` из предыдущей главы):

```
> sex <- c("male", "female", "male", "male", "female", "male",
+ "male")
> t.sex <- table(sex)
> mode <- t.sex[which.max(t.sex)]
> mode
male
```

```
5
```

Таким образом, мода нашей выборки — `male`.

Часто стоит задача посчитать среднее (или медиану) для целой таблицы данных. Есть несколько облегчающих жизнь приемов. Покажем их на примере встроенных данных `trees`:

```
> attach(trees) # Первый способ
> mean(Girth)
[1] 13.24839
> mean(Height)
[1] 76
> mean(Volume/Height)
[1] 0.3890012
> detach(trees)
> with(trees, mean(Volume/Height)) # Второй способ
[1] 0.3890012
> lapply(trees, mean) # Третий способ
$Girth
[1] 13.24839
$Height
[1] 76
$Volume
[1] 30.17097
```

Первый способ (при помощи `attach()`) позволяет присоединить колонки таблицы данных к списку текущих переменных. После этого к переменным можно обращаться по именам, не упоминая имени таблицы. Важно не забыть сделать в конце `detach()`, потому что велика опасность запутаться в том, что вы присоединили, а что — нет. Если присоединенные переменные были как-то модифицированы, на самой таблице это не скажется.

Второй способ, в сущности, аналогичен первому, только присоединение происходит внутри круглых скобок функции `with()`. Третий способ использует тот факт, что таблицы данных — это списки из колонок. Для строк такой прием не сработает, надо будет запустить `apply()`. (Если вам пришел в голову четвертый способ, то напоминаем, что циклические конструкции типа `for` в R без необходимости не приветствуются).

Стандартное отклонение, дисперсия (его квадрат) и так называемый межквартильный разброс вызываются аналогично среднему:

```
> sd(salary); var(salary); IQR(salary)
[1] 31.15934
```

```
[1] 970.9048
[1] 6
```

Последнее выражение, дистанция между вторым и четвертым квартилями IQR (или межквартильный разброс), робастен и лучше подходит для примера с зарплатой, чем стандартное отклонение.

Применим эти функции к встроенным данным `trees`:

```
> attach(trees)
> mean(Height)
[1] 76
> median(Height)
[1] 76
> sd(Height)
[1] 6.371813
> IQR(Height)
[1] 8
> detach(trees)
```

Видно, что для деревьев эти характеристики значительно ближе друг к другу. Разумно предположить, что распределение высоты деревьев близко к нормальному. Мы проверим это ниже.

В наших данных по зарплате — всего 7 цифр. А как понять, есть ли какие-то «выдающиеся» цифры, типа Катиной зарплаты, в данных большого, «тысячного» размера? Для этого есть графические функции. Самая простая — так называемый «ящик-с-усами», или боксплот. Для начала добавим к нашим данным еще тысячу гипотетических работников с зарплатой, случайно взятой из межквартильного разброса исходных данных (рис. 9):

```
> new.1000 <- sample((median(salary) - IQR(salary)) :
+ (median(salary) + IQR(salary)), 1000, replace=TRUE)
> salary2 <- c(salary, new.1000)
> boxplot(salary2, log="y")
```

Это интересный пример еще и потому, что в нем впервые представлена техника получения случайных значений. Функция `sample()` способна выбирать случайным образом данные из выборки. В данном случае мы использовали `replace=TRUE`, поскольку нам нужно было выбрать много чисел из гораздо меньшей выборки. Если писать на R имитацию карточных игр (а такие программы написаны!), то надо использовать `replace=FALSE`, потому что из колоды нельзя достать опять ту же самую карту. Кстати говоря, из того, что значения случайные, следует, что результаты последующих вычислений могут отличаться, если

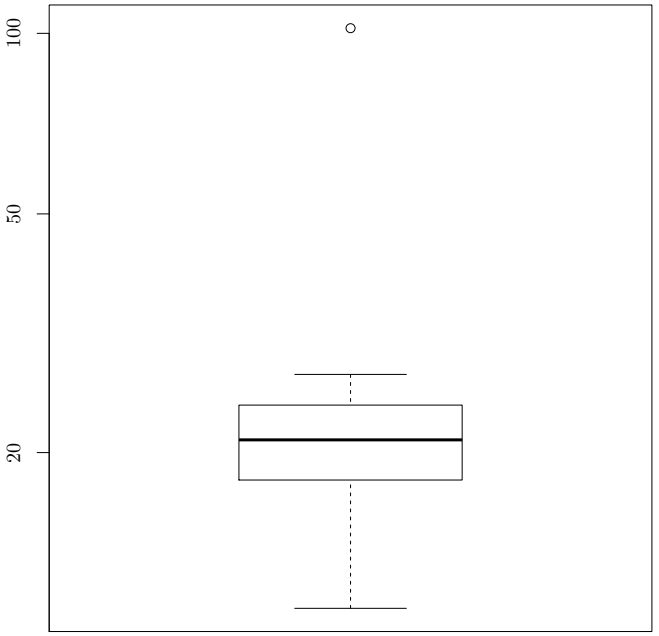


Рис. 9. «Ящик-с-усами», или боксплот

их воспроизвести еще раз, поэтому ваш график может выглядеть чуть иначе.

Но вернемся к боксплоту. Как видно, Катина зарплата представлена высоко расположенной точкой (настолько высоко, что нам даже пришлось вписать параметр `log="y"`, чтобы нижележащие точки стали видны лучше). Сам бокс, то есть главный прямоугольник, ограничен сверху и снизу квартилями, так что высота прямоугольника — это IQR. Так называемые «усы» по умолчанию обозначают точки, удаленные на полтора IQR. Линия посередине прямоугольника — это, как легко догадаться, медиана. Точки, лежащие вне «усов», рассматриваются как выбросы и поэтому рисуются отдельно. Боксплоты были специально придуманы известным статистиком Джоном Тьюки, для того чтобы быстро, эффективно и устойчиво отражать основные робастные характеристики выборки. R может рисовать несколько боксплотов сразу (то есть эта команда *векторизована*, см. результат на рис. 10):

```
> boxplot(trees)
```

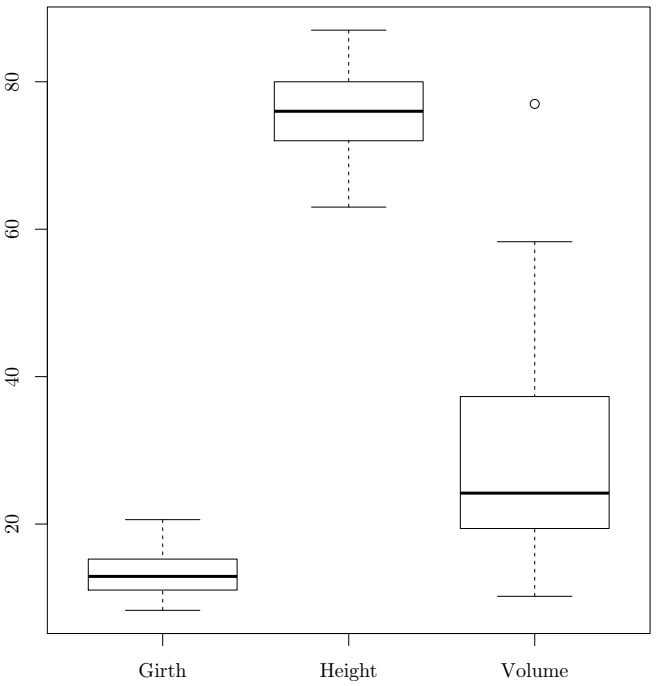


Рис. 10. Три боксплота, каждый отражает одну колонку из таблицы данных

Есть две функции, которые связаны с боксплотами. Функция `quantile()` по умолчанию выдает все пять квартилей, а функция `fivenum()` — основные характеристики распределения по Тьюки.

Другой способ графического изображения — это гистограмма, то есть линии столбиков, высота которых соответствует встречаемости данных, попавших в определенный диапазон (рис. 11):

```
> hist(salary2, breaks=20, main="")
```

В нашем случае `hist()` по умолчанию разбивает переменную на 10 интервалов, но их количество можно указать вручную, как в предложенном примере. Численным аналогом гистограммы является функция `cut()`. При помощи этой функции можно выяснить, сколько данных какого типа у нас имеется:

```
> table(cut(salary2, 20))
(10.9,15.5]   (15.5,20]   (20,24.6] (24.6,29.1] (29.1,33.7]
```

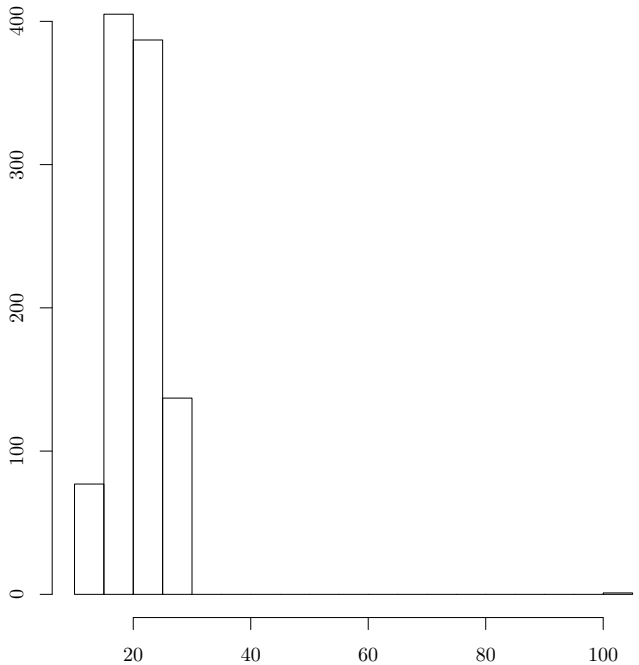


Рис. 11. Гистограмма зарплат 1007 гипотетических сотрудников

```
        76          391          295          244            0
(33.7,38.3]
         0
(38.3,42.8] (42.8,47.4] (47.4,51.9] (51.9,56.5] (56.5,61.1]
         0            0            0            0            0
(61.1,65.6]
         0
(65.6,70.2] (70.2,74.7] (74.7,79.3] (79.3,83.9] (83.9,88.4]
         0            0            0            0            0
  (88.4,93]   (93,97.5]  (97.5,102]
         0            0            1
```

Есть еще две графические функции, «идеологически близкие» к гистограмме. Во-первых, это `stem()` — псевдографическая (текстовая) гистограмма:

```
> stem(salary, scale=2)
  The decimal point is 1 digit(s) to the right of the |
```

```
  1 | 19
  2 | 1157
  3 |
  4 |
  5 |
  6 |
  7 |
  8 |
  9 |
 10 | 2
```

Это очень просто — значения данных изображаются не точками, а цифрами, соответствующими самим этим значениям. Таким образом, видно, что в интервале от 10 до 20 есть две зарплаты (11 и 19), в интервале от 20 до 30 — четыре и т. д.

Другая функция тоже близка к гистограмме, но требует гораздо более изощренных вычислений. Это график плотности распределения (рис. 12):

```
> plot(density(salary2, adjust=2), main="")
> rug(salary2)
```

(Мы использовали «добавляющую» графическую функцию `rug()`, чтобы выделить места с наиболее высокой плотностью значений.)

По сути, перед нами *сглаживание* гистограммы — попытка превратить ее в непрерывную гладкую функцию. Насколько гладкой она будет, зависит от параметра `adjust` (по умолчанию он равен единице). Результат сглаживания называют еще *графиком распределения*.

Кроме боксплотов и различных графиков «семейства» гистограмм, в R много и других одномерных графиков. График-«улей», например, отражает не только плотность распределения значений выборки, но и то, как расположены сами эти значения (точки). Для того чтобы построить график-улей, потребуется загрузить (а возможно, еще и установить сначала) пакет `beeswarm`. После этого можно поглядеть на сам «улей» (рис. 13):

```
> library("beeswarm")
> beeswarm(trees)
> boxplot(trees, add=TRUE)
```

Мы здесь не просто построили график-улей, но еще и добавили туда боксплот, чтобы стали видны квартили и медиана. Для этого нам понадобился аргумент `add=TRUE`.

Ну и, наконец, самая главная функция, `summary()`:

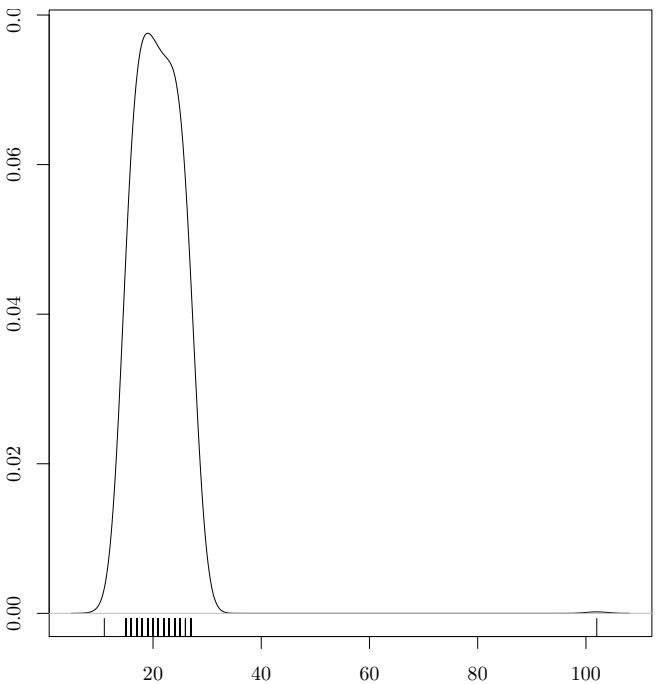


Рис. 12. Плотность распределения зарплат 1007 гипотетических сотрудников

```
> lapply(list(salary, salary2), summary)
[[1]]
   Min. 1st Qu.  Median    Mean 3rd Qu.
  11.00   20.00   21.00   32.29   26.00
   Max.
 102.00

[[2]]
   Min. 1st Qu.  Median    Mean 3rd Qu.
  11.00   18.00   21.00   21.09   24.00
   Max.
 102.00
```

Фактически она возвращает те же самые данные, что и `fivenum()` с добавлением среднего значения (`Mean`). Заметьте, кстати, что у обеих «зарплат» медианы одинаковы, тогда как средние существенно отличаются. Это еще один пример неустойчивости средних значений — ведь с

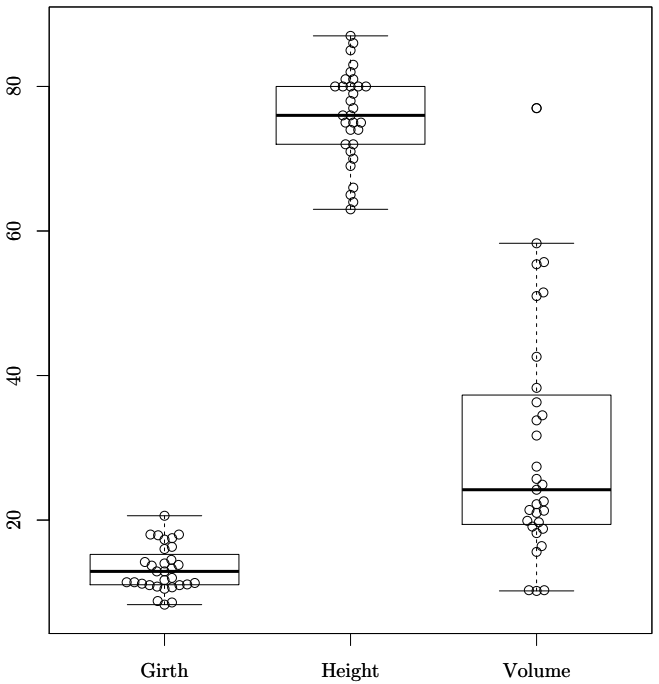


Рис. 13. График-«улей» с наложенными боксплотами для трех характеристик деревьев

добавлением случайно взятых «зарплат» вид распределения не должен был существенно поменяться.

Функция `summary()` — общая, и по законам объект-ориентированного подхода она возвращает разные значения для объектов разного типа. Вы только что увидели, она работает для числовых векторов. Для списков она работает немного иначе. Вывод может быть, например, таким (на примере встроенных данных `attenu` о 23 землетрясениях в Калифорнии):

```
> summary(attenu)

     event           mag           station        dist
 Min.   : 1.00   Min.   :5.000   117    :  5   Min.   :  0.50
 1st Qu.: 9.00   1st Qu.:5.300   1028   :  4   1st Qu.: 11.32
 Median :18.00   Median :6.100   113    :  4   Median : 23.40
 Mean   :14.74   Mean   :6.084   112    :  3   Mean   : 45.60
 3rd Qu.:20.00   3rd Qu.:6.600   135    :  3   3rd Qu.: 47.55
```

```
 Max.   :23.00   Max.   :7.700   (Other):147   Max.   :370.00
                                 NA's   : 16
     accel
 Min.   :0.00300
 1st Qu.:0.04425
 Median :0.11300
 Mean   :0.15422
 3rd Qu.:0.21925
 Max.   :0.81000
```

Переменная `station` (номер станции наблюдений) — фактор, и к тому же с пропущенными данными, поэтому отображается иначе.

Перед тем как завершить рассказ об основных характеристиках выборки, надо упомянуть еще об одной характеристике разброса. Для сравнения изменчивости признаков (особенно таких, которые измерены в разных единицах измерения) часто применяют безразмерную величину — *коэффициент вариации*. Это просто отношение стандартного отклонения к среднему, взятое в процентах. Вот так можно сравнить коэффициент вариации для разных признаков деревьев (встроенные данные `trees`):

```
> 100*sapply(trees, sd)/colMeans(trees)
    Girth    Height    Volume
23.686948  8.383964 54.482331
```

Здесь мы для быстроты применили `sapply()` — вариант `lapply()` с упрощенным выводом, и `colMeans()`, которая просто вычисляет среднее для каждой колонки. Сразу отметим, что функций, подобных `colMeans()`, в R несколько. Например, очень широко используются функции `colSums()` и `rowSums()`, которые выдают итоговые суммы соответственно по колонкам и по строкам (главная функция электронных таблиц!). Есть, разумеется, еще и `rowMeans()`.

4.2. Ошибочные данные

Способность функции `summary()` указывать пропущенные данные, максимумы и минимумы служит очень хорошим подспорьем на самом раннем этапе анализа данных — проверке качества. Предположим, у нас есть данные, набранные с ошибками, и они находятся в директории `data` внутри текущей директории:

```
> dir("data")
[1] "errors.txt" ...
```

```
> err <- read.table("data/errors.txt", h=TRUE, sep="\t")
> str(err)
'data.frame': 7 obs. of  3 variables:
 $ AGE   : Factor w/ 6 levels "12","22","23",..: 3 4 3 5 1 6 2
 $ NAME  : Factor w/ 6 levels "","John","Kate",..: 2 3 1 4 5 6 2
 $ HEIGHT: num  172 163 161 16.1 132 155 183
> summary(err)
 AGE       NAME       HEIGHT
 12:1        :1   Min.   : 16.1
 22:1   John :2   1st Qu.:143.5
 23:2   Kate :1   Median :161.0
 24:1   Lucy :1   Mean   :140.3
 56:1   Penny:1   3rd Qu.:167.5
 a :1   Sasha:1   Max.   :183.0
```

Обработка начинается с проверки наличия нужного файла. Кроме команды `summary()`, здесь использована также очень полезная команда `str()`. Как видно, переменная `AGE` (возраст) почему-то стала фактором, и `summary()` показывает, почему: в одну из ячеек закралась буква `a`. Кроме того, одно из имен пустое, скорее всего, потому, что в ячейку забыли поставить `NA`. Наконец, минимальный рост — 16.1 см! Такого не бывает обычно даже у новорожденных, так что можно с уверенностью утверждать, что наборщик просто случайно поставил точку.

4.3. Одномерные статистические тесты

Закончив разбираться с описательными статистиками, перейдем к простейшим статистическим тестам (подробнее тесты рассмотрены в следующей главе). Начнем с так называемых «одномерных», которые позволяют проверять утверждения относительно того, как распределены исходные данные.

Предположим, мы знаем, что средняя зарплата в нашем первом примере — около 32 тыс. руб. Проверим теперь, насколько эта цифра достоверна:

```
> t.test(salary, mu=mean(salary))
One Sample t-test
data:  salary
t = 0, df = 6, p-value = 1
alternative hypothesis: true mean is not equal to 32.28571
95 percent confidence interval:
  3.468127 61.103302
sample estimates:
```

```
mean of x
 32.28571
```

Это вариант теста Стьюдента для одномерных данных. Статистические тесты (в том числе и этот) пытаются высчитать так называемую тестовую статистику, в данном случае статистику Стьюдента (t-статистику). Затем на основании этой статистики рассчитывается «p-величина» (p-value), отражающая вероятность *ошибки первого рода*. А ошибкой первого рода (ее еще называют «ложной тревогой»), в свою очередь, называется ситуация, когда мы принимаем так называемую альтернативную гипотезу, в то время как *на самом деле* верна нулевая (гипотеза «по умолчанию»). Наконец, вычисленная p-величина используется для сравнения с заранее заданным порогом (уровнем) *значимости*. Если p-величина ниже порога, нулевая гипотеза отвергается, если выше — принимается. Подробнее о статистических гипотезах можно прочесть в следующей главе.

В нашем случае нулевая гипотеза состоит в том, что истинное среднее (то есть среднее генеральной совокупности) равно вычисленному нами среднему (то есть 32.28571).

Перейдем к анализу вывода функции. Статистика Стьюдента при шести степенях свободы (`df=6`, поскольку у нас всего 7 значений) дает единичное p-значение, то есть 100%. Какой бы распространенный порог мы не приняли (0.1%, 1% или 5%), это значение все равно больше. Следовательно, мы принимаем нулевую гипотезу. Поскольку альтернативная гипотеза в нашем случае — это то, что «настоящее» среднее (среднее исходной выборки) не равно вычисленному среднему, то получается, что «на самом деле» эти цифры статистически не отличаются. Кроме всего этого, функция выдает еще и доверительный интервал (confidence interval), в котором, по ее «мнению», может находиться настоящее среднее. Здесь он очень широк — от трех с половиной тысяч до 61 тысячи рублей.

Непараметрический (то есть не связанный предположениями о распределении) аналог этого теста тоже существует. Это так называемый ранговый тест Уилкоксона:

```
> wilcox.test(salary2, mu=median(salary2), conf.int=TRUE)
Wilcoxon signed rank test with continuity correction
data:  salary2
V = 221949, p-value = 0.8321
alternative hypothesis: true location is not equal to 21
95 percent confidence interval:
 20.99999 21.00007
sample estimates:
```