Максим ЭРВЕ

ПУТЕВОДИТЕЛЬ ПО ПРИМЕНЕНИЮ СТАТИСТИЧЕСКИХ МЕТОДОВ С ИСПОЛЬЗОВАНИЕМ R

Планирование исследований и анализ результатов в биологии с помощью программного обеспечения R

Перевод с французского В.К. Шитикова при участии Яндекса

Версия (2016)

Максим Эрве (***Maxime Hervé***) – профессор Университета в Ренне (Франция) и сотрудник Института генетики, окружающей среды и защиты растений. Читает лекции по экологии животных и специализируется по изучению химических аспектов взаимодействия растений с насекомыми-вредителями.
Личная страница: http://www.maximeherve.com

Разработал ряд функций и пакетов для работы в статистической среде R, в том числе:

- `GrapheR` - интерфейс, который позволяет построить наиболее распространенные графики в R, без необходимости знать какие-либо команды;
- `SequenceR` - интерфейс для кодирования поведенческих последовательностей в режиме реального времени;
- функции R для построения потоковых схем связи переменных после преобразования их в факторные координаты.

Мы предлагаем вашему вниманию перевод на русский язык методического руководства *M. Hervé* «**Aide-mémoire de statistique appliquée à la biologie**. *Construire son étude et analyser les résultats à l'aide du logiciel* **R**». Оригинал документа на французском языке можно найти на странице автора и
cran.r-project.org/doc/contrib/Herve-Aide-memoire-statistique.pdf

Путеводитель в значительной мере базируется на пакете `RVAideMemoire` для среды **R**, разработанном автором и содержащем более 100 оригинальных функций, реализующих статистические тесты и построение моделей – см.
cran.r-project.org/web/packages/RVAideMemoire. Функции из этого пакета помечены в командах **R**, приведенных по тексту книги, символом "*".

Переводчик вместе со своим многолетним коллегой Н.А.Цейтлиным (г. Гамбург) сочли необходимым снабдить путеводитель приложением со своими комментариями, подробно разъясняющими точку зрения М.Эрве по отдельным пунктам изложения (или, наоборот, дискутирующими с ней), а также добавить туда некоторые иллюстративные скрипты и примеры расчетов. Мы выражаем также искреннюю благодарность А.Серову (г. Тверь) за помощь в правке рукописи и ценные замечания.

В.Шитиков, д.б.н.
Г. Тольятти, октябрь 2018

ПРЕДИСЛОВИЕ

Не требует специального подтверждения известное мнение, что биологи в целом достаточно холодно относятся к статистике. Этап анализа результатов часто воспринимается ими как вынужденное, обязательное, но крайне неприятное действие, напоминающее иногда восхождение на Голгофу. Тем не менее, основная цель статистики – помочь вам раскрыть те неприметные на первый взгляд закономерности, которые могут содержаться в полученных данных. Игнорировать выполнение статистического анализа из-за отсутствия времени, мотивации или навыков - это, в первую очередь, риск пропустить интересный феномен, который мог бы быть выявлен уже на ваших глазах.

Цель этого путеводителя - дать некоторое руководство к действию для любого биолога, который чувствует потребность в использовании статистики, начиная от планирования исследований, вплоть до анализа полученных результатов. Он определяет некоторую канву, позволяющую вам пройти в одиночку этот достаточно строгий и последовательный путь. Разумеется, это не избавит вас от новых и новых вопросов, а приведенные скрипты кодов **R** нуждаются в приспособлении к вашим собственным данным. Выражаясь фигурально, вы учитесь кататься на велосипеде, и этот путеводитель может явиться парой колес, позволяющих вам не упасть и ехать туда, куда надо. Это обнадеживает, но не забывайте, что, прежде всего, именно вы крутите педали.

С момента написания первой версии этого путеводителя прошло уже шесть лет. За это время я много учил, учился сам и разрабатывал основы статистической методологии для биологов. Многочисленные контакты с коллегами и студентами, привели меня в структуре этой шестой и последней версии, которую я считаю наиболее подходящей для хорошего освоения статистического анализа.

Этим путеводителем охватывается довольно большое количество методов, и я сделал все возможное, чтобы упростить ориентацию в этих "джунглях". Несмотря на это, большая часть пути зависит от самого биолога. Качественно выполненный анализ – это анализ, который обоснованно отвечает на конкретный вопрос или серию вопросов. Золотое правило состоит в том, чтобы, прежде всего, корректно определить поставленную задачу, и никогда не забывать, что наиболее значимым этапом является выбор конкретного статистического метода, адекватного изучаемой проблеме.

Этот справочник напрямую связан с пакетом RVAideMemoire и соответствует его версии 0.9-60.

Я искренне надеюсь, что этот путеводитель удовлетворит ваши ожидания, и позволит почувствовать себя менее одиноким в мире уже не столь кошмарной статистики.

22 августа 2016 года

Максим ЭРВЕ

Путеводитель состоит из трех основных частей:

Часть 1 – **Основы работы с R** (пп. **1-10**). Строго говоря, эта книга не является введением в **R**. Эта часть кратко обозначает только несколько основных концепций, таких как создание и импорт наборов данных, манипуляция основными объектами (векторы, матрицы, таблицы и списки), управление пакетами, а также различные рекомендации, обозначенные как «хорошая практика».

Часть 2 – **Элементы статистической теории** (пп. **11-28**). Эта книга также не является и введением в статистику. Однако здесь приведены некоторые теоретические основы, необходимые для составления плана исследования и правильного анализа его результатов: типы переменных, наиболее распространенные законы распределения, способы оценки выборочных параметров и проверки статистических гипотез и т.д.

Часть 3 – **Анализ результатов исследования** (пп. **29-116**). Это, собственно, основная часть путеводителя, в которой подробно представлены методы обобщения и анализа данных, представленных в одном, двух или более измерениях.

Содержание

1. Основы работы с R

ОСНОВНЫЕ ОБЪЕКТЫ ДАННЫХ

1. Векторы

Вектор является одновременно основным и самым простым элементом языка **R**. Он может быть создан, например, с использованием функции `c()`, которая принимает в качестве аргументов список компонент вектора. Все эти компоненты должны быть одного типа: числовые значения, строковые переменные или уровни фактора.

Пример. Создание численного вектора :

```
> вектор <- c(7, 9, 4, 12, 18)
> вектор
[1] 7 9 4 12 18
```

Пример. Создание численного вектора символьных переменных:

```
> вектор <- c("H","C","I","G","F")
> вектор
[1] "H" "C" "I" "G" "F"
```

Пример. Создание уровней фактора:

```
> вектор <- factor(c("niv1","niv2","niv2","niv3","niv1"))
> вектор
[1] niv1 niv2 niv2 niv3 niv1 Levels: niv1 niv2 niv3
```

Есть функции или аббревиатуры языка, позволяющие сформировать последовательности элементов и упростить создание векторов:

Примеры

```
> 1:10
[1] 1 2 3 4 5 6 7 8 9 10
> seq(from=1, to=3, by=0.25)
[1] 1.00 1.25 1.50 1.75 2.00 2.25 2.50 2.75 3.00
> LETTERS[1:5]
[1] "A" "B" "C" "D" "E"
```

Для доступа к подмножеству элементов вектора `x` необходимо указать `x[i]`, где `i` может быть унитарное значение или другой вектор

Примеры

```
> вектор <- seq(from=2, to=16, by=2)
> вектор
[1] 2 4 6 8 10 12 14 16
> вектор[5]
[1] 10
> вектор[c(2, 5, 8)]
[1] 4 10 16
> вектор[-c(2,5,8)]
[1] 2 6 8 12 14
> вектор[6:3]
[1] 12 10 8 6
```

2. Таблицы

Таблицы представляют собой способ группировки совокупности независимых векторов, располагаемых по столбцам в составе одного объекта. Единственным ограничением является то, что все векторы должны иметь одинаковую длину.

Чтобы создать таблицу, часто используют функцию `data.frame()`, аргументы который определяют последовательность столбцов слева направо (возможно с указанием их названий). Если вектор состоит из наборов символов, то при интеграции в таблицу он автоматически превращается в фактор.

Примеры

```
>вектор1 <- 1:5
>вектор2 <- LETTERS[1:5]
>таблица <- data.frame(вектор1, вектор2)
>таблица
    вектор1    вектор2
1         1          A
2         2          B
3         3          C
4         4          D
5         5          E
```

Таблица может быть также создана с использованием сокращенной записи :

```
>   таблица <- data.frame(вектор1=1:5, вектор2=LETTERS[1:5])
```

Чтобы получить доступ к одному или нескольким элементам таблицы, используется тот же принцип, что и для векторов (см. п. **1**), но необходимо принимать во внимание, что объект *таблица* имеет не одно, а два измерения (т. е. строки и столбцы). Принцип индексирования одинаков для всех объектов с двумя измерениями, т.е. `таблица[строка(s),столбец(s)]`, где `строка(s)` и `столбец(s)`, являются либо унитарными значениями или векторами. Отсутствие индекса до или после запятой означает включение всех строк или столбцов соответственно.

Примеры

```
>   таблица[c(1,3),]
    вектор1    вектор2
1         1          A
3         3          C

>   таблица[c(3,5),2]
[1] C E
Levels: A B C D E
```

Для выделения всего столбца целиком есть еще три возможности

- `таблица$столбец`, где `столбец` – имя столбца;
- `таблица$"столбец"`, где `столбец` – имя столбца в кавычках;
- `таблица[,"столбец"]`, где `столбец` – имя столбца в кавычках.

3. Матрицы

В отличие от таблицы (см. пункт **2**), матрицы представляют единое целое, т. е. столбцы не являются независимыми. Это означает, что все элементы матрицы должны иметь одинаковый тип : численный, символьный, уровни фактора.

Для создания матрицы можно использовать функцию `matrix()`, которая принимает в качестве аргументов конкретные значения, необходимые для заполнения, и количество строк и/или столбцов. По умолчанию заполнение матрицы происходит по столбцам, а для заполнения по строкам необходимо добавить аргумент `byrow=TRUE`. Для того, чтобы определить имена строк и столбцов, можно использовать аргумент `dimnames=list(строки,столбцы)`, где `строки` и `столбцы` являются символьными векторами :

Примеры

```
> матрица <- matrix(1:8, nrow=2)
> matrice
     [,1]  [,2]  [,3]  [,4]
[1,]    1     3     5     7
[2,]    2     4     6     8
> матрица <- matrix(1:8, nrow=2, byrow=TRUE)
> матрица
     [,1]  [,2]  [,3]  [,4]
[1,]    1     2     3     4
[2,]    5     6     7     8
>   матрица <- matrix(1:8, nrow=2,
                      dimnames=list(letters[1:2], LETTERS[1:4]))
>   матрица
   A    B     C     D
a  1    3     5     7
b  2    4     6     8
```

Также можно создавать матрицы из нескольких векторов, которые составляют при этом ее строки или столбцы. Для этого используют функции `rbind()` или `cbind()`, которые объединяют векторы, соответственно, по строкам или столбцам:

Примеры

```
> вектор1 <- 1:3
> вектор2 <- 4:6
> матрица <- rbind(вектор1, вектор2)
> матрица
         [,1] [,2] [,3]
вектор1    1    2    3
вектор2    4    5    6
> матрица <- cbind(вектор1, вектор2)
> матрица
         вектор1 вектор2
[1,]           1       4
[2,]           2       5
[3,]           3       6
```

Матрицы являются объектами с двумя измерениями (строки и столбцы) и их индексация идентична таблицам (см. п. **2**).

4. Списки

Списки – это объекты, аналогичные купе поезда, где каждая камера является полностью независимой от других. Список может одновременно содержать вектор в одном купе, массив в другом, и даже список в третьем.

Чтобы создать список, можно использовать функцию `list()`, которая принимает в своих аргументах содержимое того, что хочется положить в каждый отсек (от первого до последнего). Можно указать имя для каждого отсека, что существенно помогает разобраться с содержимым списка.

Примеры

```
> вектор <- 1:5
> таблица <- data.frame(v1=1:3, v2=LETTERS[1:3])
> list(vecteur, tableau)
[[1]]
[1] 1 2 3 4 5
[[2]]
v1   v2
1    1    A
2    2    B
3    3    C
```

Список может быть создан непосредственно через сокращенную запись :

```
> list(1:5, data.frame(v1=1:3, v2=LETTERS[1:3]))
```

Можно также дать имена отсекам:

```
> список <- list(A=1:5,B=data.frame(v1=1:3, v2=LETTERS[1:3]))
> список
$A [1] 1 2 3 4 5
$B
     v1   v2
1    1    A
2    2    B
3    3    C
```

Для доступа к одному из отсеков списка, можно использовать номер отсека в двойных квадратных скобках: `список[[i]]`, где `i` – уникальное числовое значение. Если в списке отсеки были поименованными, можно также использовать синтаксис `список$имя`, где `имя` – имя купе.

Примеры

```
> список[[1]]
[1] 1 2 3 4 5
```

Или с использование имени компонента списка:

```
> список$A
[1] 1 2 3 4 5
```

Чтобы получить доступ к элементам отсека списка, нужно просто совместить индексацию списка и индексирование содержимого объекта в отсеке.

Примеры

```
> список[[1]][c(2,4)]
[1] 2 4
> список$B$v1[3]
[1] 3
```

ПОДГОТОВКА И ИМПОРТ ДАННЫХ

5. Построение таблицы данных

Правильная организация построения таблицы исходных данных является важным этапом исследования, так как, если это неправильно сделать, то это может привести к ложным результатам, или, чаще всего, к ошибкам в выполнении скриптов **R**.

Структура исходных данных должна прояснить один важный вопрос: какие переменные принимались во внимание в исследовании? Ответ подразумевает определение количественных переменных и факторов, а также градаций этих факторов. Если эти вещи хорошо структурированы, то статистический анализ также не будет вызывать затруднений.

Вообще желательно всегда подготавливать массив своих данных с помощью электронных таблиц. Это позволяет хранить набор данных во внешнем файле, и, следовательно, всегда можно будет легко вернуться к расчетам, поскольку **R** не изменяет внешние файлы (за исключением тех случаев, когда ее об этом попросят).

Простое правило организации таблицы исходных данных: объекты должны быть размещены в строках, а переменные в столбцах. Рекомендуется давать название каждому столбцу, который станет именем переменной в **R**. Необходимо, однако, соблюдать некоторые правила: имена переменных не должны содержать ни пробелов, ни символов арифметических операций, ни символов с надстрочным знаком (это правило существует для всех имен объектов в **R**). Если имя переменной должно содержать два слова, они могут быть разделены точкой (.) или символом подчеркивания (_). Лучше также предпочитать короткие имена, поскольку постоянно набирать в **R** длинные имена переменных медленно и утомительно.

В таблице исходных данных должно обязательно соблюдаться одно правило: ни одно поле не должно быть пустым. Единственным возможным исключением является ячейка в левом верхнем углу: если столбцы имеют названия, то в этом случае 1-й столбец будет принят **R** в качестве имен строк. Если отсутствуют данные для какого-либо объекта, то должна установлена причина, почему это происходит:

- если конкретное значение неясно (отсутствие измерения, неудачный исход, неряшливость заполнения и т.д.), то это – не проблема. Такие данные трактуются как "недостающие" и отмечаются аббревиатурой `NA` (Not Available). Электронные таблицы, совместимые с **R**, признают `NA` и правильно его истолковывают.
- другая ситуация в том, что таблицы исходных данных могут быть плохо построены и, в частности, переменные не были четко определены. Единственный выход напрашивается: уточнить столбцы с переменными и перестроить массив данных.

Не рекомендуется кодирование уровней факторов только цифрами: **R** будет включать эту переменную как числовую (а не как фактор), что может серьезно исказить или даже отклонить анализ.

Если анализ в **R** нужно делать только с некоторым подмножеством исходных данных, или для некоторых видов анализа используется только часть переменных, то рекомендуется сформировать несколько отдельных таблиц данных. Конечно, всегда можно манипулировать исходными массивами непосредственно в **R**, чтобы извлечь какую-то часть или трансформировать некоторые данные, но, если у вас недостаточно опыта, то явно проще (и составляет меньше источников ошибок), когда это делается с оригинальной таблицей наблюдений.

6. Импорт табличных данных в R

Есть много методов для импорта пользовательских данных в **R**. Мы здесь представим только тот, который является одновременно очень простым, работает в большинстве ситуаций и может быть использован для всех платформ ОС.

Процедура выполняется в три этапа :

1. В электронной таблице выберите все строки и столбцы, которые составляют массив анализируемых данных.

2. Скопируйте выделенный фрагмент в Блокнот (*Notepad*) и сохраните файл в формате .txt.

3. Загрузите массив данных в **R** с помощью функции `read.table()` и сформируйте объект типа таблица : `таблица <-read.table("fichier")`, где `fichier` –имя текстового файла с расширением .txt (при необходимости задайте в кавычках путь к каталогу, который ведет к этому файлу).

Статистическая среда **R** использует "англо-саксонскую" языковую настройку, т.е. точку в качестве разделителя целой и дробной части. Если в ваших электронных таблицах (и, следовательно, в текстовом файле) десятичный разделитель – запятая, а массив данных содержит дробные значения, то это необходимо уточнить в **R**, чтобы она интерпретировала запятую в качестве десятичного разделителя. Для этого надо в запись функции `read.table()` добавить аргумент `dec=","`.

Если заголовки столбцов в таблице данных должны быть интерпретированы как имена переменных, то добавьте аргумент `header=TRUE`.

После импорта таблицы исходных данных, полезно убедиться, что она не имеет ошибок при загрузке. Для этого можно получить резюме данных с помощью команды `summary(таблица)`. **R** возвращает сводку по каждой переменной :

- для числовой переменной – показатели вариации выборочных данных, такие как минимум, 1-й квартиль, медиана, среднее значение, 3-й квартиль и "максимум".
- для фактора – количество объектов в каждом уровне.

Если фактор задан в цифровой кодировке (например, в двоичном формате 0/1 или в виде порядковых номеров), **R** интерпретирует его при загрузке как числовую переменную. Для преобразования этой переменной в фактор, необходимо ввести команду `таблица$переменная <- factor(таблица$переменная)`, где `переменная` – имя переменной-фактора.

ПОЛЕЗНЫЕ РЕКОМЕНДАЦИИ

7. Хорошая практика

Существует десятки способов, как использовать **R** лучше, эффективнее и, по возможности, безошибочнее. Некоторые важные правила приводятся ниже.

Программное обеспечение и пакеты

Пакеты не высечены в мраморе и изменяются с течением времени. Авторы исправляют ошибки, добавляют новые функции и т.д. Чтобы сделать доступными эти улучшения/дополнения/исправления, необходимо регулярно обновлять свои пакеты (раз в месяц – это хороший темп). Это очень просто – см. п. **8**.

Программное обеспечение самой **R** также развивается. Номер установленной версии всегда дается в приветствии при запуске программного обеспечения, например, 3.3.1. Первая цифра этого числа очень важна. Действительно, все версии пакетов, созданных после выпуска версии V.x.x. недоступны для пользователей версии [V–1].x.x. Настоятельно рекомендуется следовать за этими крупными обновлениями,

которые все же относительно редки. Однако некоторые пакеты требуют также, чтобы среда **R** была обновлена до некоторой минимальной версии, чтобы они могли нормально функционировать. Поэтому разумно регулярно устанавливать новую полную версию программного обеспечения **R** раз в 1–2 года, что позволяет получить доступ ко всем последним обновлениям (см. п. **10**).

Создание объектов

Кроме обязательных правил именования объектов (не может начинаться с цифры и т. д.), существует несколько правил, которые могут иногда привести к ошибкам статистического анализа (или, в любом случае, усложнят вам жизнь):

- всегда давать объектам информативные имена, чтобы легко их найти (сохраняя при этом лаконичность, поскольку длинные имена, в конечном итоге, приводят к потере времени!)
- всегда создавать свои объекты с помощью синтаксиса `имя <- содержимое`, а не `имя = содержимое`;
- никогда не называть два объекта одинаковыми именами;
- никогда не присваивать объекту имена функций;
- всегда кодировать факторы, по крайней мере, одной буквой (а не в цифровом виде), чтобы **R** могла распознавать эти переменные как факторы.

Использование функций

Если функция принимает в качестве аргумента формулу (см. п. **40**, хотя формулу могут использовать и другие функции, не создающие модель), то она имеет такой аргумент, как `data`. С его помощью можно указать таблицу данных, в которой присутствуют переменные, содержащиеся в формуле. Это не только упрощает написание команды (или придает ей ясность), но и позволяет избежать невынужденных ошибок.

Пример: для любых подобных функций следует использовать вместо:

```
> lm(таблица$y ~ таблица$x + таблица$z)
```

сокращенный синтаксис:

```
> lm(y ~ x + z, data = таблица)
```

Если функция принимает несколько аргументов, то лучше явно называть их по имени, чтобы избежать ошибочной передачи аргументу значений, предназначенных для другого аргумента. Действительно, если не использовать имена аргументов, то следует строго соблюдать их последовательность в точности так, как это указано в справочной документации, а это часто является источником ошибок. Первый или первые два аргумента обычно вне этого правила, поскольку они определены логикой расчетов и тут ошибка маловероятна.

Стоит избегать использования функций `attach()` и `detach()`, которые являются источником многих ошибок. Благодаря аргументу `data` функции, использующей формулы, или взяв на вооружение некоторые функции, такие как `with()`, можно сделать жизнь такой же простой, ничем не рискуя.

Наконец, это очень настоятельно рекомендуется сохранять полностью варианты скриптов выполненного анализа, чтобы вернуться к ним позже (или по крайней мере, быть уверенным, что вы можете это сделать!). Очень хорошая привычка – красиво оформлять свои скрипты, и подробно комментировать их (все, что находится после символа `#`, признается **R** в качестве комментария). Это придает ясность и понимание смысла анализа другим людям (или самому себе позднее). Не забудьте также указать пакеты, необходимые для выполнения скрипта.

8. Установка, загрузка и обновление пакетов

Установка пакетов

Чтобы установить необходимый вам пакет, компьютер должен быть подключен к Интернет, так как его загрузка осуществляется с сервера CRAN (*the Comprehensive R Archive Network*: http://cran.r-project.org). Установка выполняется только один раз.

Если вы используете консоль **R**, введите `install.packages("package")`, где `package` - имя нужного вам пакета в кавычках. Затем **R** попросит выбрать ресурс для скачивания, например, "*Турция*".

Если **R** используется с системной консолью, процедура выполняется в два этапа :

1. Скачайте исходники пакета с сайта его разработки, либо с основного веб-сайта CRAN из раздела *Packages*.

2. Установите пакет, введя команду операционной системы

```
R CMD INSTALL package
```

где **package** – имя архивного файла *tar.gz*, содержащего инсталляцию пакета.

Действия, описанные здесь, просты, но есть и много других вариантов. Посмотрите FAQ по **R** для получения дополнительной информации:
http://cran.r-project.org/doc/manuals/R-admin.html#Installing-packages.

Загрузка пакета

Загрузка пакета должна проводиться для каждой сессии, где он используется. Команда проста : `library(package)`, где `package` – имя пакета без кавычек.

Обновление установленных пакетов

Для автоматического обновления всех установленных пакетов, введите `update.packages(ask=FALSE)`. В этом случае **R** загружает все обновления и устанавливает их. Внимание: обновляются только незагруженные пакеты, поэтому лучше сделать обновления, прежде чем приступить к работе с **R**.

Рекомендуется регулярно обновлять постоянно используемые пакеты, чтобы проследить за их эволюцией, часто достаточно быстрой.

9. Цитирование методов, реализованных в пакетах R

При оформлении статьи исследователь обычно приводит библиографические источники. Он также должен сослаться на программное обеспечение, используемое при проведении статистического анализа. Конечно, **R** – бесплатный продукт, но, тем не менее, десятки людей участвуют в ее развитии, и хорошей практикой является отметить цитированием их работу.

Чтобы узнать, как цитировать использование самой статистической среды **R**, нужно просто ввести `citation()` и скопировать источник, указанный после
To cite R in publications use:.

В отношении пакетов это правило не столь прямолинейно, поскольку трудно упомянуть все пакеты, которые загружаются автоматически или устанавливаются самим пользователем. Тем не менее, чтобы узнать, как осуществить конкретное цитирование, наберите `citation("package")`, где `package` – имя пакета в кавычках и скопировать то, что появится после
To cite the xxx package in publications use:.

10. Изменение версии R

Чтобы установить более новую версию **R**, которая предоставляет доступ к обновленным пакетам или к не поставляемым со старой версии (см. п. **7**), необходимо выполнить процедуру, состоящую из нескольких этапов:

1. Получить путь к папке, в которой установлены пакеты, с помощью `.libPaths()` (в этой папке есть подкаталоги для каждого пакета).
2. Удалить **R**, как любое другое программное обеспечение. Все системные файлы будут также удалены за исключением установленных пакетов.
3. Установить новую версию R, загрузив ее с официального сайта: https://cran.r-project.org.
4. Открыть и восстановить путь к папке, в которой установлены пакеты новой версии, с помощью `.libPaths()`. Этот путь не тот же, что и полученный на шаге 1, поскольку он зависит от конкретной версии.
5. Скопировать все вложенные папки (относящиеся, например, к каждому из пакетов), находящиеся в папке, указанной на шаге 1, и вставить их в папку, указанную на шаге 4. Таким образом, все пакеты, которые были установлены в предыдущей версии **R**, доступны теперь в новой версии.
6. Можно удалить папку, указанную на шаге 1, поскольку она уже бесполезна.
7. Обновить все пакеты, прежде чем приступить к работе (см. п. **8**).

2. Элементы статистической теории

ОБЩИЕ ЗАМЕЧАНИЯ

11. Различные типы переменных

Существует два типа переменных :

1. ***Количественные***: их значения представляют собой численные величины, чаще всего связанные с определенными единицами измерения. С этими переменными можно выполнять математические операции. Количественные переменные могут быть двух типов:

- непрерывными, когда они могут принимать бесконечное множество значений на заданном интервале (масса, время, расстояние, объем и т.д.);
- дискретными, если они могут принимать только определенные значения на заданном интервале. Эти переменные связаны чаще всего с процессом подсчета числа объектов или событий, поэтому наблюдаемые значения могут быть только целыми (положительными или равными нулю).

2. ***Качественные***: их значения не обозначают количество, а только некоторые категории. Поэтому с этими переменными не могут выполняться математические операции. Их называют факторами, и значения, которые они могут принимать, соответствуют градациям (кластерам, уровням или условиям). Категориальные переменные могут быть двух типов :

- порядковые, когда уровни могут быть ранжированы по порядку: место в рейтинге, степень удовлетворенности и т.д.;
- номинальные, когда уровни не могут быть упорядочены : пол, страна...

Категориальные переменные могут быть закодированы в цифровой форме, но очень важно отличать их от количественных переменных, так как они обрабатываются по разному в ходе статистического анализа. Простой принцип, чтобы почувствовать разницу: если можно заменить значения числовой переменной некоторыми высказываниями, то она является качественной. Например, если состояние растения, закодированное в баллах от 0 до 5, можно заменить набором оценок типа "плохо", "удовлетворительно" и т.д., то эта переменная является качественной. Чтобы не рисковать, никогда не рекомендуется кодировать категориальные переменные численными значениями.

Существует два типа факторов:

- с фиксированными эффектами ("основные факторы"), если его уровни были намеренно выбраны, и если целью исследования является их сравнение. Например, если мы хотим сравнить диаметр ствола трех видов деревьев, то фактор «вид», является фиксированным (с тремя уровнями).
- со случайными эффектами ("рандомизированные факторы"), если его уровни были выбраны из большого числа возможных состояний, и если целью исследования является не сравнивать, а просто принять во внимание изменчивость, которая этими состояниями обусловлена. Например, если измерения диаметров стволов трех видов выполнены двумя разными наблюдателями, можно считать фактор "*экспериментатор*" случайным. Цель здесь не сравнить измерения, проводимые двумя исследователями, а только принять во внимание тот факт, что изменчивость результата может определяться манерой проводить замеры.

Есть две обстоятельства,которые необходимо помнить: (1) решение объявить фактор как фиксированный или случайный имеет фундаментальное значение, поскольку одна и та же цель статистического анализа достигается в обоих случаях с использованием принципиально различных алгоритмов построения моделей; (2) это решение должно быть принято в зависимости от цели исследования, т.е. вопроса, на который исследование должно дать ответ, потому что ни один фактор сам по себе не является фиксированным или случайным в абсолютном понимании этого термина. Поэтому необходимо основательно подумать, прежде чем объявить фактор фиксированным или случайным.

Как для факторов, так и для количественных переменных важно оценить, являются ли выполненные наблюдения независимыми между собой, и можно ли их трактовать как повторные серии. Самый простой случай, это когда несколько измерений выполняются на одном и том же объекте (например, до и после лечения больного). Но в ряде случаев необходимо учитывать весьма тонкие ассоциации. Например, если изучается влияние фактора на отдельные группы людей, имеющие между собой родственные связи, то эти наблюдения не являются формально независимыми, так как между группами может существовать корреляция генетической природы. Или, если серии измерений проводятся в различных регионах, то эти наблюдения также не являются независимыми, так как каждая серия зависит от особенностей местной окружающей среды. Если серии измерений проводятся в разное время, то эти наблюдения также не являются независимыми, так как каждая серия зависит от тех событий, которые произошли раньше. Очень важно определить соответствие между собой отдельных подмножеств измерений, и, если с ними связаны посторонние эффекты, то они должны быть учтены при анализе статистических данных. Перечисленные примеры нуждаются во введении уровней случайного фактора: фактора "*семья*", фактора "*местоположение*" и фактора "*время*" соответственно.

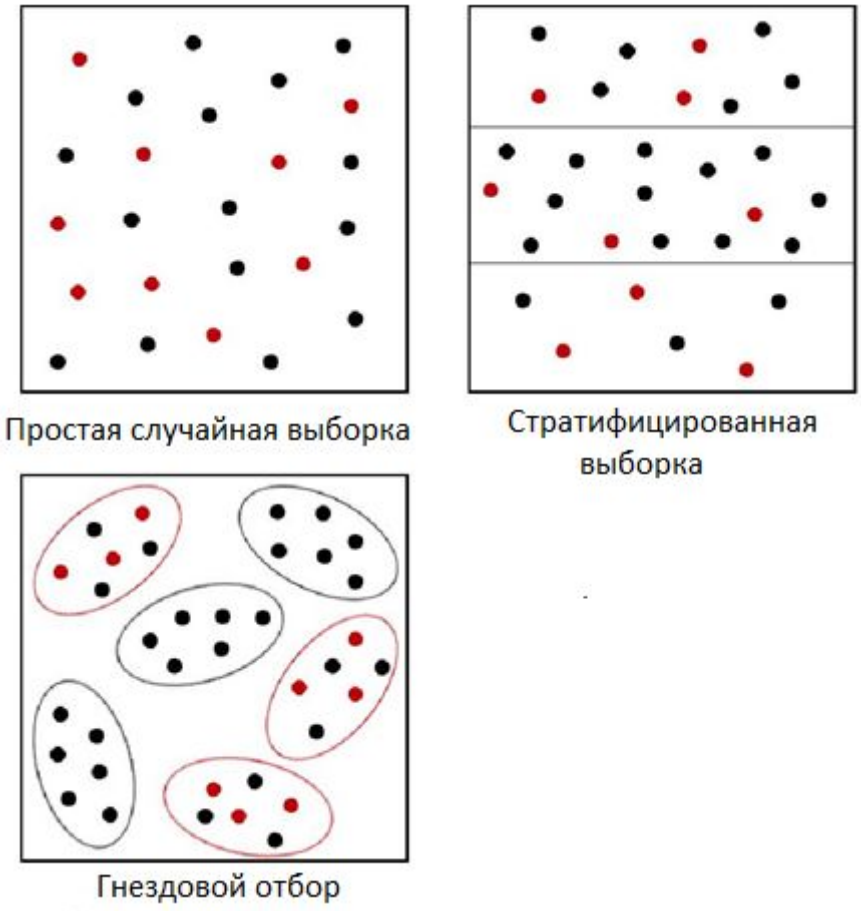
12. Планирование выборки

Обычно в исследованиях используется выборочный метод, поскольку при сборе сведений о группе объектов в их среде обитания далеко не все индивидуумы могут быть доступны. При этом разрабатывается план взятия выборок, определяющий стратегию эксперимента или полевых исследований.

Основные методы отбора проб могут быть отнесены к двум группам:

1. **Случайная выборка**: все объекты (в статистическом смысле) имеют одинаковую вероятность быть выбранными, и каждый выбор независим от других. Существуют различные методы взятия случайной выборки (см. иллюстрации):

- *простая* случайная выборка: выбор осуществляется среди всех объектов популяции (в статистическом смысле), которая составляет одну большую совокупность.
- *стратифицированная* выборка: если популяция очень разнородна, она может быть разделена на суб-эксклюзивные наборы (или ***страты***). В рамках этих подмножеств формируются простые случайные выборки. Страты определяются в последующем статистическом анализе как уровни фиксированного фактора.
- *гнездовой* отбор: если страт очень много, выбирают некоторые случайные кластеры, в рамках которых формируются простые случайные выборки. Выделенные кластеры в ходе статистического анализа трактуются как уровни случайного фактора.
- *иерархическая* выборка: является обобщением гнездового отбора (который является первой ступенью иерархии). В популяции выбираются "первичные" кластеры, а затем внутри них (всегда случайно) – "вторичные" блоки, и так далее... На последнем этапе формируются простые случайные выборки.



Каждая точка представляет один объект.
Отобранные пробы представлены красным.

2. **Систематическая** выборка: первый объект выбирается случайным образом, а затем другие выбираются на регулярной основе в зависимости от предыдущего (во времени или пространстве). Исследование этого типа выборок, который использует пространственный статистический анализ или анализ временных рядов, в этой книге не рассматривается.

13. Планирование эксперимента

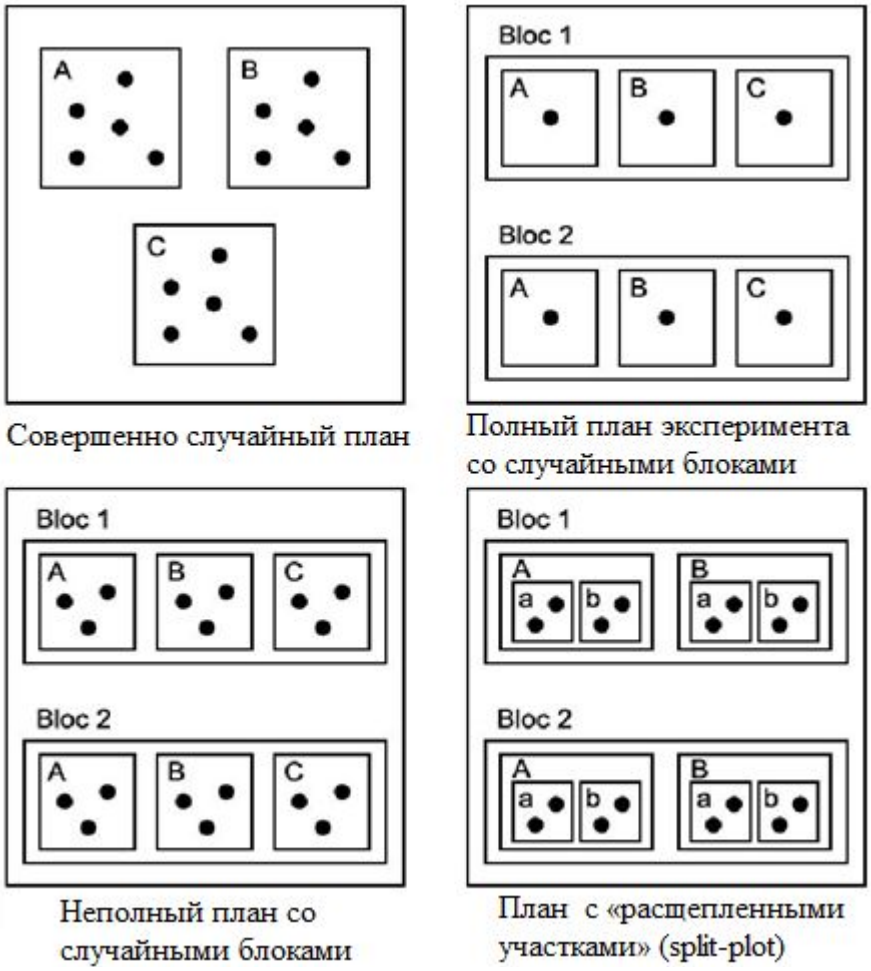
План эксперимента разрабатывается, когда исследователь может активно влиять на ход эксперимента, выполняя наблюдения в фиксированных условиях. План эксперимента включает в себя, в частности, варьируемые факторы, наблюдаемые экспериментальные единицы (индивидуумы или устройства) и количество их повторностей. Некоторая совокупность уровней нескольких факторов трактуется как ***воздействие***.

Существует множество типов планов эксперимента, основными из которых являются (см. рисунки):

- совершенно *случайный план* эксперимента: каждая единица (в статистическом смысле) назначается для воздействия случайным образом;
- *полный план* эксперимента со случайными блоками: если есть (или может быть) большая изменчивость между единицами, они объединяются в возможно более однородные группы (или блоки). В рамках этих блоков каждая единица назначается для воздействия случайным образом, так что весь набор факторов воздействует на все

единицы каждого из блока. Блоки определяются в статистическом анализе как уровни случайного фактора.

- *неполный план* эксперимента со случайными блоками: в этом случае не все воздействия имеют место в каждом из блоков.
- план эксперимента с «*расщепленными участками*» (*split-plot*): этот принцип чаще всего связан с целиком случайными блоками. Для этого в каждом из блоков создается столько суб-блоков, сколько есть уровней первого фактора. Затем каждый суб-блок делится на столько частей, сколько есть уровней второго фактора. Если в эксперименте более, чем два фактора, то ситуация становится весьма сложной.



Каждая точка представляет одну экспериментальную единицу.
Каждый квадрат представляет собой воздействие.
Уровни 1-го фактора обозначаются заглавными буквами, уровни 2-го фактора записаны строчными буквами.

Какой бы метод не использовался, его суть должна быть четко определена, поскольку она должна быть учтена в статистическом анализе.

ИСПОЛЬЗОВАНИЕ СТАТИСТИЧЕСКИХ ТЕСТОВ

14. Принципы проверки статистических гипотез и связанные с ними риски

Принцип реализации всех статистических испытаний заключается в следующем:

1. Формулируется нулевая гипотеза H_0, имеющая смысл «нет никаких изменений» (например, математические ожидания μ_A и μ_B равны) или «точки равнозначны» (например, генеральное среднее $\mu = 10$, а доля возможных альтернатив $w = 50$ %).
2. Формулируется альтернативная гипотеза H_1 таким образом, чтобы H_0 и H_1 были взаимоисключающими (например, генеральные средние значения μ_A и μ_B отличаются, а доля определенного исхода в среднем равна $w = 10\%$).
3. Вычисляется значение тестовой статистики или критерия (*VT*) согласно формулам, которые зависят от типа используемого теста.
4. Значение *VT* используется для определения *p*-значения, т.е. условной вероятности (в долях или процентах) достигнуть величины вычисленного критерия в том случае, если H_0 верна.
5. Благодаря *p*-значению, статистический вывод основан на двух предположениях:
   - если *p*-значение выше критического порога, априори заданного перед тестированием (обычно 5 %, см. п. **15**), то нет оснований отклонять H_0.
   - если *p*-значение ниже критического порога, то H_0.отвергается в пользу альтернативы H_1.

Заключение, основанное на этих двух предположениях имеет два риска:

— *риск 1-го рода*, обозначаемый α, оценивает вероятность ошибочного отклонения H_0, если она в действительности верна. Мы можем сравнить найденный риск с критическим значением, которое устанавливается, чаще всего на 5 % уровне (см. п. **15**).

— *риск 2-го рода*, обозначаемый как β, что соответствует вероятности не отвергнуть H_0, если она является на самом деле ложна. Мы не можем оценить этот риск (см. п. **17**).

	Фактическая реальность (чаще всего, неизвестна)	
Наш вывод	H_0 истинна	H_0 ложна
H_0 не отвергается	Правильный вывод	Ошибка β
H_0 отвергается	Ошибка α	Правильный вывод

Вероятность (1 - β) отклонить H_0, если она является ложной, называется мощностью теста (см. п. **17**).

Важно различать понятия «эффект статистически значим» и «эффект биологически значим». Статистически всегда можно отклонить H_0, например, из-за того, что выборка недостаточно велика (см. п. **17**).

С точки зрения статистики H_0 не является ни "истинной", или "ложной". Понятие истинности или ложности возникает только в биологической перспективе. На практике всегда наблюдаются эффекты, даже если они бывают небольшими. Вопрос лишь в том, является ли этот эффект, полученный в результате длительных исследований, достаточно большим, чтобы иметь биологическое значение. Если *да*, то H_0 на самом деле ложна, если *нет*, то H_0 является действительно истинной.

15. Критический уровень статистической значимости α

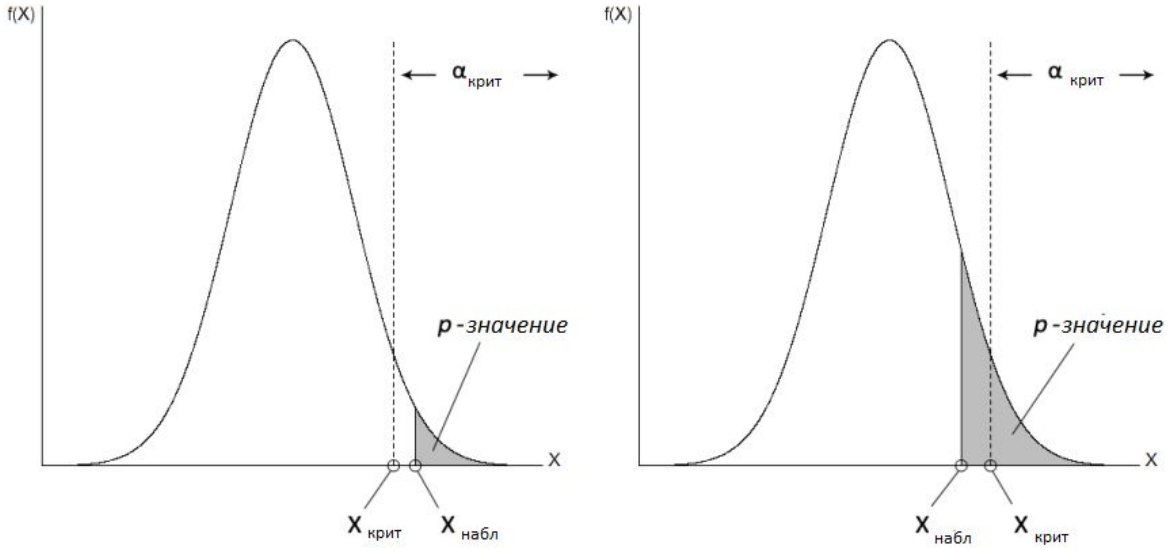
Критический уровень риска α, или порог значимости гипотезы H_0, устанавливается произвольно до выполнения любого статистического теста. Он соответствует вероятности ошибиться, отвергнув H_0, если она действительно верна (что, конечно, никто никогда не узнает).

После того, как в ходе тестирования рассчитывается *p*-значение (см. п. **14**), он сравненивается с порогом значимости α и принимают два предположения:

- если *p*-значение ниже порога α, то H_0 отклоняется. Следует считаться с тем, что при этом мы принимаем риск α ошибиться.
- если *p*-значение превышает пороговое значение α, то можно не отвергать H_0.

Получить *p*-значение выше критического уровня α можно в двух возможных случаях:

- при реальной истинности H_0 (в биологическом смысле, см. п. **14**)
- при отсутствии достаточной мощности теста (1 - β, см. п. **14**), т.е. из-за стремления не "упустить" реальной ложности H_0. Наблюдаемый эффект, следовательно, может быть биологически значимым, но размер выборки недостаточен, чтобы обосновать его с точки зрения статистики. Поэтому, если порог статистической значимости превышен, важно оценить объем выборки в зависимости от принятого порога биологической значимости (см. п. **18**).



$X_{крит}$ - значение тестовой статистики (VT), которое соответствует критическому уровню α для правой части функции плотности распределения критерия $f(X)$ (показан односторонний тест).

$X_{набл}$ - значение VT, вычисленное на основе выборочных данных.

Слева гипотеза H_0 отклоняется, справа – нет.

С точки зрения практики, мы часто используем пороговое значение α = 5 %. Это означает, что мы полностью берем на себя риск ошибиться в 5% случаев и отклонить H_0, если она в действительности верна.

16. Коррекция критического значения α (или *p-значений*)

Если выполняется некоторая последовательность статистических испытаний, каждое из которых с критическим уровнем значимости α (см. п. **15**), то общий риск отклонить H_0, если она верна, увеличивается. Действительно, чем больше проводится испытаний, тем больше вероятность извлечения мало репрезентативной выборки, искажающей общий результат и дающей *p*-значение ниже порога α.

Поэтому необходимо скорректировать критический уровень α для каждого частного теста (или их вычисленные *p*-значения, что то же самое), чтобы получить общий желаемый риск равный α.

Эту коррекцию необходимо осуществлять:

- когда необходимо принять однозначное решение по совокупности последовательных тестов, и, по крайней мере, хотя бы один из них отклоняет H_0;
- когда проводится серия тестов парных сравнений, либо непосредственно, либо *post hoc* после проведения общего анализа (ANOVA, тест χ^2 на однородность и т.д.).

Существует несколько методов коррекции, в том числе, три следующих:

1. *Метод Бонферрони*.

Если выполняется k тестов, этот метод состоит просто в разделении общего критического значения α на k, т.е. для каждого частного теста порог значимости устанавливается как α/k.

Это также означает умножение каждого *p*-значения на k, не меняя порога α.

2. *Последовательный метод Холма*.

Процедура осуществляется в несколько этапов :

1. Сортировка *p*-значений всех проведенных испытаний в порядке возрастания ($p_1 <...< p_k$), k – количество проведенных испытаний.

2. Отклонение H_0 для тестов, в которых *p*-значений соответствует условию $p_i \leq \alpha/(k - i + 1)$, где i – порядковый номер *p*-значения в отсортированном ряду.

3. *Алгоритм Бенджамини и Хохберга оценки доли ложных отклонений* (FDR, False Discovery Rate)

Процедура осуществляется в несколько этапов :

1. Сортировка *p*-значений всех проведенных испытаний в порядке возрастания ($p_1 <...< p_k$), k – количество проведенных испытаний.

2. Отклонение H_0 для тестов, в которых *p*-значений соответствует условию: $p_i \leq i \cdot \alpha/k$, где i – порядковый номер *p*-значения в отсортированном ряду.

Наиболее строгая (и наименее мощная) техника соответствует методу Бонферрони, менее строгая (и более мощная) соответствует FDR, которая часто применяется по умолчанию. В любом случае метод коррекции критического порога значимости должен быть выбран до проведения тестирования.

В среде R, если имеется вектор `p`, содержащий нескорректированные *p*-значения, то можно использовать функцию `p.adjust()`, чтобы получить вектор с исправленными *p*-значениями (в том же порядке):

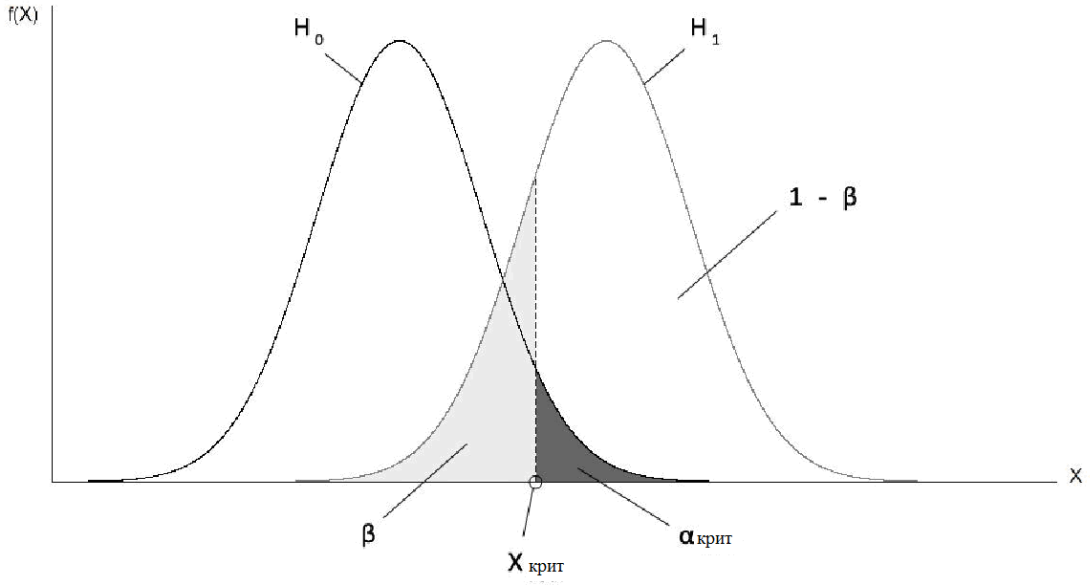
- для коррекции Бонферрони
  `p.adjust(p, method="bonferroni")`
- для коррекции Холма
  `p.adjust(p, method="holm")`
- для коррекции Бенджамини-Хохберга (FDR)
  `p.adjust(p, method="BH")` или `p.adjust(p, method="fdr")`

17. Риск β ошибки II рода и мощность критерия

Вероятность β определяет риск ошибочно не отклонить нулевую гипотезу H_0, если она фактически ложна (см. п. **14**). В отличие от статистической значимости α, величина риска β не может быть оценена. Действительно, если α зависит от распределения тестовой статистики (*VT*) при справедливости H_0 (см. п. **15**), то значение β зависит от распределения этого же критерия при справедливости H_1. Разумеется, это распределение неизвестно, так как гипотеза H_1 объединяет бесконечное множество распределений. Например, если гипотеза H_1 утверждает, что два средних $\mu_A \neq \mu_B$, то эти различия могут быть объяснены бесконечным множеством способов.

Мощность теста – это гипотетическая вероятность отвергнуть H_0, если она действительно ложна (т.е. сделать правильный выбор). Она равна (1 - β), и, следовательно, также является переменной, зависящей от распределения тестовой статистики *VT* при справедливости H_1.

Риск β и мощность теста зависят от установленного критического уровня значимости α.



$X_{крит}$ - значение тестовой статистики (VT), которое соответствует критическому уровню α для правой части функции плотности распределения критерия $f(X)$ (показан односторонний тест).

Мощность теста (1 - β) увеличивается :

- если увеличивается критический уровень значимости α;
- когда увеличивается объем выборки (это уменьшает оценку дисперсии распределения тестовой статистики);
- когда увеличивается разница между тестируемыми значениями (средними, долями и т.д.) .

18. Определение необходимого объема выборкт

Существует взаимосвязь между критическим уровнем значимости α, используемом в статистическом тесте (см. пп. **14** и **15**), мощностью (1 - β) этого теста (см. п. **17**), разностью между выборочными параметрами для измеряемой величины и объемом выборки. Для того, чтобы определить объем выборки и спланировать эксперимент, необходимо тем или иным образом установить три остальных переменные. Это означает два важных выбора:

- прежде всего, основываясь на цели исследования, используемых статистических тестах и степени ответственности экспериментатора за полученные выводы, необходимо задасться уровнем значимости и мощностью. Из практических соображений считается, что, если целью исследователя является отклонить H_0, то мощность теста должна быть не менее 80 %; но если цель – не отклонить H_0, то мощность должна быть не менее 95 %.
- необходимо получить представление о естественной изменчивости измеряемой переменной, и/или минимальной разнице обнаружения эффекта (см. оценка порогов биологической и статистической значимости, п. **14**). Это происходит путем изучения литературы, с помощью консультаций специалистов, либо при проведении сокращенного опыта или предварительного отбора образцов.

В этом разделе мы будем использовать для решения поставленной задачи пакет `pwr` среды **R**. Семейство функций `power()` позволяет для некоторых параметрических тестов рассчитать необходимый объем выборки `n`, если известны все остальные параметры. Даже если мы заранее знаем, что не будем использовать параметрические тесты, мы можем оценить эквивалентную мощность непараметрического теста, которая обычно составляет около 95 % от мощности аналогичного параметрического теста.

Все описанные функции основаны на одном принципе: параметр `n` должен иметь значение `NULL`, в то время как все другие должны быть заданы конкретными значениями. Все функции также считают, что численность различных групп сопоставима (обычно говорят о сбалансированой численности). Настоятельно рекомендуется всегда предусматривать условие эквивалентной численности групп при планировании выборки или опыта, поскольку тем самым (1) облегчается анализ и (2) создается возможность использовать большинство непараметрических тестов.

Сравнение выборочного среднего значения с теоретическим математическим ожиданием (см. п. 72) или сравнение двух групповых средних (см. п. 73) – тест Стьюдента

`power.t.test(n, delta, sd, sig.level, power, type)`,

где:

здесь и далее `n` – оцениваемый объем выборок (который одинаков для обеих групп, в случае сравнения двух средних);

`delta` – минимальная обнаруживаемая разность между двумя средними, или между выборочным средним и теоретической величиной;

`sd` – стандартное отклонение (одинаковое для обеих групп, в случае сравнения двух средних);

`sig.level` – уровень значимости α (обычно 0.05)

`power` – минимальная мощность теста (от 0.8 до 0.95 в зависимости от цели тестирования);

`type` – тип теста (`"one.sample"` для сравнения выборочного среднего с теоретическим значением, `"two.sample"` для сравнения двух средних, `"paired"` для сравнения двух средних в серии сопряженных наблюдений).

Сравнение средних для более, чем двух групп, в зависимости от уровней фактора – дисперсионный анализ (см. п. 76)

```
power.anova.test(groups, n, between.var, within.var,
                 sig.level, power),
```

где:

`groups` – количество сравниваемых групп;

`between.var` – минимальная обнаруживаемая сумма квадратов отклонений между группами;

`within.var` – сумма квадратов внутригрупповых отклонений (одинаковая для всех групп).

На практике очень трудно оценить *a priori* суммы квадратов меж- и внутригрупповых отклонений. Если выполняются условия применения дисперсионного анализа, то можно в этом случае прибегнуть к функции `power.t.test ()`, которая дает приемлемые результаты.

Сравнение двух долей – тест χ^2 на гомогенность (см. п. 58)

```
power.prop.test(n, p1, p2, sig.level, power)
```

где `p1, p2` – пропорции, обнаруживаемые как существенно различные.

Статистическая значимость коэффициента линейной корреляции Пирсона (см. п. 84)

```
pwr.r.test(n,r,sig.level,power)
```
[1]

`r` : минимальный коэффициент корреляции, существенно отличающийся от 0.

ОСНОВНЫЕ ЗАКОНЫ РАСПРЕДЕЛЕНИЯ СЛУЧАЙНЫХ ВЕЛИЧИН

19. Законы распределения дискретных величин (резюме)

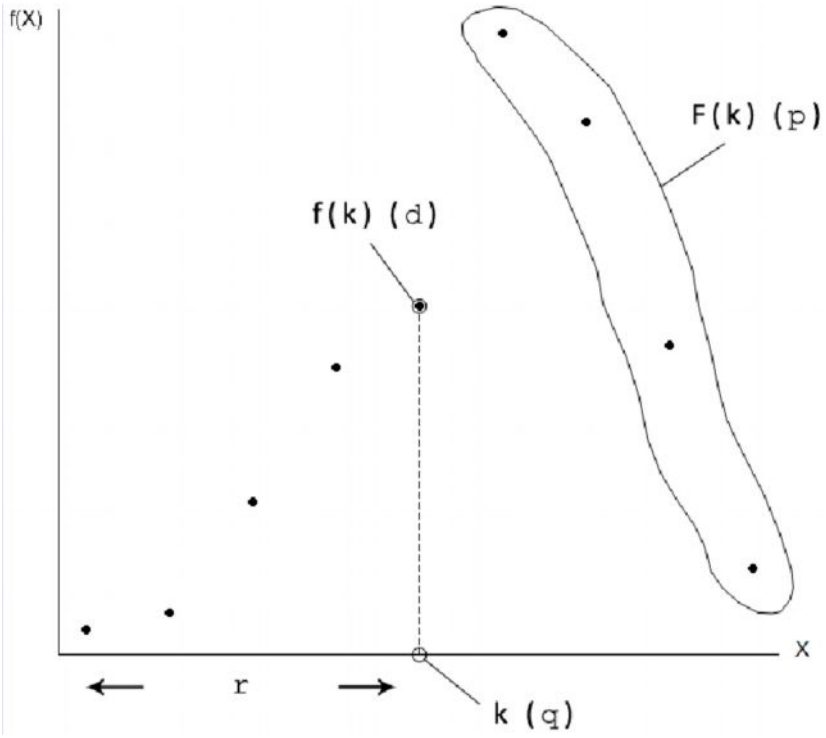
Эти законы применяются в отношении количественных переменных дискретного типа (см. п. **11**).

С ними связаны следующие понятия :

- k – любое возможное значение, которое принимает дискретная переменная X. Также называется "квантиль".
- $f(k)$ – частота, или вероятность, связанная со значением дискретной переменной k. Также называется "функцией плотности распределения вероятностей X". Изменяется в диапазоне между 0 и 1.
- $F(k)$ – сумма вероятностей $f(k)$, расположенных справа или слева от k, в зависимости от ситуации. Также называется "функцией распределения X". Обозначается как $F(k)_{справа} = P(X > k)$ и $F(k)_{слева} = P(X < k)$. Изменяется в диапазоне между 0 и 1.

Семейство функций **R** , выполняющих моделирование распределений:

- `dY()` – дает значение вероятности $f(k)$ для распределения типа Y.
- `pY()` – дает значение функции распределения $F(k)$ для распределения типа Y. **R** по умолчанию считает накопление плотности вероятности слева, нужно указать `lower.tail = FALSE` для правостороннего распределения.
- `qY()` – дает значение k переменной X, соответствующее значение $F(k)$ для распределения типа Y. **R** по умолчанию рассматривает распределение слева от k, нужно указать `lower.tail = FALSE` для правостороннего распределения.
- `rY()` – дает серию значений случайной переменной X для распределения типа Y.



20. Биномиальное распределение

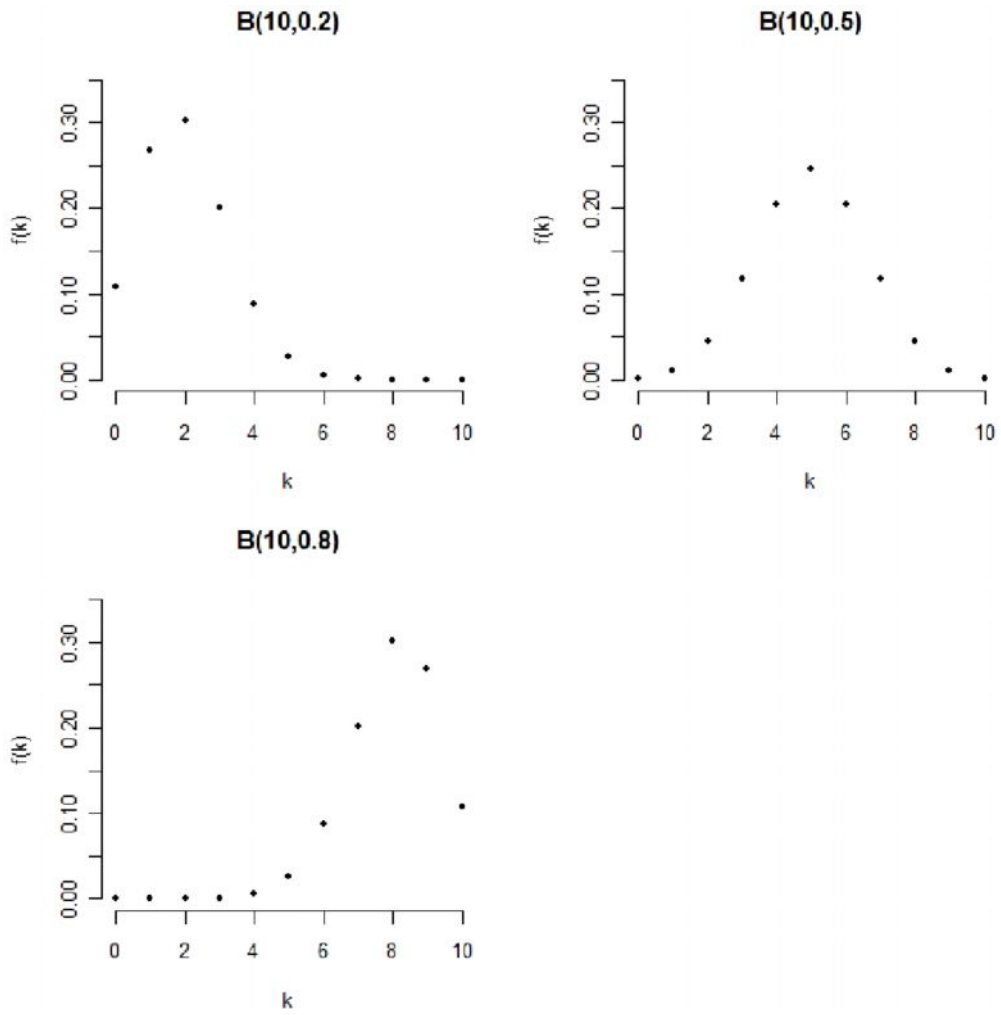
Случайную величину, распределенную по биномиальному закону, интерпретируют как число "успехов" k в серии из n независимых случайных испытаний Бернулли с двумя взаимно исключающими результатами ("успех"/"неудача" или A/B), причем вероятность p любого исхода постоянна в течение всего опыта.

Запись : $B(n, p)$,
где n – количество испытаний, p – вероятность, связанная с результатом A.

В **R** используются функции (см. п. **19** для объяснения):

```
dbinom(k,n,p)          pbinom(k,n,p)
qbinom(F(k),n,p)       rbinom(x,n,p)
```

где `x` – количество случайных значений, извлекаемых из распределения.



21. Распределение Пуассона

Распределение Пуассона - это частный случай биномиального распределения (см. п. **20**), когда p стремится к 0, а n к бесконечности. Закон распределения Пуассона часто называют «законом редких событий». Аппроксимация биномиального распределения распределением Пуассона возможна, когда $p < 0{,}1$ и $n > 30$.

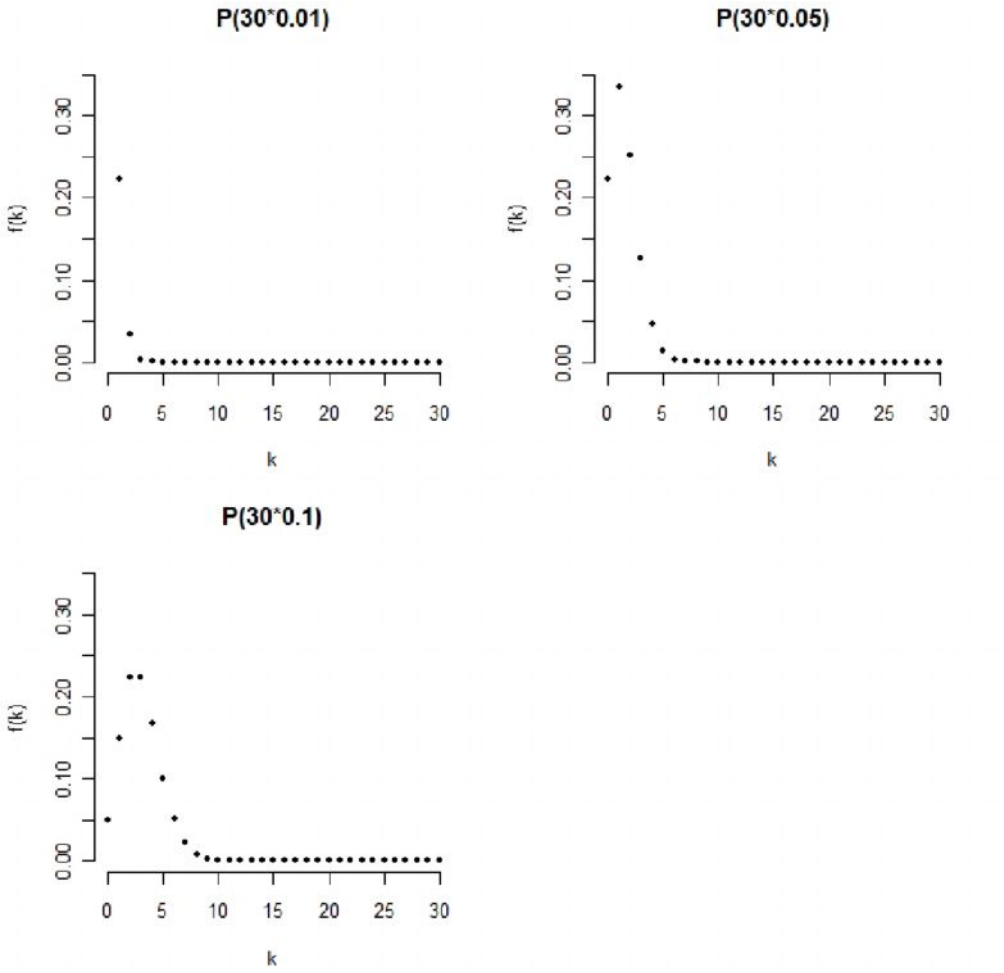
Математическое ожидание случайной величины, распределенной по закону Пуассона, равно ее дисперсии и соответствует np (или λ)

Запись : $P(np)$ или $P(\lambda)$:
где n – количество испытаний, p – вероятность, связанная с появлением редкого события.

В **R** используются функции (см. п. **19** для объяснения) :

```
dpois(k,n*p)        или dpois(k,lambda)
ppois(k,n*p)        или ppois(k,lambda)
qpois(F(k),n*p)     или qpois(F(k),lambda)
rpois(x,n*p)        или rpois(x,lambda)
```

где `x` – количество случайных значений, извлекаемых из распределения.



22. Отрицательное биномиальное распределение

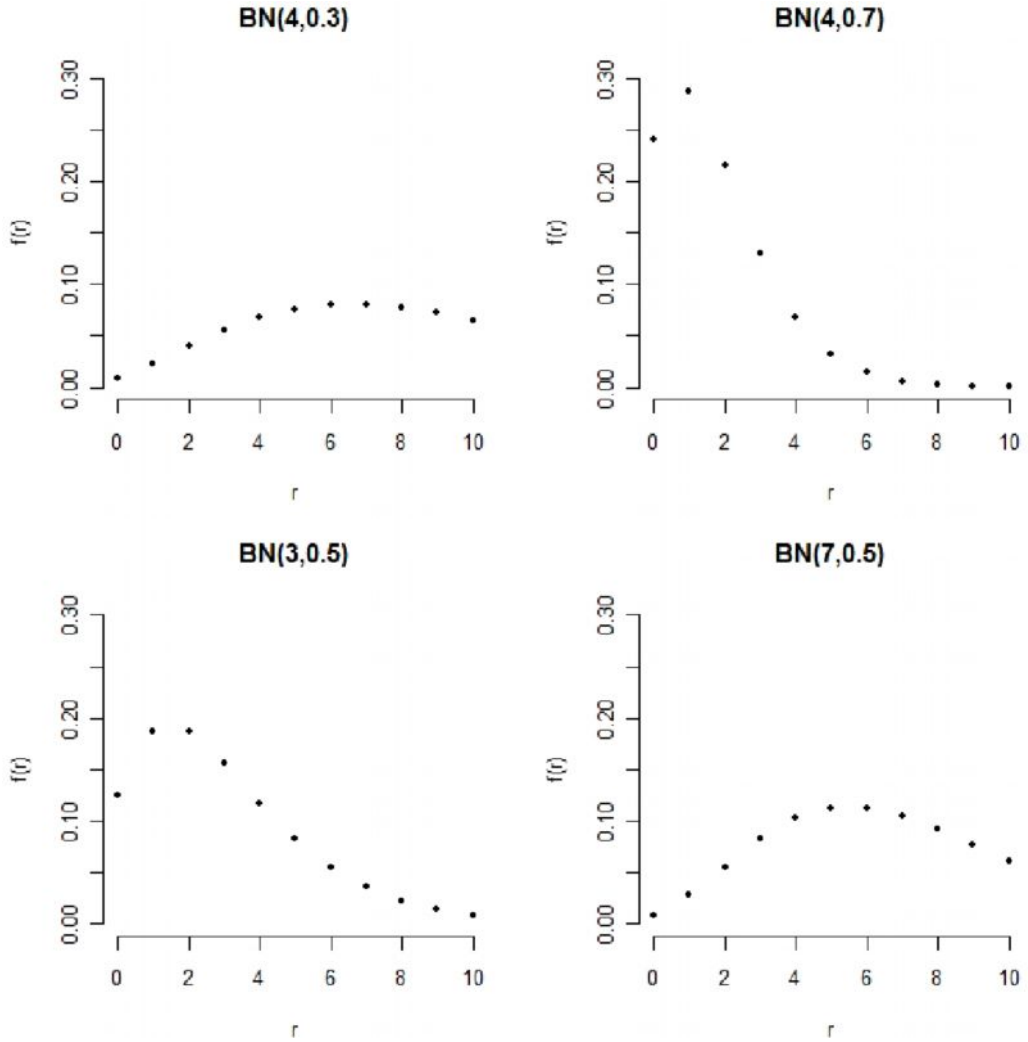
Отрицательное биномиальное распределение соответствует тем же предпосылкам, что и биномиальное распределение (см. п. **20**), но испытания продолжаются до появления k результатов исхода А. Распределение дискретной случайной величины X соответствует вероятности получить в этих условиях r исходов В.

Запись : $BN(k, p)$
где: k – число исходов А , p – вероятность, связанная с результатом А.

В R используются функции (см. п. **19** для объяснения) :

```
dnbinom(r,k,p)          pnbinom(r,k,p)
qnbinom(F(r),k,p)       rnbinom(x,k,p)
```

где `x` – количество случайных значений, извлекаемых из распределения.



23. Законы распределения непрерывных величин (резюме)

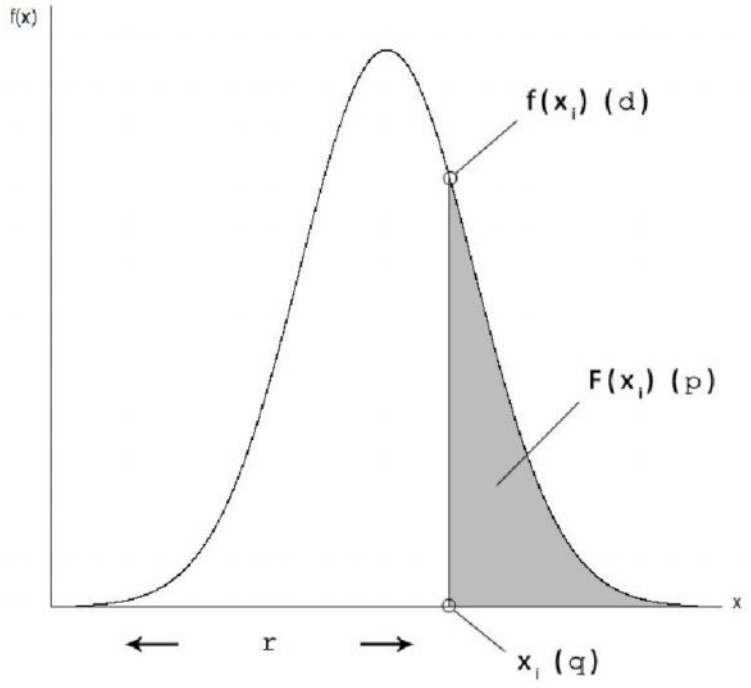
Эти законы применяются в отношении количественных переменных непрерывного типа (см. п. **11**).

С ними связаны следующие понятия :

- x_i – любое возможное значение, которое принимает непрерывная переменная X. Также называется «квантиль».
- $f(x_i)$ – распределение вероятности, связанной со значением непрерывной переменной x_i. Также называется «функцией плотности распределения вероятностей X». Изменяется в диапазоне между 0 и 1.
- $F(x_i)$ – площадь под кривой, расположенной справа или слева от x_i, в зависимости от ситуации. Также называется «функцией распределения X». Обозначается как $F(x_i)_{\text{справа}} = P(X > x_i)$ и $F(x_i)_{\text{слева}} = P(X < x_i)$. Изменяется в диапазоне между 0 и 1.

Семейство функций **R** , выполняющих моделирование распределений:

- `dY()` – дает значение вероятности $f(x_i)$ для распределения типа Y.
- `pY()` – дает значение функции распределения $F(x_i)$ для распределения типа Y. **R** по умолчанию считает накопление плотности вероятности слева, нужно указать `lower.tail = FALSE` для правостороннего распределения.
- `qY()` – дает значение x_i переменной X, соответствующее значение $F(x_i)$ для распределения типа Y. **R** по умолчанию рассматривает распределение слева от x_i, нужно указать `lower.tail = FALSE` для правостороннего распределения.
- `rY()` – дает серию значений случайной переменной X для распределения типа Y.



24. Нормальное распределение

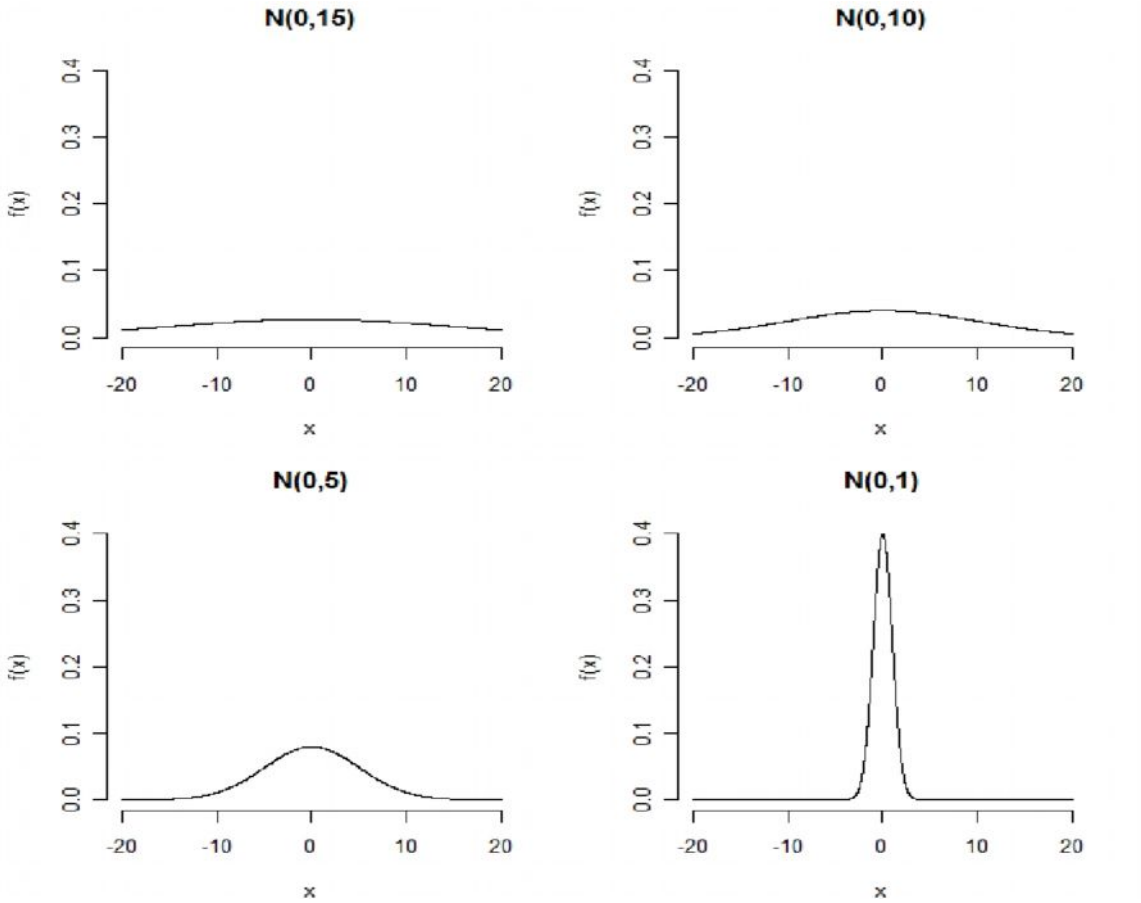
Нормальный закон распределения является основным фундаментом статистики, поскольку служит для моделирования случайного шума относительно среднего. Он осуществляет теоретическую поддержку многих моделей и тестов для оценки *p*-значений, выступая в качестве предварительного условия их использования.

Запись: $N(\mu, \sigma)$,
где μ – среднее значение переменной X, σ – стандартное отклонение переменной X. В случае стандартизованной случайной величины нормальное распределение имеет вид: $N(0, 1)$.

В **R** используются функции (см. п. **23** для объяснения) :

```
dnorm(xi,mu,sigma)          pnorm(xi,mu,sigma)
qnorm(F(xi),mu,sigma)       rnorm(z,mu,sigma)
```

где `z` – количество случайных значений, извлекаемых из распределения.



25. Экспоненциальное распределение

По экспоненциальному закону часто распределены события, вероятность появления которых уменьшается с течением времени. Поэтому он часто используется для моделирования процессов выживания.

Запись : *exp*(λ),

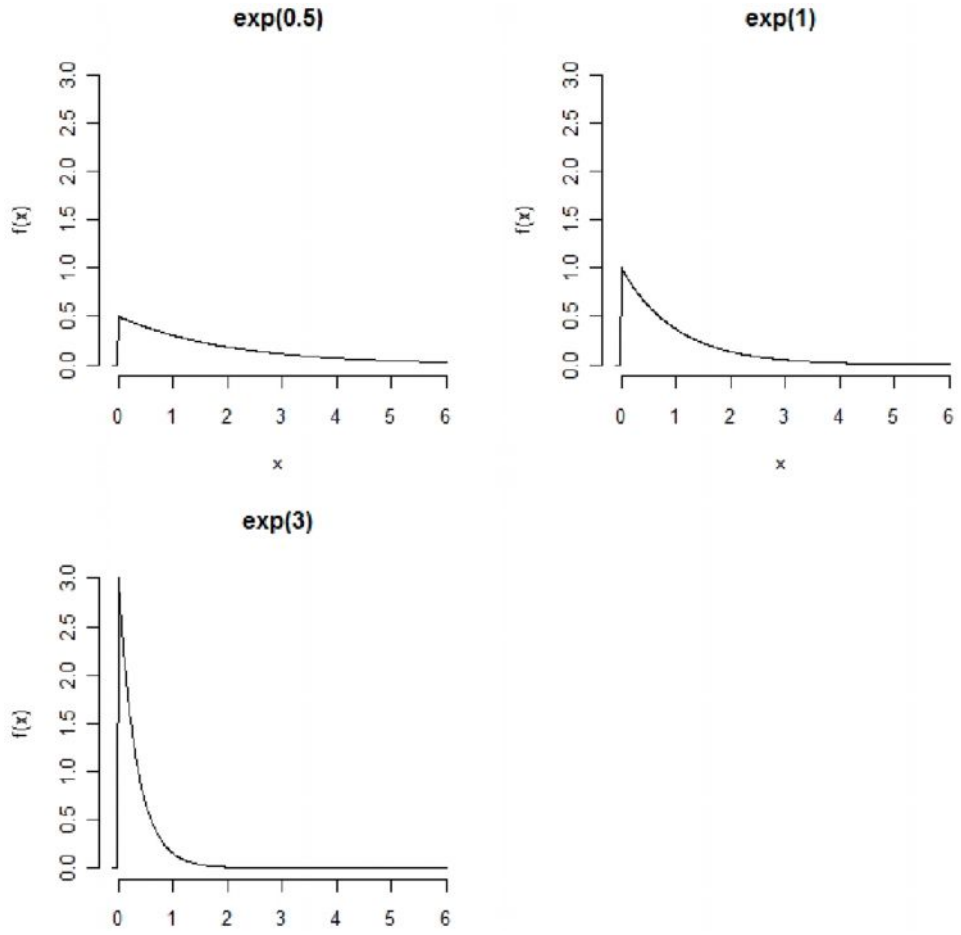
где λ – параметр распределения ($0 < \lambda < +\infty$).

В **R** используются функции (см. п. **23** для объяснения) :

`dexp(xi,lambda)` `pexp(xi,lambda)`

`qexp(F(xi),lambda)` `rexp(z,lambda)`

где `z` – количество случайных значений, извлекаемых из распределения.



26. Распределение χ^2

Распределение χ^2 используется во многих статистических тестах (не только в качестве "теста χ^2") как моделирующее распределение для расчета *p*-значений.

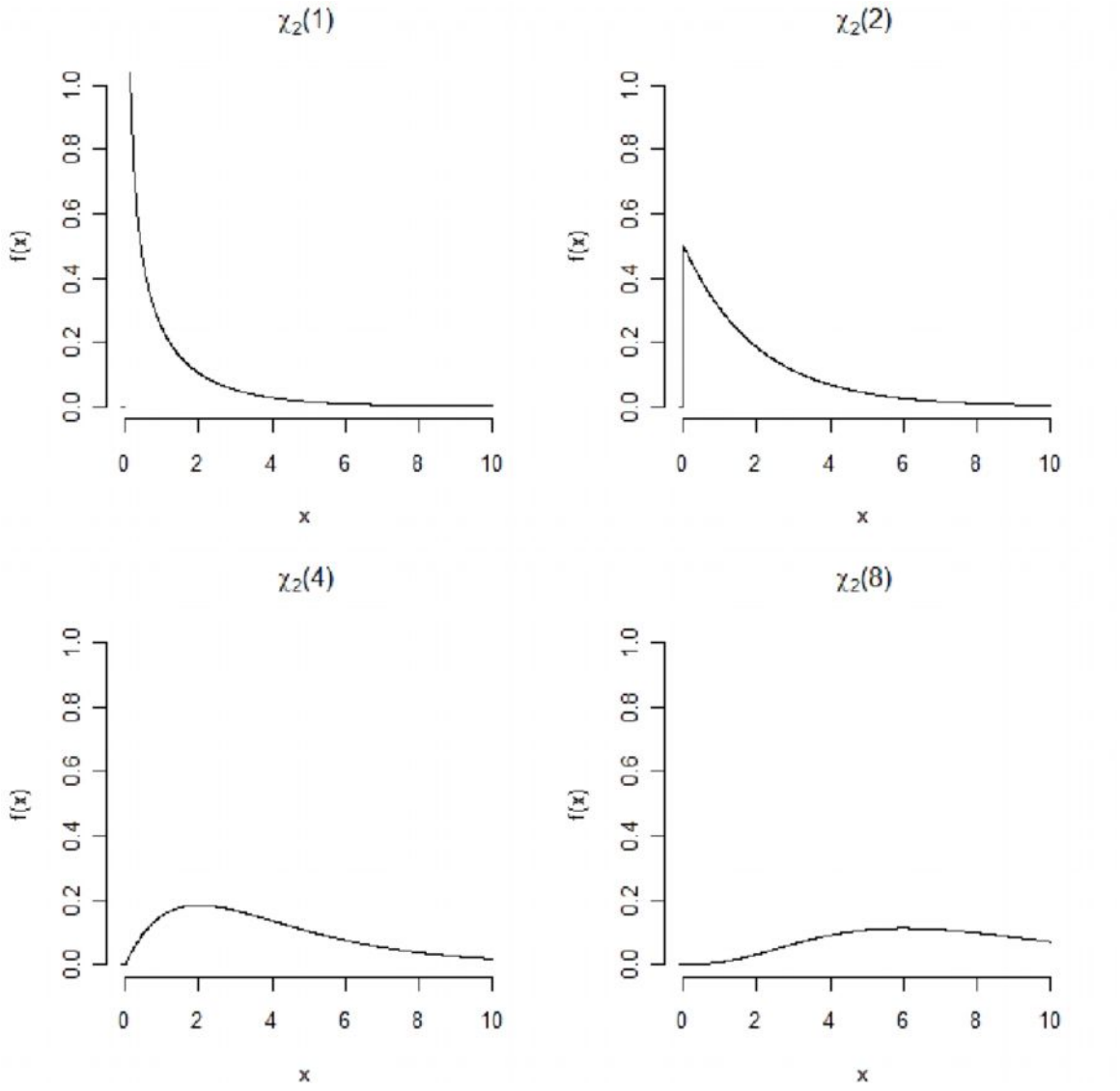
Запись : $\chi 2(\nu)$

где ν – число степеней свободы (`df`), т. е. независимых параметров, участвующих в распределении, $(0 < \nu < +\infty)$.

В **R** используются функции (см. п. **23** для объяснения) :

```
dchisq(xi, df)          pchisq(xi, df)
qchisq(F(xi), df)       rchisq(z, df)
```

где `z` – количество случайных значений, извлекаемых из распределения.



27. Распределение Фишера-Снедекора

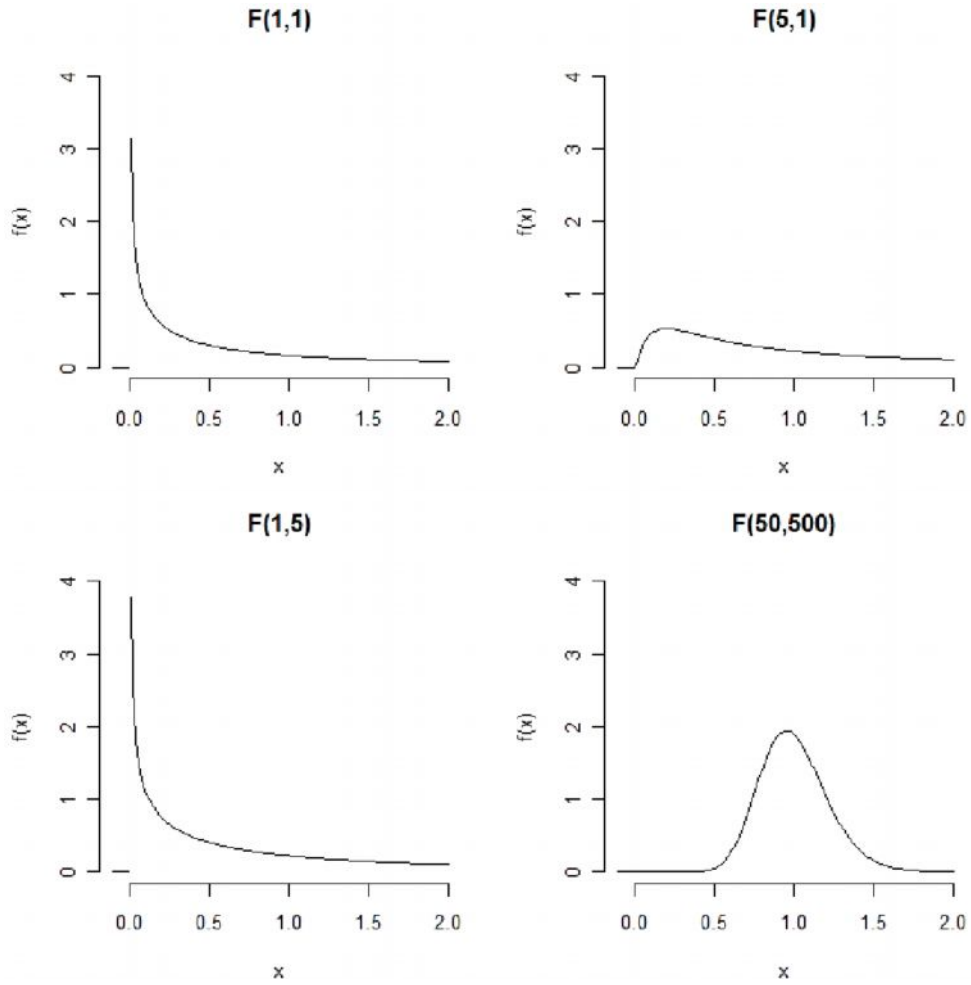
Распределение Фишера-Снедекора используется во многих статистических тестах (не только в дисперсионном анализе ANOVA) как моделирующее распределение для расчета *p*-значений.

Запись : $F(\nu_1, \nu_2)$
где ν_1 и ν_2 – число степеней свободы (`df`), $(0 < \nu_1 < +\infty)$, $(0 < \nu_2 < +\infty)$

В **R** используются функции (см. п. **23** для объяснения) :

```
df(xi, df1, df2)        pf(xi, df1, df2)
qf(F(xi), df1, df2)     rf(z, df1, df2)
```

где `z` – количество случайных значений, извлекаемых из распределения.



28. Распределение Стьюдента

Распределение Стьюдента используется во многих статистических тестах (не только в качестве "*t*-теста") как моделирующее распределение для расчета *p*-значений.

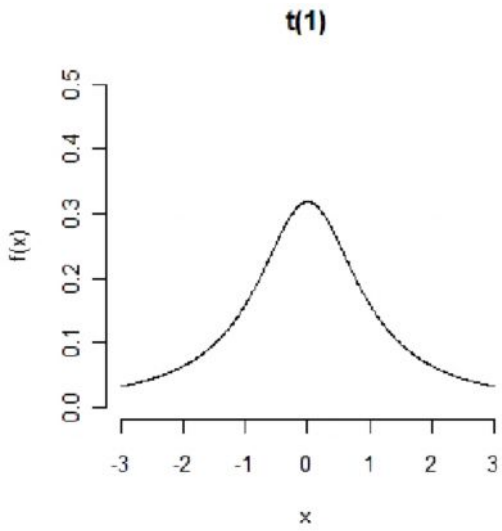
Запись : $t(\nu)$

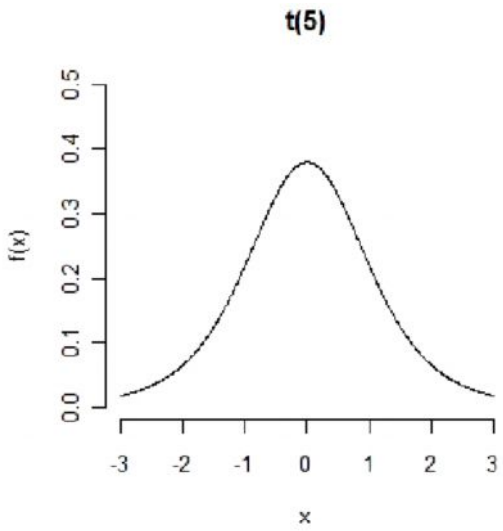
где ν – число степеней свободы (`df`), т. е. независимых параметров, участвующих в распределении, $(0 < \nu < +\infty)$.

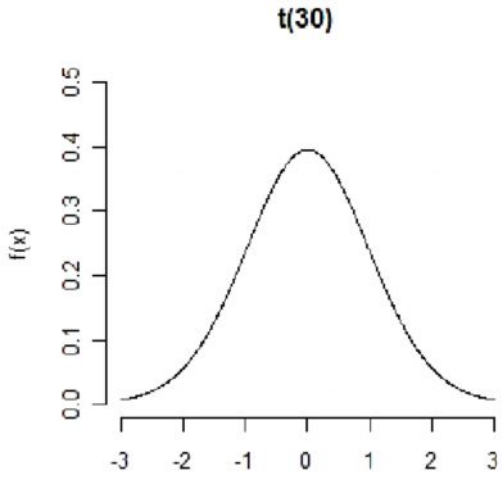
В **R** используются функции (см. п. **23** для объяснения) :

```
dt(xi, df)          pt(xi, df)
qt(F(xi), df)       rt(z, df)
```

где `z` – количество случайных значений, извлекаемых из распределения.







3. Анализ результатов исследования

Поскольку изложение любого материала нуждается в некоторой классификации (возможно, субъективной), выделим в рамках общего статистического анализа три основных раздела: одномерный, двумерный и многомерный. На практике их довольно просто различать:

- если мы имеем одну случайную переменную и стремимся узнать ее свойства (в том числе, не определяется ли изменчивость ее значений воздействием одной или нескольких внешних переменных), то мы имеем **одномерный статистический анализ**;
- если исследуется связь между двумя переменными одно и того же типа, количественных или качественных, причем не всегда можно указать, какая из них является независимой, а какая объясняемой (т.е. речь идет о взаимозависимых переменных), то мы имеем **двумерный статистический анализ**;
- если имеется несколько взаимозависимых переменных, и надо объяснить, в каких сочетаниях проявляются связи между ними, то мы имеем **многомерный статистический анализ**.

Одномерный и многомерный (т. е. более общий) подходы могут использоваться совместно и дополнять друг друга, когда проводится анализ нескольких переменных. Например, многомерный подход становится особенно интересным, когда переменные коррелируют между собой, поскольку он учитывает эффекты множественной корреляции и позволяет выделить явления, которые не видны в масштабе отдельных переменных. Не существует критического порога коррелированности, чтобы оценить, когда становится оправданным применение многомерного подхода; однако ничего не стоит попробовать и посмотреть, не появятся ли при этом интересные феномены!

Примечание: часто несколько тестов могут быть использованы для ответа на один и тот же вопрос. Условия их применения должны учитывать некоторые предпочтения, основанные на относительной строгости и мощности отдельных тестов (более строгий тест является, как правило, более мощным; см. п. **17**). Когда доступны несколько тестов, нами представлены "деревья решений", чтобы помочь выбрать наилучший тест. Они опираются как на проверку теоретических условий применимости каждого теста, так и на их "качество" (в том числе в плане мощности).

ОДНОМЕРНЫЙ СТАТИСТИЧЕСКИЙ АНАЛИЗ

ГРАФИКИ

30. Графики рассеивания

Этот тип графика представляет все элементы данных вектора, матрицы или таблицы. Он позволяет получить представление о вариабельности данных и выделить нехарактерные значения (выбросы). Для создания графика используется функция `stripchart()`:

- для представления одного вектора: `stripchart(вектор)`;
- для отображения нескольких векторов:
  `stripchart(list(вектор1, вектор2,...))`.

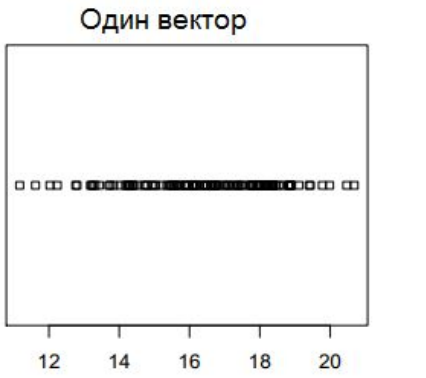
Для того, чтобы присвоить имена векторам на диаграмме, нужно добавить аргумент `group.names=c("Наименование1", "Наименование2",...)`.

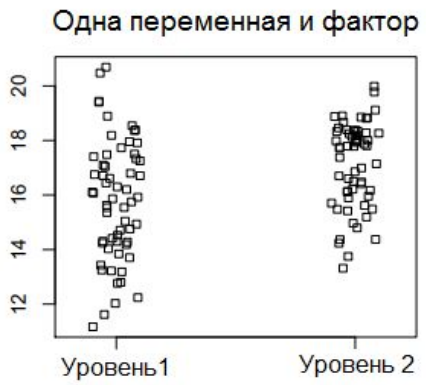
Для представления зависимости данных от одного действующего фактора используют: `stripchart(отклик ~ фактор)`, где все объекты являются векторами, содержащие значения каждого элемента (в одном и том же порядке). Символ `~` означает "*установить соответствие*".

Для представления данных по вертикали, добавляют аргумент `vertical=TRUE`.

Чтобы одинаковые значения не накладывались друг на друга, добавляют аргумент `method="jitter"` (по умолчанию `method="overplot"`).

Чтобы добавить заголовок к графику, используют аргумент `main="Заголовок"`. Чтобы изменить заголовок горизонтальной оси, используют аргумент `xlab="Легенда"`, а для вертикальной оси – `ylab="Легенда"`. Для того, чтобы познакомиться с остальными графическими параметрами, введите команду `? par`.





31. Гистограммы

Этот тип диаграммы делит данные, объединенные в вектор, на классы по частоте встречаемости значений переменной или плотности вероятности, и представляет эти классы столбчатой диаграммой. Это позволяет получить общее представление о распределении данных. Для создания графика используется функция `hist()`.

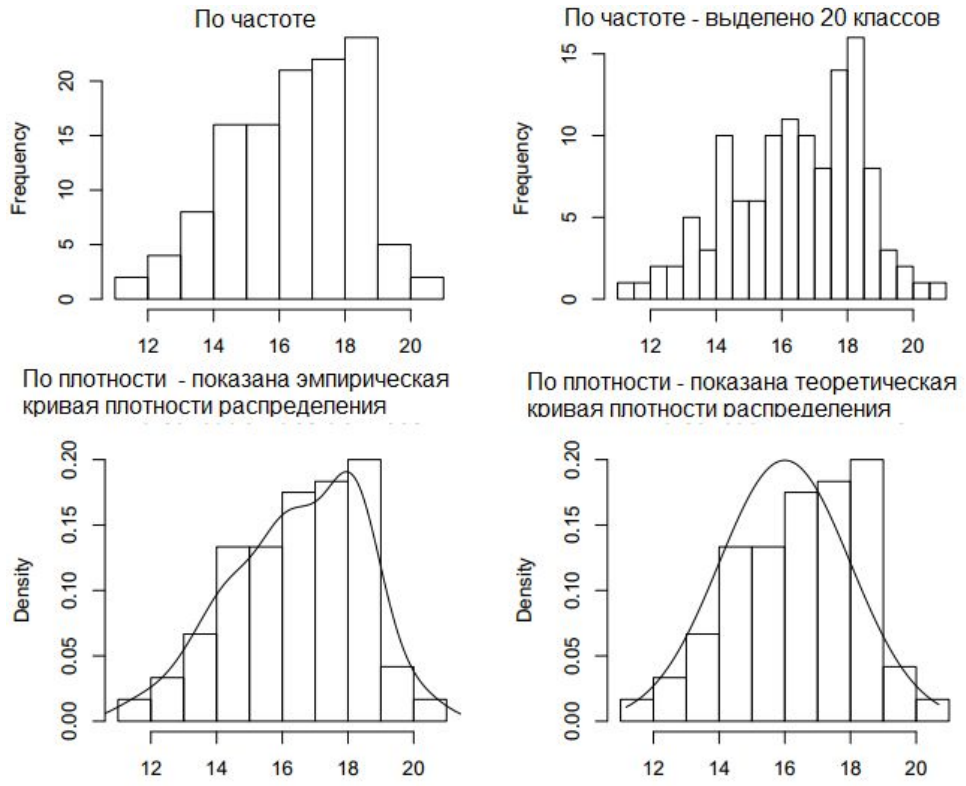
Для представления классов по частоте: `hist(вектор)` (т.е. параметр `freq=TRUE` принимается по умолчанию).

Для представления классов по плотности : `hist(вектор,freq=FALSE)`. Чтобы добавить поверх такой диаграммы кривую ядерной плотности, подается вторая команда: `lines(density(вектор))`. Чтобы добавить кривую теоретического распределения, подается команда:
`lines(seq2(вектор)1, dloi(seq2(вектор)1, par))`, где `seq2()*` – функция, определенная в пакете `RVAideMemoire`[1], `loi` – выбранной закон распределения вероятности, и через запятую – параметры соответствующей функции (см. п. **19 - 28**), разделенные запятыми.

Для изменения количества классов, добавьте аргумент `breaks = n`, где `n` – число необходимых разбиений (т. е. выделяется `n` + 1 классов).

Можно также использовать следующие дополнительные аргументы:

- добавить заголовок к графику: `main="Заголовок"`;
- добавить заголовок горизонтальной оси: `xlab="Легенда"`;
- добавить заголовок вертикальной оси: `ylab="Легенда"`.



Для получения серии гистограмм переменной в зависимости от уровня фактора в пакете `RVAideMemoire` предусмотрена функция: `byf.hist(отклик ~ фактор)`, где `отклик` и `фактор` являются векторами, содержащими значения каждого элемента выборки (в одном и том же порядке). Символ `~` означает "*установить соответствие*".

[1] Символом * мы в дальнейшем будем отмечать функции из пакета `RVAideMemoire`

32. Ящики с усами

Этот тип представляет собой упрощенный график разброса данных, содержащихся в векторе. Он позволяет получить общее представление о распределении и изменчивости данных, а также выявить нехарактерные значения (выбросы). Для создания графика используется функция `boxplot()`.

Для отображения одного вектора: `boxplot(вектор)`. Жирная линия представляет медиану, "ящик" формируется из значений 1-го и 3-го квартилей, а "усы" имеют максимальный размах, в 1.5 раза превышающий разность между 3-м и 1-м квартилями. Значения, выходящие за усы, представлены по отдельности.

Для представления нескольких векторов используют команду: `boxplot(list(вектор1, вектор2,...))`. Для того, чтобы дать имена ящикам, необходимо добавить аргумент `names=c("Name1", "Name2",...)`.

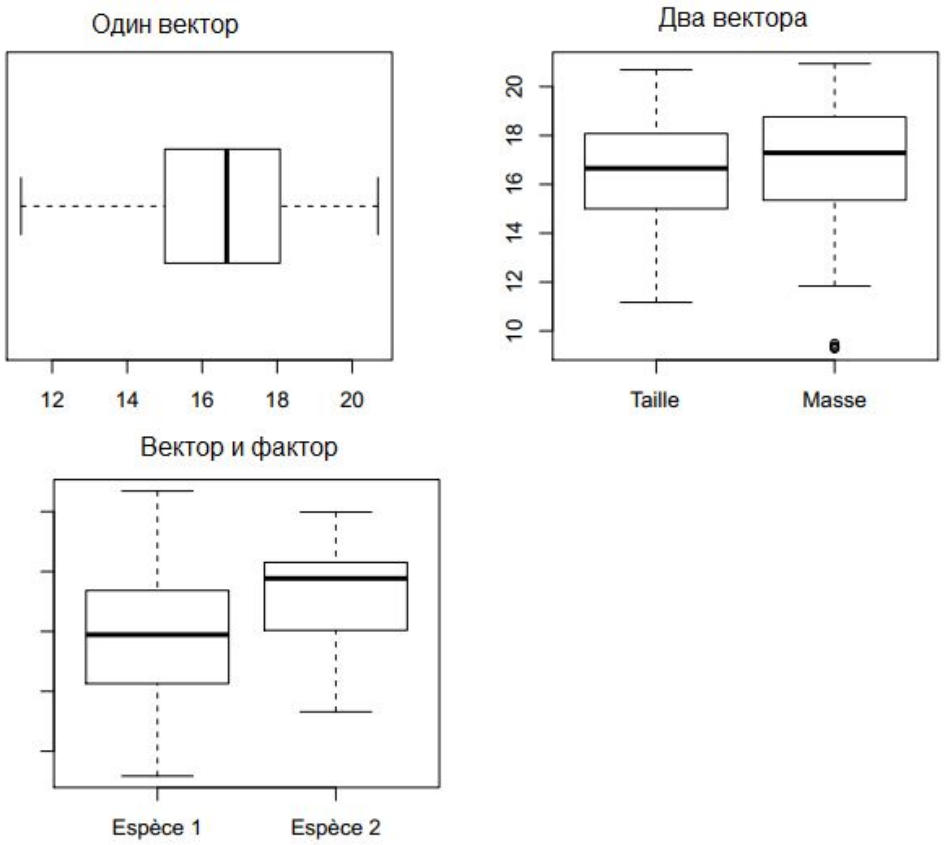
Для представления серии ящиков для разных уровней фактора: `boxplot(отклик ~ фактор)`, где `отклик` и `фактор` являются векторами, содержащими значения каждого элемента выборки (в одном и том же порядке). Символ `~` означает "*установить соответствие*".

Для представления ящиков по горизонтали необходимо добавить аргумент `horizontal=TRUE`.

Чтобы добавить заголовок к графику, используют аргумент `main="Заголовок".`

Чтобы изменить подписи горизонтальной оси (если ящики ориентированы горизонтально), нужно использовать аргумент `xlab="Легенда"`.

Чтобы изменить заголовок вертикальной оси (если ящики ориентированы вертикально), нужно использовать аргумент `ylab="Легенда"`.



33. Графики типа "фасоль"

Этот тип графика сочетает в себе преимущества диаграмм рассеивания (т.е. визуализацию каждого значения данных, см. п. **30**), гистограммы (т.е. показывает характер распределения выборки, см. п. **31**) и ящика с усами (т.е. дает общее представление, какова изменчивость по уровням фактора, см. п. **32**).

Для создания графика используется функция `beanplot(вектор)` из пакета `beanplot`. Каждое индивидуальное выборочное значение представлено небольшой горизонтальной линией (для близких значений ширина полосы соответственно увеличивается), кривая эмпирического распределения показана вертикально и зеркально отражена, среднее значение для групп показано длинной жирной полосой, а общее среднее – прямой пунктирной линией.

Для представления нескольких векторов используется команда: `beanplot(list(вектор1, вектор2,...))`. Для того, чтобы дать имена фигурам, необходимо добавить аргумент `names=c("Name1", "Name2",...)`.

Для представления серии фигур для разных уровней фактора: `beanplot (отклик ~ фактор)`, где `отклик` и `фактор` являются векторами, содержащими значения каждого элемента выборки (в одном и том же порядке). Символ `~` означает "*установить соответствие*".

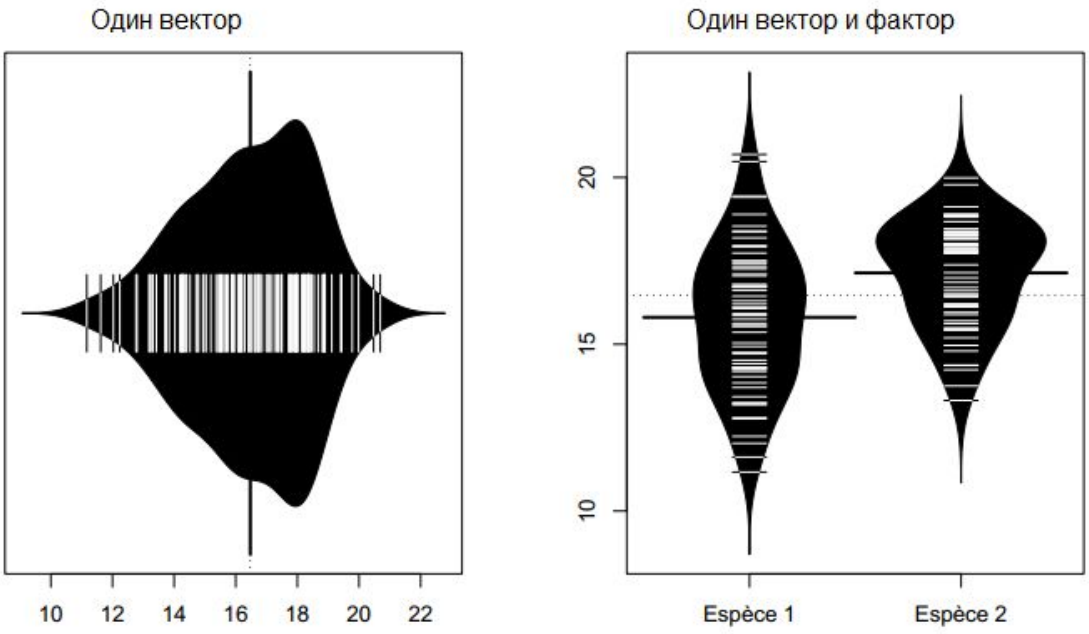
Для представления фигур в горизонтальной ориентации необходимо добавить аргумент `horizontal=TRUE`.

Чтобы добавить заголовок к графику, используют аргумент `main="Заголовок"`.

Чтобы изменить подписи горизонтальной оси (если фигуры ориентированы горизонтально), нужно использовать аргумент `xlab="Легенда"`.

Чтобы изменить заголовок вертикальной оси (если фигуры ориентированы вертикально), нужно использовать аргумент `ylab="Легенда"`.

Для знакомства с другими вариантами графика можно вызвать справку `? beanplot` (в том числе аргументы, определяющие цвета фигур).



34. Столбчатые диаграммы с интервалом погрешности

В примере, представленном ниже, возможны несколько вариантов визуализации средних значений и их стандартных ошибок. Выполняемые команды **R** могут быть, конечно, адаптированы и к любому иному виду.

Один фактор

На предварительном этапе выполняется расчет для каждого уровня фактора средних значений и ошибок средних (см. п. **37**). Оба вектора `средние` и `ошибки` должны содержать эти значения, сопряженные между собой, и в той последовательности, в какой они представляются на графике слева направо. Обычно очень полезной в этой ситуации является функция `tapply()` – см. п. **35**. Процедура формирования графика заключается в следующем :

```
> x <- barplot(средние)
> arrows(x, средние-ошибки, средние, средние+ошибки, code=3,
                    angle =90, length=0.15)
```

Два фактора

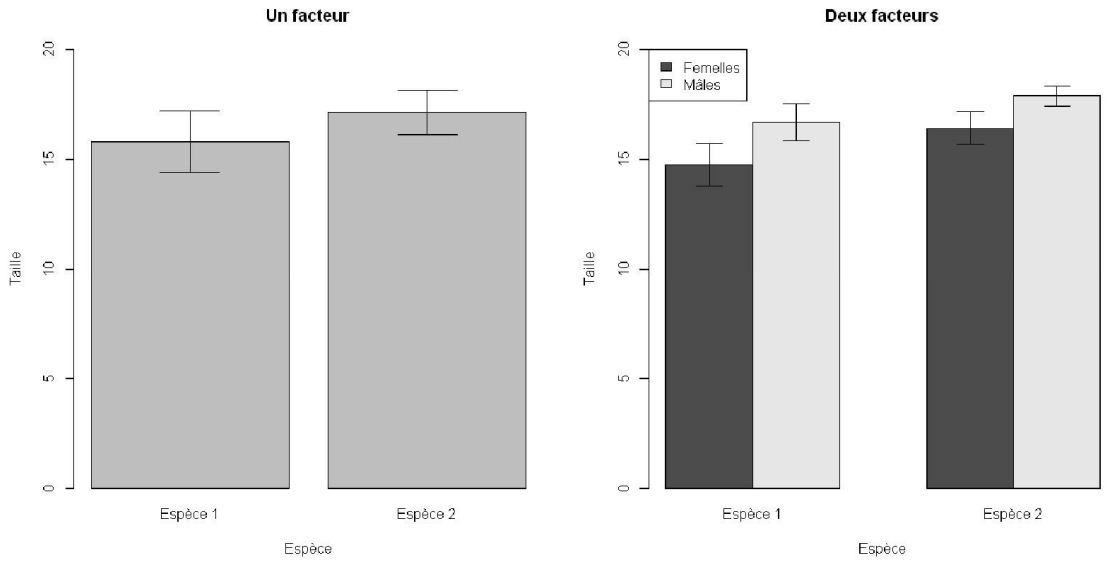
Средние и стандартные ошибки должны быть включены в состав матрицы. Эта матрица должна иметь строки, соответствующие уровням 2-го фактора, и столбцы, соответствующие уровням 1-го фактора. Здесь также может быть полезна функция `tapply()`, см. п. **35**. Процедура построения графика аналогична: необходимо только добавить аргумент `beside=TRUE` в функции `barplot()`.

Аргумент `names.arg = имена` функции `barplot()` добавляет или изменяет имена панелей (`имена` – вектор, содержащий эти имена слева направо). Аргумент `legend=TRUE`, в случае двух факторов добавляет на график легенду.

Как всегда, можно также использовать следующие дополнительные аргументы:

- добавить заголовок к графику: `main="Заголовок"`;
- добавить заголовок вертикальной оси: `xlab="Легенда"`;

Используйте `? barplot`, чтобы получить больше справочной информации.



ВЫБОРОЧНЫЕ ПАРАМЕТРЫ

35. Параметры положения

Параметры положения позволяют выявить среднюю тенденцию набора данных. Три из них – наиболее часто используемы и реализуются функциями:

- среднее – `mean(вектор)`, где `вектор` представляет собой одномерную выборку значений;
- медиана – `median(вектор)`;
- мода – `mod(вектор)*`.

Если векторы содержат недостающие данные (`NA`), то в функции `mean()` и `median()` необходимо добавить аргумент `na.rm=TRUE`. Функция `mod()*` по умолчанию сама управляет недостающими данными.

Для расчета значений параметра, соответствующих уровням фактора, используют функцию `tapply(вектор, фактор, function(x) function)`, где `вектор` и `фактор` являются векторами, содержащими значения каждого наблюдения для обоих переменных (в одном и том же порядке). В обозначении используемой функции `вектор` должен быть заменен на `x`: `mean(x)`, `median(x, na.rm=TRUE)` и т.д.

ПРИМЕРЫ

```
> переменная <- c(1:60)
> фактор <- factor(rep(LETTERS[1:3], each=20))
> tapply(переменная, фактор, function(x) mean(x))
    A    B    C
 10.5 50.5 30.5
```

Если есть второй фактор, то функция `tapply()` возвращает матрицу :

```
> фактор2 <- factor(rep(letters[1:2], 30))
> tapply(переменная, list(фактор2, фактор1),
                          function(x) mean(x))
   A  B  C
a 10 30 50
b 11 31 51
```

в которой первый фактор определяется строки, а второй определяет столбцы.

36. Параметры вариации

Параметры вариации позволяют оценить изменчивость ряда данных. Три из них используются наиболее часто и реализуются следующими функциями:

- дисперсия – `var(вектор)`, где `вектор` представляет собой одномерную выборку значений;
- стандартное отклонение (standard deviation) – `sd(вектор)`;
- коэффициент вариации (по абсолютной величине или в процентах) – `cv(вектор)*`.

Если векторы содержат недостающие данные (`NA`), в функции `var()` и `sd()` необходимо добавить аргумент `na.rm=TRUE`. Функции `cv()*` по умолчанию сама управляет недостающими данными.

Функции `var()` и `sd()` вычисляют несмещенные оценки дисперсии и стандартного отклонения, т. е. на основе (`n` – 1), а не `n`, где `n` – объем выборки. Функция `cv ()*`, использующая `sd()`, также вычисляет несмещенное значение параметра.

Для расчета значений параметра, соответствующих уровням фактора, используют функцию `tapply(вектор, фактор, function(x) function)`, где `вектор` и `фактор` являются векторами, содержащими значения каждого измерения для обоих переменных (в одном и том же порядке). В обозначении используемой функции `вектор` должен быть заменен на `x`: `var(x)`, `sd(x,na.rm=TRUE)` и т.д.

ПРИМЕРЫ

```
> переменная <- c(1:60)
> фактор <- factor(rep(LETTERS[1:3], each=20))
> tapply(переменная,фактор, function(x) sd(x))
    A    B    C
5.92 5.92 5.92
```

Если есть второй фактор, то функция `tapply()` возвращает матрицу:

```
> фактор2 <- factor(rep(letters[1:2], 30))
> tapply(переменная, list(фактор2, фактор1),
                              function(x) sd(x))
     A    B    C
a 6.06 6.06 6.06
b 6.06 6.06 6.06
```

в которой первый фактор определяется строки, а второй определяет столбцы.

Не путайте параметры вариации с доверительным интервалом или ошибкой среднего (см. п. **37**). Параметры вариации оценивают изменчивость данных, в то время как доверительный интервал и ошибка среднего указывают на точность параметра положения (см. лист **35**). В частности, объем выборки данных не влияет на разброс генеральной совокупности (изменчивость не увеличивается или не уменьшается от числа измерений), но определяет точность оценки положения (эта точность возрастает с увеличением числа измерений).

37. Доверительный интервал и стандартные ошибки

Доверительный интервал и стандартные ошибки позволяют оценить точность различных параметров. Из них наиболее распространены расчеты в отношении трех параметров: среднего, медианы (для которой имеет смысл только доверительный интервал) и пропорции.

Для всех функций, вычисляющих доверительные интервалы, точность этого интервала может быть изменена с помощью аргумента `conf.level` (по умолчанию `conf.level=0.95`, которая вычисляет 95 %-й доверительный интервал).

Среднее

Доверительный интервал. Если выборочные данные имеют достаточно большой объем (`n` $\geq$ 30 измерений в серии), то доверительный интервал может быть рассчитан параметрически : `t.test(серия)$conf.int`, где `серия` – вектор, содержащий набор выборочных данных.

Если объем выборки мал (`n` < 30 наблюдений), то доверительный интервал рассчитывается непараметрическим методом, например, с использованием бутстрепа, реализуемого, например, функцией:
`bootstrap(серия, function(x,i) mean(x[i]))*`.

Стандартная ошибка среднего вычисляется независимо от объема выборки – `se(серия)*`. Для расчета стандартных ошибок для различных уровней фактора также можно воспользоваться функцией `tapply()`, см. п. **36**.

Медиана

Доверительный интервал. Если набор данных не содержит 0, то независимо от объема выборки он вычисляется с использованием теста знаков Уилкоксона: `wilcox.signtest(серия)$co-nf.int*`.

Если выборочный ряд содержит значения 0, то необходимо задать теоретическое значение медианы `mu = значение` и тогда:

```
wilcox.signtest(серия, mu = значение)$co-nf.int*
```

Доля

Доверительный интервал для долей, независимо от от объема выборки вычисляется как `binom.test(a, b)$conf.int` где `a` – число объектов, относящихся к интересуемой категории, из общей численности `b`.

ПРИМЕР: доверительный интервал для соотношения полов, где насчитывается 9 самок из группы в 25 индивидуумов (т. е. 0.36):

```
> binom.test(9,25)$conf.int
[1] 0.1797168 0.5747937
```

Стандартная ошибка для доли, не зависящая от объема выборки, вычисляется как: `se(a,b)*` .

38. Идентификация и удаление выбросов

Идентификация выбросов является обязательным этапом любого статистического анализа. Она выполняется в основном визуально, с помощью диаграммы типа *гистограмма* (см. п. **31**) или *ящик с усами* (см. п. **32**).

Удаление из выборки значений возможных выбросов является значительно более щекотливой процедурой, чем их простая идентификация. Если удалить одно или несколько значений, то это может оказать влияние на результаты анализа, и этот эффект будет проявляться тем сильнее, чем меньше объем выборки. Обычно стараются удалить наблюдения, которые потенциально могут привести к выводам, которые обратные ожидаемым в исследовании.

В целом, есть только две причины, которые должны подтолкнуть исследователя к удалению блока данных:

- была очевидная техническая ошибка в измерении или в записи данных (например, измерительный прибор оказался неисправен);
- если наблюдаемое значение является настолько маловероятным, что выходит за пределы шкалы измерения переменной.

Помимо этих двух случаев, чаще всего выброс является просто "аномальной" величиной, отражающей, например, некоторую особенность биологического индивида. Она является выражением естественной изменчивости биологического характера, и ничто не указывает на то, что наблюдение быть исключено из исследования.

В любом случае, идентификация и удаление выбросов необходимо проводить до начала любого другого анализа.

СТАТИСТИЧЕСКИЕ МОДЕЛИ И ТЕСТЫ

Как ориентироваться в "джунглях" тестов и моделей?

Чтобы увереннее приступить к ориентированию в множестве алгоритмов статистического анализа, попробуем на время забыть обычный словарный запас ("средние", "пропорции", "частота" и т.д.), и сосредоточиться только на природе анализируемой переменной, независимо от того, что именно она кодирует. Мы рассмотрим в качестве отправных точек следующие ситуации (см. п. **11**, если не ясны основные предпосылки):

1. Отклик в виде **качественной переменной**:
   (а) *номинальный*:
      - состоит из 2 классов (т. е. двоичный 0/1);
      - состоит более чем из 2-х классов;

   (б) *порядковый* (т. е. место в рейтинге).
2. Отклик в виде **количественной переменной**:
   (а) *счетный* (т. е. подсчет числа экземпляров) :
      - относится к одной категории
      - относится к различным категориям:
         — к 2 категориям;
         — более чем к 2 категориям;

   (б) *количественный* (теоретически, хотя не обязательно в действительности):
      - на ограниченном диапазоне (фактически), т. е. случайная величина варьирует в пределах некоторого интервала возможных значений (например, от 0 до 1 для долей);
      - неограниченный, т.е. если даже переменная не может принимать все возможные значения, но границы ее варьирования не определены.

Примечание: если отклик – время до возникновения некоторого события, он обрабатывается с помощью специальных методов.

Как правило, независимых переменных должны быть гораздо меньше, чем измерений, и они не должны быть слишком коррелированы между собой.

Мы условно разделили все статистические методы на два типа: **"простые" тесты** и анализ с использованием **моделей**. Рассматриваемые тесты также условно разделены на две группы: *непараметрические* (в основном) тесты, для использования которых не требуются специальные знания в области статистики, и *параметрические* тесты, основанные на свойствах статистических распределений и использующие немного более продвинутые понятия.

Модели являются объективно более мощными и позволяют выполнить более глубокий анализ, чем "простые" тесты, и их использование априори является более предпочтительным, если есть несколько альтернатив. Однако, можно ориентироваться и на простые тесты, если не соблюдаются условия использования модели, или мы хотим сознательно использовать тот или иной тип тестирования.

Ниже в разделах **39 – 42** мы предварительно определим основные правила построения, анализа и селекции моделей, которые могут быть полезны читателю в практической работе.

39. Общий подход к построению моделей

Построение статистических моделей (во всяком случае, в биологической статистике) концептуально заключается в выполнении почти всегда типичных шагов:

1. *Определяется конструкция модели*: (1) выбирается тип используемой модели (линейная, нелинейная, обобщенная линейная или аддитивная и т.д.); (2) *apriori* постулируется закон распределения, на котором должны быть основаны модели; (3) определяются связи между независимыми переменными в этой модели. Шаги (1) и (2) обсуждаются в рамках этого путеводителя, в то время как шаг (3) обязательно должен быть выполнен самим пользователем, так как он зависит от задач исследования. На этом шаге задается формула модели (см. п. **40**).

2. *Проверяется качество подгонки модели к данным*. Важно, чтобы эта проверка осуществлялась с использованием возможно более широкого набора реальных данных, иначе модель может оказаться предвзятой (или даже быть полностью искажена). На практике такая проверка происходит путем анализа остатков модели (см. п. **41**).

Если модель недостаточно адекватна, то есть несколько возможностей : (1) полностью изменить модель, (2) изменить закон, на котором она основана, (3) изменить характер связей между независимыми переменными, (4) преобразовывать данные. В любом случае, модель, не очень хорошо соответствующая анализируемым данным, не должна рассматриваться дальше.

3. Дальнейший процесс анализа модели может проходить в двух разных направлениях, в зависимости от типа и контекста проводимого исследования:

- (А) подход, основанный на *проверке влияния независимых переменных*. Для этого выполняется тестирование значимости параметров модели, т.е. рассчитываются *p*-значения, оценивающие значимость эффекта воздействия каждого из предикторов, включенных в формулу (см. п. **42**). При необходимости выполняются множественные сравнения: если статистически значимый сопутствующий фактор имеет более двух уровней, то множественные сравнения позволяют уточнить, на каких конкретно группах наблюдений изменчивость отклика имеет место на самом деле (см. п. **43**).
- (Б) подход, основанный на *селекции моделей*. Обычно рассматривается множество моделей, производных от наиболее полной, но не включающих некоторые ее члены. Эти модели сравниваются между собой и с исходной моделью, чтобы выбрать ту из них, которая является лучшим компромиссом между точностью подгонки данных и числом независимых переменных (см. п. **44**).

Обычно считается, что, когда целью является объяснение роли наблюдаемых данных, то предпочтительным является подход (А), связанный с тестированием переменных, а когда цель заключается в создании прогнозирующей модели, то здесь важнее селективный подход (Б).

40. Запись формулы модели

Запись формулы модели является важным элементом статистического анализа. Этот этап является относительно простым, если исследователю вполне ясны следующие основные отправные точки для построения модели:

- определен тип отклика и предикторов модели (независимых переменных);
- предикторы разделены на количественные переменные и факторы (см. п. **11**);
- уточнен статус фиксированных и случайных факторов (см. п. **11**);
- известен план взятия выборок (см. п. **12**) и проведения эксперимента (см. п. **13**).

В принципе любая формула модели имеет следующий вид:

```
объясняемые.переменные ~ независимые.переменные
```

Символ `~` означает "*установить соответствие* или *функциональную зависимость*").

На практике чаще всего используют только одну объясняемую переменную, что упрощает левую часть формулы. Зато обычно необходимо установить роль каждой независимой переменной в правой части и задать соответствующие связи между ними. Важно понимать, что эти отношения зависят от сущности исследования. Например, установить являются ли факторы случайными или фиксированными, следует, скорее всего, адекватно конкретной ситуации в эксперименте.

Существует три типа связи между двумя независимыми переменными, представляемые тремя различными символами:

- влияние каждой переменной рассматривается отдельно: `A + B`.
- анализируется влияние каждой переменной и их парное взаимодействие: `A*B`. Синтаксис этой записи строго эквивалентен `A + B + A:B`, что означает «*переменная* A *и переменная* B *и взаимодействие между* A *и* B». Если переменных более двух, то возникает проблема: `A*B*C` эквивалентно `A + B + C + A:B + A:C + B:C + A:B:C`, что означает «*влияние базовых переменных* (A, B, C) *и взаимодействия* 2-го *порядка* (между A и B, A и C, B и C) и *взаимодействие* 3-го *порядка* (между A, B и C)». Может показаться заманчивым, получить все возможные взаимодействия факторов, но есть одна вещь, которую хорошо бы избежать. Действительно, вытекающие из приведенной формулы взаимодействия 3-го порядка очень трудно интерпретировать, в частности: «*эффект воздействия* A *зависит от* B, *но также и от* C...». Разумно ограничить модель эффектами, которые мы можем объяснить. Кроме того, эффекты тройного взаимодействия факторов очень слабо влияют на целевую переменную.
- особым является случай, когда один из двух факторов, является вложенным (т.е. располагается ниже по иерархии): `A/B`, что означает «*переменная* A *и переменная* B, *вложенная в* A» (например, фактор численностей различных популяций зафиксирован для конкретного уровня фактора региона). Можно расширить далее подобные отношения `A/B/C` (например, если рассматривается иерархия факторов: страна, регион и популяция).

Эти три символа (на самом деле четыре, считая "`:`") позволяют определить все отношения между независимыми переменными, будь то количественными или качественными.

Определенная особенность заключается в обозначении случайных факторов (см. пп. **11**-**13**). При этом используется специальный синтаксис, который зависит от используемой функции. Встречаются несколько вариантов записи:

Для "простых" тестов случайный фактор записывается в конце формулы после знака "`|`". Например, в формуле `отклик ~ фактор|блок` эффект `фактор` является фиксированным, а эффект `блок` определен как случайный.

Особый класс образуют смешанные модели (так называют модели со смешанными параметрами), которые содержат, по крайней мере, один случайный фактор. Несколько пакетов управляют построением этих моделей (каждый с собственным синтаксисом), но мы в этом документе будем рассматривать только использование пакета `lme4`. Случайный фактор представляется всегда в конце таких формул в виде `(1|случайный.фактор)`. Например, в формуле `отклик ~ фактор+(1|блок)` переменная `фактор` является фиксированной, а фактор `блок` является случайным. Несколько случайных факторов могут быть введены в формулу в виде `(1|фактор1) + (1|фактор2)`, если они независимы, или `(1|фактор1/фактор2)`, если они находятся в иерархических отношениях.
Можно конструировать гораздо более сложные формулы смешанных моделей и мы рекомендуем книгу Бейтса (Bates, 2010), чтобы получить больше информации.

Формулы представляют своеобразный шаблон функциональной части модели, причем, как правило, все функции, основанные на формуле, принимают аргумент `data`. С его помощью можно указать таблицу данных, в которой следует искать переменные, содержащиеся в формуле, что позволяет избежать удлинения синтаксиса формулы (а иногда и избежать ошибок).

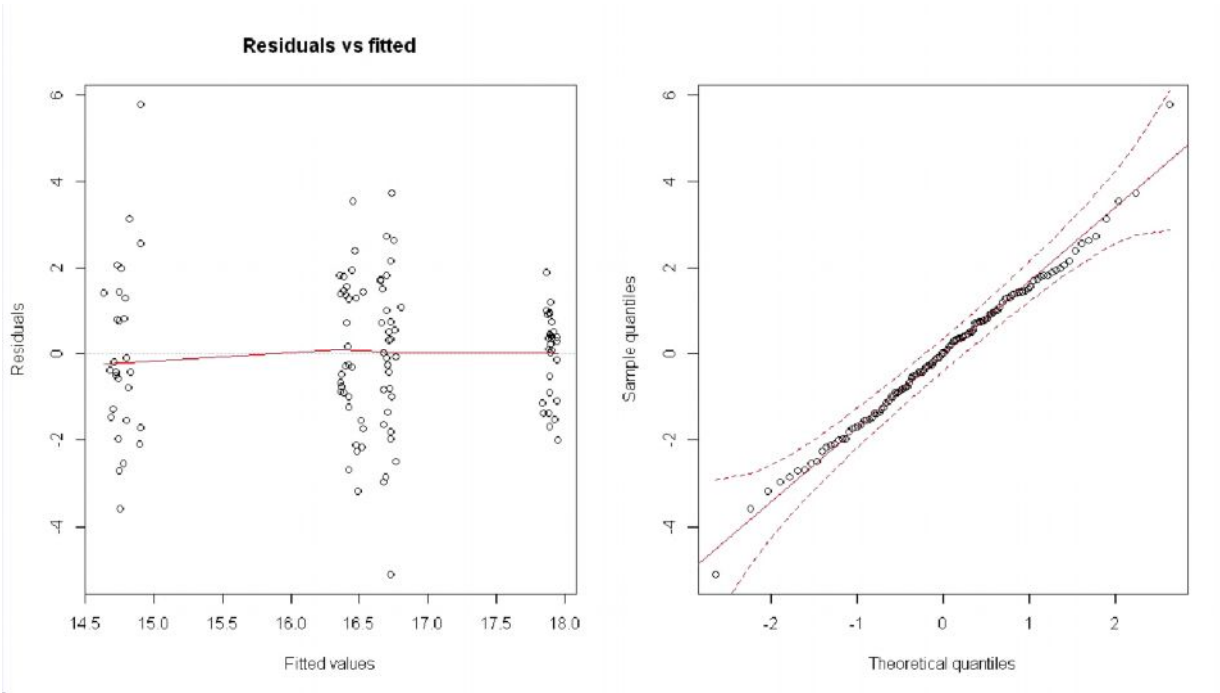
41. Проверка адекватности модели

Построить модель для анализа наблюдений – это только первый этап, но далее очень важно убедиться в том, что эта модель хорошо описывает выборочные данных. Если это не так, то любой анализ, вытекающих из этой модели будет, как минимум, предвзятым или даже совершенно неверным.

Процедуры, выполняемые для проверки адекватности модели, основаны, как правило, на графической интерпретации и сосредоточены в целом относительно трех пунктов: *гомоскедастичность*, *независимость* и *нормальность* остатков модели. Остатки – это разности между действительно наблюдаемыми значениями и теми, которые предсказаны с помощью модели (или *fitted values*).

В пакете `RVAideMemoire` эти три основные предпосылки адекватности проверяются с помощью функции `plotresid(модель)`*. На выходе формируются два графика:

- график слева служит для проверки гомоскедастичности (или однородного разброса случайных ошибок) и независимости остатков. На этом графике красная линия представляет кривую тренда облака точек. Гипотеза о независимости принимается, если ориентация облака точек по горизонтали симметрична относительно центральной тенденции (т. е. красная линия проходит примерно посередине этого облака). Гипотеза гомоскедастичности принимается, если разброс точек по вертикали является почти постоянным на всей длине оси абсцисс.
- график справа служит для проверки нормальности остатков. Гипотеза нормальности не отклоняется, если точки находятся почти на одной прямой (см. п. **65**).



42. Тестирование модели

После того, как выполнена проверка того, насколько хорошо построенная модель описывает данные (см. п. **41**), может быть проверена степень влияния независимых переменных. Существует три способа выполнения этого теста, так называемые типы I, II и III. Если есть только одна независимая переменная, то все три вида теста дают в точности один и тот же результат. Если среди независимых переменных имеется несколько факторов, но численность групп сбалансирована (точнее, одинаковая численность во всех комбинациях факторов, влияющих на взаимодействие), то тесты снова дадут одинаковые результаты. Три типа тестов дают разные результаты, если несколько факторов не сбалансированы по численности. Важно понимать, откуда проистекают эти различия, поскольку они отражают различные гипотезы статистики (и, следовательно, биологии). Возьмем, например, модель типа `отклик ~ A + B + A:B`, где `A` и `B` – факторы (см. п. **40** для объяснения, что означает эта запись формулы).

Тип I: анализ выполняется последовательно, т.е. условия проверяются в том порядке, в котором они размещены в формуле. Заметим, что когда создается модель, **R** автоматически выбирает из формулы сначала по порядку главные факторы, а затем взаимодействия 2-го порядка, взаимодействия 3-го порядка и т.д. Таким образом, в нашем примере мы: (1) осуществляем тестирование эффекта `A`; (2) после того, как исключили эффект `A` , выполняем оценку влияния фактора `B`; (3) тестируется совместный эффект `A:B` после снятия эффектов `A` и `B`. Это означает, что результаты проверки значимости факторов `A` и `B` зависят от порядка, в котором они указаны в формуле. Эта последовательность может не соответствовать тем гипотезам, которые мы хотим проверить в биологии, поэтому тип I лучше не использовать (за исключением, конечно, тех случаев, когда необходимо использовать именно этот механизм проверки). Процедуры типа I реализуются функцией `anova(модель)`.

Тип II: анализ проводится не последовательно, т.е. не зависит от порядка расположения каждого члена в формуле. Зато соблюдается принцип маргинальности, который гласит, что «в случае, когда взаимодействие между факторами имеет значимый эффект, то любые эффекты этих переменных, учитываемые в членах более высшего порядка, являются вторичными перед этим взаимодействием». Другими словами, этот принцип утверждает, что если взаимодействие между двумя факторами значимо, то нет смысла учитывать локальное влияние главных факторов. Аналогично для модели с тремя факторами: если тройное взаимодействие факторов оказывает значимое влияние, то не стоит обращать внимание как на эти факторы отдельно, так и на взаимодействия 2-го порядка с участием этих факторов. Следовательно, считается, что, если выделено влияние только одного фактора, то все комбинации высшего порядка, связанные с этим фактором, не обладают значимым эффектом. В нашем примере это сводится к следующей процедуре: (1) тестируется значимость эффекта `A` после снятия эффекта `B`; (2) тестируется эффект `B` после того, как сняли эффект `A`; (3) тестируется значимость совместного действия `A:B` после того, как сняли эффекты `A` и `B`. Этот подход постоянно используется в этом путеводителе, потому что в подавляющем большинстве случаев он является наиболее эффективным при проверке биологических гипотез. Тест II типа осуществляется на основе функции `Anova(модель, type ="II")` из пакета `car`, или более просто `Anova(модель)`, поскольку тип II используется по умолчанию.

Тип III: анализ также выполняется не последовательно и по иерархическому механизму. При этом способе парное взаимодействие тестируют, когда оба других фактора уже есть в модели, но и каждый фактор тестируют, когда все другие факторы

и взаимодействие уже есть в модели. В нашем примере это реализует следующая процедура: (1) тестируется значимость эффекта `A` после снятия главного эффекта `B` и парного взаимодействия `A:B`; (2) тестируется эффект `B` после того, как сняли эффекты `A` и `A:B`; (3) тестируется значимость совместного действия `A:B` после того, как сняли эффекты `A` и `B`. Этот подход может показаться интересным, но, во-первых, его очень сложно использовать, потому что он связан с условием, что какой-то фактор может не иметь значимого эффекта только потому, что оказался значимым эффект другого фактора. Кроме того, он редко дает корректные результаты при проверке биологических гипотез. Тест III типа осуществляется на основе функции `Anova(модель, type ="III")` из пакета `car`

43. Множественные сравнения, основанные на моделях

Если при тестировании модели было выявлено значимое влияние, по крайней мере, одного фактора (или взаимодействия с участием этого фактора), необходимо провести множественные сравнения, чтобы увидеть, на каких именно уровнях имеет место значимое воздействие на отклик. Метод сравнения, представленный здесь, является одним из самых интересных, поскольку он учитывает влияние других переменных модели. Другими словами, сравнения делаются после того, как учтена доля вариации отклика за счет влияния других независимых переменных. В целом это напоминает оценку и последующее сравнение групповых средних (например, мы их вычисляли в п. **35**), но в нашем случае эти средние значения скорректированы с учетом других независимых переменных. Речь идет о методе *средних наименьших квадратов*, или **LSMeans** (сокращенная аббревиатура от *Least Squares Means*). Будем при этом использовать функции пакетов `lsmeans` и `RVAideMemoire`.

Примечание 1: для оценки взаимодействия между фактором и количественной независимой переменной (далее называемой ковариатой), сравниваются не средние значения, а угловые коэффициенты (т.е. тангенсы угла наклона между между переменной-откликом и ковариатой, рассчитанные для каждого уровня фактора).

Примечание 2: этот раздел касается всех моделей, представленных в этом путеводителе, кроме тех, которые моделируют номинальной отклик, имеющий более чем две категории (см. п. **51**), или подсчет экземпляров для двух категорий (см. п. **63**).

Расчет скорректированных средних / коэффициентов угла наклона

Один фактор или взаимодействие двух факторов – сравнение средних

Первый шаг заключается в расчете скорректированных групповых средних, которые хранятся в объекте под названием `LSM`. Синтаксис создания объектов этого типа:
`LSM <- lsmeans(модель, ~фактор)`, где `фактор` – обозначение главного фактора или парного взаимодействия (внимание!, не забывайте символ `~`).
Для расчета средних значений отдельно для каждого уровня другого фактора:
`LSM <- lsmeans(модель, ~фактор1|фактор2)`, где `фактор1` – фактор (или взаимодействие), для которого мы хотим выполнить сравнение по группам и `фактор2` – фактор, для которого мы хотим осуществить парные сравнения внутри каждой группы.

Примечание 1: для моделей GLMM, основанных на отрицательном биномиальном распределении (см. п. **55**) и созданных функцией `glmer.nb()`, необходимо уточнить таблицу данных, для которой рассчитана модель, с использованием аргумента `data`. Во всех остальных случаях этого не требуется.

Примечание 2: для многомерных линейных моделей MLM (см. п. **108**) скорректированные средние могут быть рассчитаны сразу отдельно для каждой

объясняемой переменной. Для этого применяется следующий синтаксис, в котором использование `rep.meas` обязательно:

```
LSM <- lsmeans(модель, ~фактор|rep.meas).
```

ПРИМЕРЫ

Будем использовать модель под названием `модель`, имеющую формулу `отклик ~ A*B+C`, где `A`, `B` и `C` – факторы (см. п. **40** для объяснения формулы).

Расчет скорректированных средних для каждого уровня фактора `A`:

```
> LSM <- lsmeans(модель, ~A)
```

Для всех вариантов парного взаимодействия `A:B` :

```
> LSM <- lsmeans(модель, ~A:B)
```

Для каждой группы фактора `A` отдельно сравниваются уровни фактора `C` :

```
> LSM <- lsmeans(модель, ~A|C)
```

Для каждого варианта парного взаимодействия факторов `A` и `B` отдельно сравниваются уровни фактора `C` :

```
> LSM <- lsmeans(модель,~A:B/C)
```

Если анализируется модель GLMM, основанная на отрицательном биномиальном распределении, то указывается таблица данных, называемая `таблица`:

```
> LSM <- lsmeans(модель,~A, data=таблица)
```

Если анализируется модель `MLM` и нужно вычислить скорректированные средние значения отдельно для каждой объясняемой переменной:

```
> LSM <- lsmeans(modele,~A|rep.meas)
```

Взаимодействие фактора с ковариатой - сравнение коэффициентов угла наклона

Коэффициенты угла наклона рассчитываются с помощью функции `lstrends()`, синтаксис которой очень близок к `lsmeans()`:

```
наклон <- lstrends(модель, ~фактор, var="ковариата"),
```

где `фактор` может быть взаимодействием нескольких факторов. Обратите внимание на кавычки вокруг имени `ковариата`.

Примечание: функция `lstrends()` недоступна для использования с MLM. Обычно для таких многомерных моделей остается довольно мало информации, чтобы проводить анализ влияния ковариат (см. п. **108**).

Матрицы контрастов

Функции, приведенные в данном разделе, позволяют по умолчанию реализовать все возможные попарные сравнения. В случае, если мы хотим осуществить сравнения в особом пользовательском варианте, необходимо начать с создания матрицы контрастов. Она имеет вид :

	Уровень1	Уровень2	Уровень3
Контраст1	1	-1	0
Контраст2	0	1	-1
Контраст3	2	-1	-1

В этой матрице сравниваемые комбинации (или *контрасты*) представлены в строках, а уровни фактора (или *взаимодействия*) – в столбцах. Существуют следующие условия создания корректных записей контрастов:

- комбинации, не участвующие в сравнении должны иметь нулевое значение;
- сравниваемые уровни должны иметь ненулевое значение и противоположный знак;
- можно выделить контрольные и подопытные группы (например, в третьей

строке, группа Уровень1 сравнивается с группами Уровень2 и Уровень3);

- сумма значений положительных и отрицательных значений для каждого контраста должна быть равна нулю.

Внимание: **R** считает, что уровни факторов заданы в алфавитном порядке, т. е. 1-й столбец соответствует 1-му уровню в алфавитном порядке, и так далее.

Матрицы контрастов – это обычная таблица, которая может быть создана непосредственно в **R** (см. п. **5**) или импортирована из электронных таблиц (см. п. **6**).

Реализация сравнений

Для выполнения всех возможных парных сравнений используют функцию: `contrast(объект, "pairwise")*`, где `объект` – рассчитанные выше групповые средние `LSM`, либо угловые коэффициенты `наклон`.

В случае пользовательских сравнений процедура выполняется в два этапа :

```
> cont.lsmc <- user.cont(контрасты)
> contrast(объект, "cont")
```

где `контрасты` – матрица контрастов, задающая комбинаторику сравнений.

Функция `contrast()*` имеет много преимуществ в применении, поскольку рассчитывает *p*-значения для каждой комбинации сравнения, но требует предварительной группировки уровней (группы типа a, ab, b,...). Существует метод автоматической группировки: `cld(объект)`, однако эта функция не возвращает *p*-значение для каждого акта сравнения и обычно используется как дополнение к первой.

Примечание 1: для того, чтобы использовать функцию `cld()` необходимо установить пакет `multcompView`.

Примечание 2: в случае MLM (см. п. **108**) проведенные множественные сравнения, учитывают все независимые переменные одновременно. Однако, если объект `LSM` был создан с помощью `rep.meas`, то в этом случае сравнение осуществляется отдельно для каждой объясняющей переменной.

Тесты статистической значимости различия средних/углов наклона

Всегда интересно проверить, отличаются ли от 0 разности между отдельными групповыми средними или углами наклона. Серию этих тестов можно осуществить автоматически с помощью `summary (объект, infer=TRUE)`.

Обратное преобразование скорректированных средних / коэффициентов угла наклона

Один фактор или взаимодействие двух факторов - сравнение средних

Представить на диаграмме средние значения, скорректированные с учетом уровней фактора (или сочетания двух факторов), часто более эффективно, чем просто привести их численные значения. Это позволяет проиллюстрировать, какие изменения отклика происходят на самом деле под влиянием этого фактора, после того, как сняли долю вариации за счет других независимых переменных модели.

Для всех моделей, за исключением моделей ранговой классификации (в п. **52** рассматривается подход, используемый в этом случае), обратное преобразование скорректированных средних выполняется следующим образом:

- Если объясняемая переменная не была преобразована до создания модели: `значения <- as.data.frame(summary(LSM, type="response"))`. Объект `значения` – это таблица, в которой первые столбцы соответствуют факторам по порядку, заданному в `lsmeans()`. Столбцы, названные `lsmean`, `response`, `prob`, `rate` и `hazard`, содержат информацию о скорректированных средних. Столбец `SE` содержит стандартную ошибку, связанную с каждой средней.

- Если объясняемая переменная была преобразована до создания модели (на практике это проводится только для некоторых LM/LMM):

`значения <- back.lsmeans(LSM,transform="трансф"),add=значение)`, где `трансф` – имя преобразующей функции в кавычках (`"log"`, `"sqrt"`, `"inverse"` или `"logit"`) и `значение` – возможная константа, добавляемая к объясняющей переменной до трансформации. Объект `значения` – это таблица, в которой первые столбцы соответствуют факторам по порядку, заданному в `lsmeans()`. Столбец `Mean` содержит скорректированные средние, а столбцы. `SE.inf` и `SE.sup` содержат граничные значения стандартной ошибки, которая обязательно асимметрична в связи с преобразованием объясняемой переменной.

ПРИМЕРЫ

Используется модель, называется `модель`, построенная с помощью формулы `отклик ~ фактор` (см. п. **40** для объяснения формулы), в которой отклик преобразован путем извлечения квадратного корня $x^{0.5}$. Скорректированные средние получаются следующим образом :

```
> LSM <- lsmeans(модель, ~фактор)
> back.lsmeans(LSM, transform="sqrt")*
```

Если преобразование типа ln(x + 1):

```
> back.lsmeans(LSM, transform="log", add=1)*
```

Если преобразование типа log10(x + 1) :

```
> back.lsmeans(LSM, transform="log", add=1, base=10)*
```

После того как средние и их стандартные ошибки восстановлены, они могут быть представлены на графике в виде гистограммы (см. п. **34**).

Примечание: в случае MLM (см. п. **108**), корректировки средних имеют смысл только в том случае, если они рассчитываются для каждой объясняющей переменной. Таким образом, необходимо использовать `rep.meas`, чтобы восстановить их.

Взаимодействие фактора с ковариатой - сравнение коэффициентов угла наклона

Коэффициенты угла наклона восстанавливаются следующим образом : `значения <-as.data.frame(summary(наклон)).` Объект `значения` – это таблица, в которой первые столбцы соответствуют факторам по порядку, заданному в `lsmeans()`. Столбец, имя которого заканчивается `.trend`, содержит скорректированные коэффициенты угла наклона. Столбец `SE` содержит стандартные ошибки, связанные с каждым угловым коэффициентом.

44. Селекция моделей

Чтобы увеличить эффективность прогнозирующих моделей, обычно используют различные алгоритмы выбора наиболее информативного набора переменных. Целью селекции является достижение компромисса между точностью подгонкой модели к данным (которая растет с числом независимых переменных), и, по возможности, небольшим числом независимых переменных (их неконтролируемое увеличение приводит к потере способности модели обобщать результаты, что становится проблематичным для прогнозирования).

Выбор оптимальной модели основывается из условия минимума задаваемого критерия, оценивающего этот компромисс. При его увеличении либо существенно снижается точность модели, либо неоправданно включается слишком много малосущественных переменных. Существует несколько таких критериев, из которых наиболее часто используется информационный критерий Акаике (AIC).

На практике селекция осуществляется автоматически путем исключения переменных из исходной полной модели. Для выбора субоптимальных моделей методом полного перебора используется функция: `dredge(модель)` из пакета `MuMIn`, где `модель` – это модель, содержащая все имеющиеся независимые переменные. Эта функция вычисляет AIC всех возможных моделей, начиная от исходной модели, и возвращает отсортированную таблицу наилучших моделей-претендентов. Среди необязательных аргументов функции два особенно интересны:

- `m.max = значение`, где `значение` –максимальное количество независимых переменных для включения в тестируемые модели;
- `fixed = список.переменных`, где `список.переменных` – вектор, содержащий имена переменных для включения в модели (в кавычках).

Отметим, что, если $n/df < 40$, где n – это объем выборки, и df – количество параметров, включаемых в модели, возвращаемые функцией `dredge()`, нужно использовать AIC_c (т.е. скорректированный информационный критерий AIC). Поскольку эта ситуация является стандартной практикой, функция `dredge()` использует по умолчанию AIC_c для селекции моделей.

Для моделей, предполагающих квази-пуассоновское или квази-биномиальное распределения, в качестве критерия используется QAIC, являющийся производным от AIC для распределения "квази". Для селекции таких моделей необходимо знать значение параметра избыточной дисперсии, который можно получил через `summary(модель)` и задать с помощью аргумента `chat`.

Чтобы использовать для селекции моделей какой-либо другой критерий, можно задать аргумент `rank` (для квази-моделей возможны лишь значения "AIC" или "QAIC" в зависимости от ситуации).

ПРИМЕРЫ

Для модели типа "квази":

```
> модель <- glm(формула,family="quasipoisson")
```

Значение параметра избыточности дисперсии установлено с помощью:

```
> summary(модель)
[...]
(Dispersion parameter for quasipoisson family taken to be 1.126135)
[...]
```

И теперь может быть проведен выбор автоматическая селекция моделей:

```
> dredge(модель, rank="QAIC", chat =1.126135)1
```

Тот же подход используется для моделей с квази-биномиальным законом распределения.

Примечание: для линейных смешанных моделей (LMM) полная исходная модель должна быть создана с параметром `REML=FALSE` (по умолчанию задается `REML=TRUE`). После того, как лучшие модели определятся, чтобы их проанализировать, они должен быть переформированы с параметром `REML=TRUE` (или просто не указывать параметр `REML`).

При использовании процедуры автоматического выбора следует знать, что модели с минимальным значением АИК (или его производных) – это не обязательно те, что имеют высокий биологически смысл. Поэтому не следует использовать эту процедуру слепо, а всегда думать о роли переменных и их взаимодействий, которые включены в модель. Таким образом, можно удалить некоторые члены полной модели вручную, если они не актуальны, или слишком сложны в интерпретации (например, взаимодействия порядка 3 или 4).

АЛЬТЕРНАТИВНЫЙ ОТКЛИК (0/1)

45. Согласие выборочной частоты и теоретической вероятности

Выбор соответствующего теста:

Выборки не связаны

Точный биномиальный тест (непараметрический). Используется функция: `binom.test(n1, n, p=теор.вер)`, где `n1` – это количество объектов, обладающих необходимым свойством, `n` – объем выборки и `теор.вер` – теоретическая вероятность анализируемой категории объектов (0.5 по умолчанию).

ПРИМЕР

Ставится задача проверить гипотезу, сбалансировано ли соотношение полов (и, следовательно, равная их вероятность 0.5) для выборки 20 экземпляров, которая включает 7 самок и 13 самцов:

```
> binom.test(7,20,p=0.5)
```

или более просто, так как `p=0.5` задается по умолчанию

```
> binom.test(7,20)
```

Связанные выборки

Тест Вальда (параметрический), Используется функция: `wald.ptheo.test(отклик, категория, p= теор.вер)*`, где `отклик` – случайная двоичная величина (вектор в цифровой форме или фактор) и `категория` – фактор, описывающий группировку объектов по уровням в том же порядке, что и отклик. Если отклик кодируется в виде 0/1, то проверяется вероятность того, что объект = 1; если отклик является фактором, проверяется вероятность 2-го уровня.

Функция строит модель логистической регрессии со смешанными эффектами, рассматривая `категория` как случайный фактор, и применяет тест Вальда, чтобы оценить H_0 о равенстве свободного члена регрессии теоретической вероятности. Если фактор `категория` не задан, то предпочтительнее точный биномиальный тест, тогда как тест Вальда основан на приблизительной аппроксимации.

46. Согласие нескольких частот теоретическим вероятностям

Условия теста: наблюдаемые численности объектов для каждой градации должны быть ненулевыми и 80 % из них должны быть ≥ 5 ("правило Кохрана", см. ниже).

Наблюдаемые численности рассчитываются с использованием функции: `chisq.bin.exp(отклик ~ фактор, p= теор.вер)*`, где `отклик` и `фактор` являются векторами, содержащие значения этих переменных для каждого объекта (в одном и том же порядке), и где `теор.вер` – это вектор, содержащий теоретической вероятности для каждой модальности (в порядке уровней фактора). Если отклик кодируется в виде 0/1, то проверяется вероятность того, что объект = 1; если отклик

является фактором, то проверяется вероятность 2-го уровня. Символ ~ означает "*установить соответствие*". Вычисленные наблюдаемые численности позволяют проверить, соблюдается ли правило Кохрана.

Примечание: эмпирические вероятности не зависят от процедуры группировки. Они не должны давать в сумме 1, так как уровни фактора не сравниваются друг с другом, а каждый из них проверяется на свою собственную теоретическую вероятность.

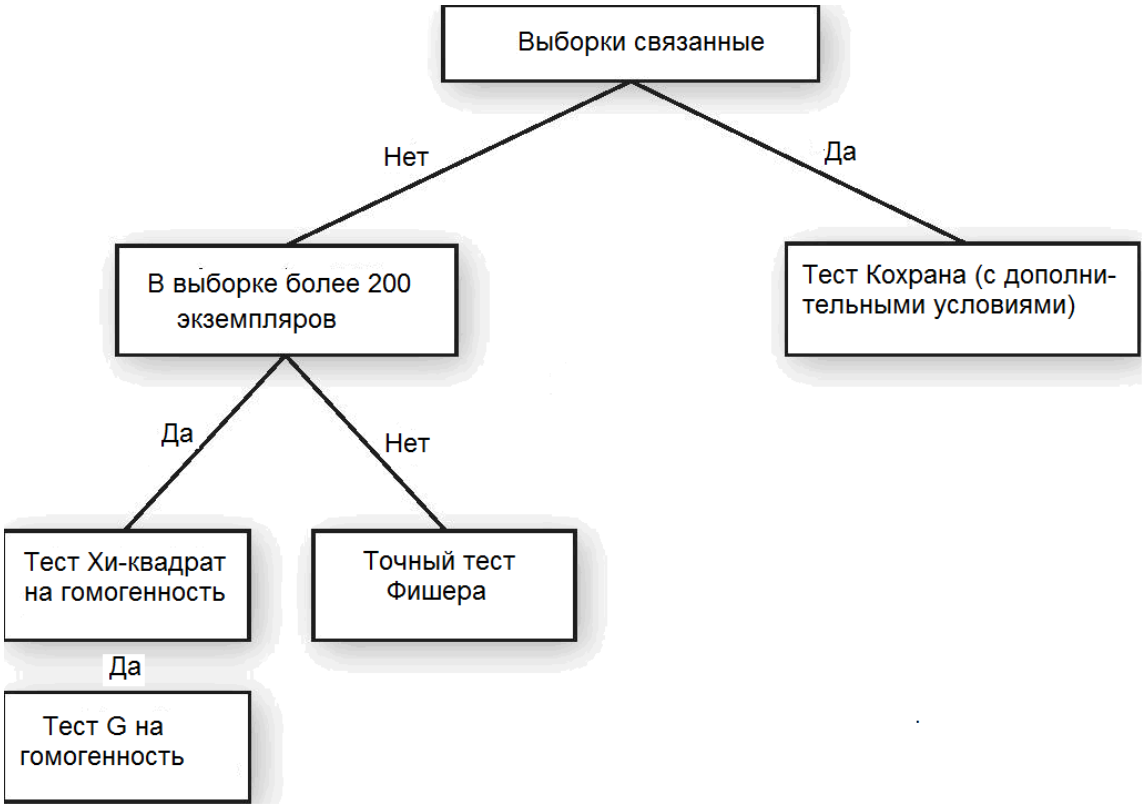
Критерий согласия χ^2 (непараметрический). Чтобы выполнить тест, используют функцию : `chisq.theo.bintest(отклик ~ фактор, p= теор.вер)*`.

Статистически значимое *p*-значение указывает, что, по крайней мере, одна эмпирическая численность отличается от теоретической вероятности (без указания, на каких уровнях фактора это имеет место). Чтобы определить различающиеся вероятности, необходимо выполнить множественные парные сравнения функцией: `prop.bin.multcomp(отклик ~ фактор, p= теор.вер)*`.

Может случиться так, что парные сравнения не показывают никаких существенных различий, в отличие от глобального теста. В этом случае, наиболее осторожное решение состоит в том, чтобы считать, что не можем знать, какие эмпирические частоты ответственны за отклонение от нулевой гипотезы в общем тесте.

47. Сравнение нескольких вероятностей объектов 2-х классов

Выбор соответствующего теста



Точный тест Фишера здесь всегда наиболее надежный, но время его расчета значительно увеличивается с ростом эмпирических численностей. Если общее количество экземпляров достаточно велико, то приближение, сделанное тестами χ^2 и G, является достаточно удовлетворительным, чтобы их можно было использовать.

Результаты этих двух тестов очень похожи; поэтому выбор между ними осуществляется скорее по привычке, чем по статистическим причинам.

В следующих тестах, если отклик кодируется в виде 0/1, то проверяется вероятность того, что объект = 1; если отклик является фактором, проверяется вероятность 2-го уровня.

Несвязанные выборки

Точный тест Фишера (непараметрический). Чтобы выполнить тест, можно использовать функцию: `fisher.bintest(отклик ~ фактор)*`, где `отклик` и `фактор` являются векторами, содержащими значения этих переменных для каждого объекта (в одном и том же порядке). Символ ~ означает "*установить соответствие*".

Статистически значимое *p*-значение указывает, что, по крайней мере, два уровня фактора имеют различное влияние на объясняемую переменную (но не ясно почему). Функция также выполняет автоматически все возможные парные сравнения по серии точных тестов Фишера.

Может случиться так, что парные сравнения не показывают никаких существенных различий, в отличие от глобального теста. В этом случае, наиболее осторожное решение состоит в том, чтобы считать, что не можем точно знать, какие эмпирические частоты ответственны за отклонение от нулевой гипотезы в общем тесте.

Тест χ^2 *на однородность* (непараметрический). Чтобы провести тест, используют функцию: `chisq.bintest(отклик ~ фактор)*`.

Если *p*-значение статистически значимо, то автоматически осуществляются парные сравнения с помощью серии тестов χ^2 на однородность.

Тест G *на однородность* (непараметрический). Чтобы провести тест, используют функцию : `G.bintest(отклик ~ фактор)*`.

Если *p*-значение статистически значимо, то автоматически осуществляются парные сравнения с помощью серии G-тестов на однородность.

Связанные выборки

Тест Q Кохрена (непараметрический)

Условия выполнения теста: план эксперимента должен быть составлен в виде полных случайных блоков без повтора (см. п. **13**).

Чтобы провести тест, используют функцию: `cochran.qtest(отклик ~ фактор.фиксированный|фактор.случайный)*`, где `фактор.фиксированный` – фактор, стационарные уровни которого участвуют в сравнении и `фактор.случайный` – фактор со случайными уровнями, который используется для обозначения различных серий. Эти обе переменных задаются векторами, содержащими значения, сопутствующими объясняемой переменной для каждого выборочного объекта (в одном и том же порядке).

Статистически значимое *p*-значение указывает, что, по крайней мере, два уровня фиксированного фактора имеют различное влияние на объясняемую переменную (но не ясно, почему). Функция выполняет автоматически все возможные парные сравнения по серии теста знаков Уилкоксона.

48. Анализ бинарного отклика с использованием моделей

Обсуждаемые здесь процедуры анализа используют основные понятия построения моделей, изложенные в п. **39- 42**.

Отклик

При использовании бинарного отклика случайная величина принимает одно из двух возможных значений. Однако обычно одна категория определяется условно как "группа интересов". Поэтому, если анализируется вероятность того, что объект принадлежит этой группе, то это трактуется как "вероятность отклика".

Объясняемая переменная может быть закодирована в цифровой форме (0/1) или быть фактором из двух уровней. В первом случае группу интереса принято обозначать цифрой 1; во втором – 2-м уровнем фактора.

Типы модели

В зависимости от того, связан ли отклик с другими факторами, определяющими некоторые условия эксперимента, используются модели разного типа. Если выборка не связана с сопутствующими факторами, то используются Обобщенные Линейные Модели (*Generalized Linear Model* или GLM). Напротив, если в наблюдениях следует учесть хотя бы один случайный фактор с нестационарными уровнями (см. п. **11**) , то используется Обобщенная Линейная Модель со Смешанными эффектами (*Generalized Linear Mixed Model* или GLMM), или просто "*смешанная модель*".

В отличие от ординарной линейной модели (или ее смешанной разновидности, см. далее п. **76**), GLM/ GLMM не основаны на нормальном законе распределения. В случае двоичного отклика обычно используется биномиальное распределение.

Примечание: существует несколько типов GLM/GLMM, основанных на биномиальном распределении. Наиболее частый случай, описанный здесь – это *логистические модели*. Если все независимые переменные являются количественными, то мы имеем дело с логистической регрессией.

Построение модели

Для создания модели используют следующие функции:

- без случайных факторов:

```
модель <-glm(формула, family="binomial")
```

- со смешанными эффектами:

`модель <-glmer(формула,family="binomial")` из пакета `lme4`.

Посмотрите п. **40** с подробным объяснением построения формулы.

В целом можно сказать, что:

- включение в модель фиксированного фактора позволяет проверить, с какой вероятностью различается реакция отклика в зависимости от уровней этого фактора.
- включение ковариаты позволяет проверить, существует связь между этой переменной и откликом.
- включение взаимодействия двух независимых переменных позволяет проверить, зависит ли один эффект от другого (это же рассуждение – для наблюдений, приуроченных к двум или более отрезкам времени). Особый случай – анализ взаимодействия между ковариатой и фактором, который позволяет проверить, зависит ли отношение между независимой переменной и откликом от уровней фактора.

Проверка адекватности модели

Прежде чем проводить дальнейший анализ, необходимо проверить, насколько хорошо модель подогнана к анализируемым данным. Этот этап является фундаментальным для любой модели, так как никакой тест, проводимый на недостаточно адекватной модели, просто ненадежен (или не соответствует истине). Смотрите п. **41** для подробного описания этой проверки.

Общая объясняющая сила

Для любой многомерной модели можно оценить объясняющую способность, используя коэффициент детерминации (R^2), который представляет собой долю дисперсии отклика, которая обуславливается независимыми переменными. Этот коэффициент рассчитывается функцией `r.squaredGLMM(модель)` из пакета `MuMIn`. Функция возвращает два значения: маргинальный R^2 (`R2m`), который соответствует доле дисперсии, обусловленной только фиксированными факторами и количественными переменными, и условный R^2 (`R2c`), который соответствует доле дисперсии, объясненной всей совокупностью переменных (включая случайные факторы). В случае GLM оба значения совпадают, поскольку случайный фактор отсутствует.

Тестирование

Независимо от модели, значимость влияния независимых переменных проверяется единой функцией: `Anova(модель)` из пакета `car` (см. п. **42** для подробного описания проверки гипотез). Однако этот тест для моделей имеет характерные особенности:

- для GLM функция выполняет анализ отношения максимального правдоподобия *(Likelihood Ratio Test* или LR Test) и тестирует отдельно каждый член модели (т. е. по строкам возвращаемый таблицы, содержащих оценки коэффициентов);
- для GLMM функция выполняет тест Вальда и анализирует отдельно каждый член модели (т. е. по строкам возвращаемый таблицы с оценками коэффициентов).

Если один фактор (или взаимодействие с участием фактора) имеет статистически значимый эффект, необходимо осуществить множественное парное сравнение его уровней (или сочетаний этих уровней), чтобы выявить, в каких комбинациях имеются различия. Посмотрите п. **43** как выполнить эти сравнения.

Если ковариата имеет значимое влияние, направление эффекта определяется знаком параметра. Полную информацию о всех параметрах модели можно получить с помощью команды `summary(модель)`. Значения самих коэффициентов называются `Estimate` и находятся в таблице `Coefficients` для GLM и `Fixed effects` для GLMM. Если знак коэффициента для количественной переменной отрицательный, то вероятность отклика уменьшается, когда значение ковриаты увеличивается; а если он положительный, то вероятность увеличивается, когда значение ковариаты увеличивается.

Прогноз на основе модели

Смысл моделей – не только показать роль различных независимых переменных в изменчивости объясняемой переменной, но и предсказать значение, которое принял бы отклик при известных значений независимых переменных. Поэтому, чтобы оценить вероятность отклика, требуется зафиксировать значение всех переменных.

Для прогнозирования могут быть использованы два метода (для GLMM доступен только второй), и оба они основаны на функции `predict()`:

- значение каждой из независимых переменных дается непосредственно в теле функции в виде списка (с одним или несколькими разделами): `predict(модель, newdata=list(переменные), type="response")`, где `переменные` – это последовательность `var1= значение,var2=значение` и т.д.
- создать таблицу, содержащую столбцы по каждой независимой переменной (имена столбцов должны строго совпадать с именами переменных модели), и заполнить каждую строку значениями, для которых должно быть сделано предсказание. Таким образом, делается прогноз по каждой строке таблицы: `predict(модель, newdata = таблица, type = "response")`.

Примечание: в случае GLMM некоторые значения случайных факторов могут не иметь данных. Если это не произошло, то необходимо добавить в функцию

predict() аргумент re.form=NA. Тогда при оценке прогноза будут приниматься, во внимание все эффекты случайных факторов модели.

ПРИМЕРЫ

Имеем модель, учитывающую фактор с двумя уровнями (A и B), ковариату ковариата в пределах от 0 до 30, и их взаимодействия:

```
> модель <- glm(отклик ~ фактор*ковариата, family="binomial")
```

Можно предсказать вероятность отклика таким образом :

```
> predict(модель, newdata= list(фактор=" A", ковариата=10),
                                    type="response").
```

Или для нескольких прогнозов :

```
> predict(модель, newdata=list(фактор=c("A","B"),
                    ковариата =c(10,10), type="response")
```

Или предварительно создать таблицу следующего вида:

```
> таблица
  Фактор ковариата
1      A        10
2      B        10
> predict(модель, newdata= таблица, type="response")
```

Графики

Влияние фактора. Для демонстрации эффекта влияния одного фактора строятся, как правило, гистограммы, где представлены вероятности отклика, средние для каждого уровня. Могут быть представлены два типа средних:

— средние брутто (т. е. вычисляемые по исходным данным) и их стандартные ошибки (см. пп. **35** и **37**, как использовать функцию tapply()). *Внимание*: в случае GLMM эти средние и стандартные ошибки не учитывают случайные факторы.

— средние значения, которые были скорректированы с учетом других переменных модели, а также их стандартные ошибки (см. п. **43**).

После вычисления средних и стандартных ошибок может быть построен и сам график (см. п. **34**).

Зависимость от ковариаты. Логистическая модель предполагает сигмоидальную (т. е. S-образную) зависимость между объясняемой переменной и каждой из независимых количественных переменных. Чтобы проиллюстрировать эту связь, необходимо выполнить три действия :

1. Представить на графике точки наблюдений plot(отклик ~ ковариата).

2. Создать вектор, имеющий те же минимум и максимум, что ковариата, но с очень маленьким интервалом между его значениями x <- seq2(ковариата)*.

3. Добавить на график кривую модельных значений.

Определение всех точек модельной кривой основано на прогнозе значений, которые принимает объясняемая переменная для каждого значения вектора x. Большое количество значений этого вектора, и очень маленький интервал, который их отделяет, определяет гладкость кривой (которая на самом деле состоит из сотен точек, соединенных между собой). Если модель содержит другие независимые переменные, то к прогнозу необходимо добавить одинаковые значения, такие, чтобы вероятность отклика изменяла свою величину только при изменении анализируемой ковариаты. Кривая на график может быть добавлена функцией
lines(x, predict(модель, newdata= x, type="response")).

Примечание: функция lines() может быть использована несколько раз, чтобы начертить несколько кривых на одном графике, например, для нескольких уровней одного и того же фактора, включенного в модель. Чтобы изменить тип линий графика и цвет кривых, можно использовать аргументы lty и col соответственно. Наконец,

чтобы добавить заголовки, используют функцию `legend()`. Смотрите справку `?` для получения дополнительных сведений об используемых параметрах и знакомства с различными вариантами графики.

ПРИМЕРЫ

Рассмотрим следующую модель :

```
> модель <- glm(отклик ~ фактор*ковариата,family="binomial")
```

Шаг 1: нарисуем точки, соответствующие наблюдаемым данным

```
> plot(отклик ~ ковариата)
```

Шаг 2: создадим вектор `x`:

```
> x <- seq2(ковариата)
```

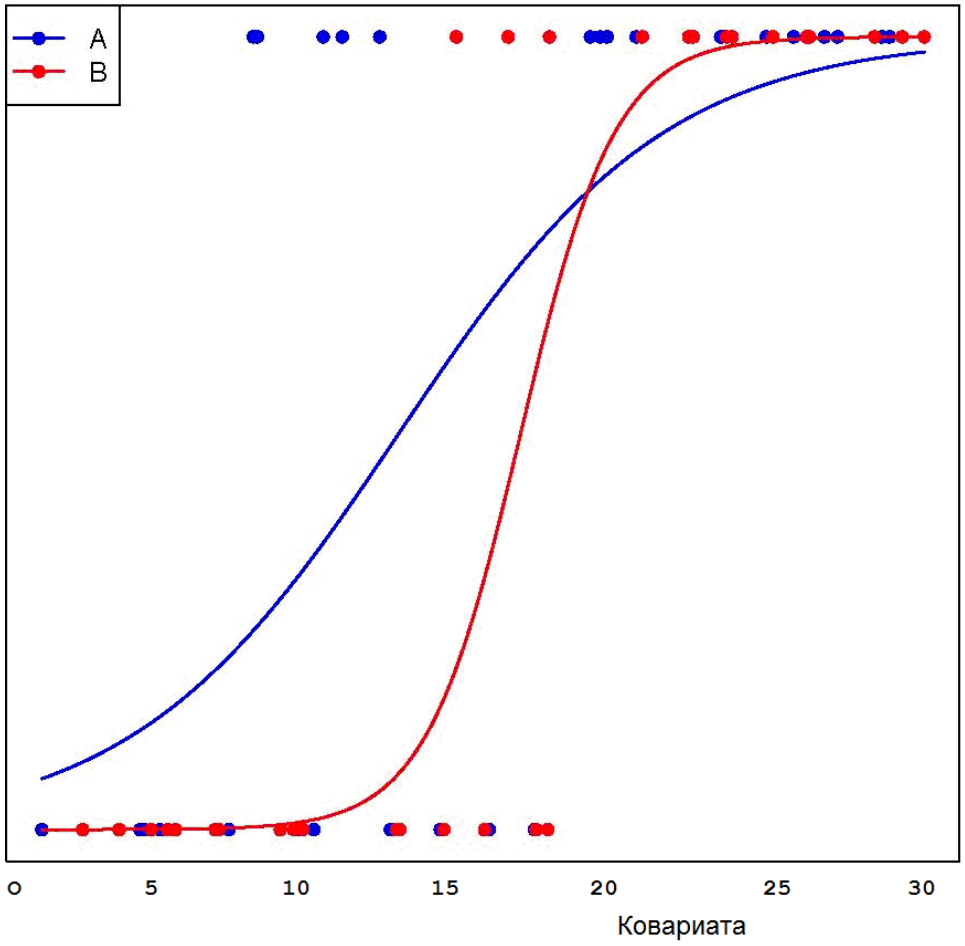
Шаг 3: добавляем модельную кривую. Мы выбираем один из уровней фактора, включенного в модель. Функция `rep()` является очень полезной в этой ситуации, поскольку она позволяет очень просто создать вектор той же длины, что и `x`, содержащий одно и тоже повторяющееся значение `A`

```
> lines(x, predict(модель, newdata=list(ковариата = x,
      фактор =rep (A,length(x))), type="response"), col = 2)
```

Чтобы добавить модельную кривую, соответствующую уровню `B` фактора :

```
> lines(x, predict(модель, newdata=list(ковариата = x,
          фактор =rep (B, length(x))), type="response"))
```

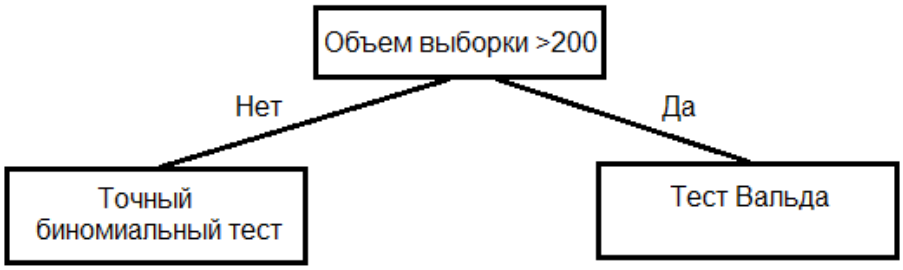
На полученном рисунке синим цветом представлена модельная кривая логистической регрессии для уровня `A` воздействующего фактора, а красным цветом – уровня `B`.



НОМИНАЛЬНЫЙ ОТКЛИК С БОЛЕЕ ЧЕМ 2 КЛАССАМИ

49. Сравнение эмпирических вероятностей с теоретическими значениями – более чем 2 класса

Выбор соответствующего теста



Точный биномиальный тест здесь всегда наиболее надежен, но время его расчета значительно увеличивается с ростом эмпирических численностей. Если количество экземпляров достаточно велико, то приближение, сделанное тестом Вальда, является достаточно удовлетворительным, чтобы его можно было практически использовать.

Точный биномиальный тест (непараметрический).
Чтобы провести тест, можно использовать функцию: `multinomial.theo.multcomp(отклик, p=теор.вер, prop=TRUE)*`, где `отклик` – фактор, определяющий для каждого объекта один из трех (по крайней мере) возможных уровней, и `теор.вер` – вектор теоретических вероятностей для каждого уровня (в том же порядке). Сумма этих вероятностей должна быть равна 1.

Тест Вальда (параметрический), Используется функция: `wald.ptheo.multinom.test (отклик, p= теор.вер)*`.

50. Сравнение нескольких вероятностей при более чем 2 классах

Тест Вальда (параметрический). Для выполнения теста используем функцию: `prop.multinom.test(отклик)*`, где `отклик` – определяющий для каждого объекта один из трех (или более) возможных уровней.

В этом тесте не делается никаких глобальных выводов. Функция `prop.multinom.test()` фактически выполняет непосредственно множественные сравнения между разными уровнями отклика (т. е. сравниваются между собой соответствующие вероятности этих классов).

ПРИМЕР

Предположим, проведен эксперимент, в котором каждый испытуемый выбирал между тремя вариантами Opt1, Opt2, Opt3. Получены следующие результаты :

```
> результат <- factor(c("Opt1","Opt2","Opt1","Opt2","Opt2",
          "Opt1","Opt3","Opt2","Opt1","Opt2","Opt2","Opt3",
          "Opt2","Opt2","Opt2","Opt1","Opt3","Opt2","Opt1",
          "Opt2","Opt1","Opt2","Opt1","Opt2","Opt2"))
```

Вероятность каждого класса (и стандартные ошибки):

```
>    prop.multinom(результат)
```

```
$probs
Opt1 Opt2 Opt3
0.32 0.56 0.12

$se
  Opt1   Opt2   Opt3
0.0952 0.1013 0.0663
```

Теперь сравниваем эти вероятности:

```
> prop.multinom.test(результат)
```

51. Основанный на моделях анализ номинального отклика с более чем 2 классами

Обсуждаемые здесь процедуры анализа используют основные понятия построения моделей, изложенные в п. **39- 42**.

Отклик

Номинальный отклик представляет собой случайную переменную, каждая реализация которой может принимать одно значение из трех или более неупорядоченных категорий. Таким образом, речь идет о том, чтобы проанализировать набор вероятностей принадлежности объекта к каждому из K классов $\{\pi_1, ..., \pi_K\}$, $\sum_K \pi_K = 1$. Это серьезно усложняет интерпретацию результатов, по сравнению с анализом альтернатив (см. анализ двоичного отклика, п. **48**).

Для анализа такого отклика, необходимо определить "*базовый класс*" – для определенности примем, что это класс 1. Работа модели основана на том, чтобы рассчитать соотношения вероятностей π_i/π_1 и отнести объект к тому классу, для которого это соотношение максимально. Потому для K классов необходимо проанализировать K - 1 частных моделей (т.е. оценить именно столько параметров модели для каждой независимой переменной!), и, наконец, интерпретировать результаты для всех возможных комбинаций $K(K–1)/2$ парных сравнений. Очевидна сложность работы с такими откликами. Поэтому ограничиваются, по возможности, максимально простым планом эксперимента, т. е. несколько независимых переменных, учет только очевидных взаимодействий и минимум количества классов отклика. По аналогичным причинам лучше избегать использовать факторы с более чем двумя уровнями в качестве независимых переменных (или тогда надо быть готовым, что интерпретация результатов может быть весьма утомительной).

В **R** объясняемая переменная должна быть фактором с более чем двумя уровнями. По умолчанию 1-й уровень определяется как базовый класс. Выбор этого класса не влияет на результаты анализа.

Типы модели

Для моделирования отклика используется *мультиномиальная модель* (или "политомичесая неупорядоченная"), которая является расширением обобщенной линейной модели (GLM) для отклика в двоичном формате (см. п. **48**).

Примечание: существует несколько типов мультиномиальных моделей. Мы рассматриваем здесь наиболее частый случай – логистическую модель. Если все независимые переменные являются количественными, то мы имеем мультиномиальную логистическую регрессию.

Построение модели

Для создания модели используют функцию из пакета `multinom`:
`модель<-multinom(формула,abstol=1e-15,reltol=1e-15,maxit=1000)`.
См. п. **40** для подробного объяснения построения формул. Аргументы `abstol`, `reltol` и `maxit` - параметры, обеспечивающие более высокую точность результатов.

Примечание: построение мультиномиальная модели функцией `multinom()` – итерационный процесс (как это очень часто), ход которого может отображаться на экране. Чтобы удалить все сообщения при построении модели, необходимо добавить аргумент `trace=FALSE`.

Проверка адекватности модели

Поскольку объясняемая переменная не является количественной, проверка модели обычным образом здесь не выполняется. Однако мы часто хотим реально знать, насколько оцененные параметры модели зависят от состава обрабатываемых данных. Если параметры очень чувствительны, то малейшее изменение в данных приводит к тому, что модель становится вырожденной. Это может означать (среди прочего), что некоторые параметры непредсказуемы и модель может быть упрощена.

Значение, которое может оценить эту тенденцию – обусловленность матрицы Гессе. Ее расчет осуществляется с помощью функции `cond.multinom(модель)*`. Здесь, в действительности, нет какого-либо критического абсолютного порога, но обычно считается, что, если обусловленность превышает 10^5 , то это является признаком того, что модель склонна к вырождению.

Тестирование

Влияние независимых переменных проверяется с помощью дисперсионного анализа `Anova(модель)` из пакета `car` (см. п. **42** с подробным описанием проверки гипотез). Функция выполняет анализ отношения максимального правдоподобия (*Likelihood Ratio Test* или LR Test) и тестирует отдельно каждый член модели (т. е. по строкам возвращаемый таблицы с оценками коэффициентов).

Если независимая переменная оказывается статистически значимой, это означает, что она оказывает влияние на соотношение вероятностей, по крайней мере, для двух классов переменной отклика. Чтобы определить, на какие именно уровни отклика оказывается значительное влияние, используют функцию: `test.multinom (модель, переменная)*`, где `переменная` является независимой переменной, для которой требуется изучить детальное влияние.

Результат, возвращаемый функцией `test.multinom()`, зависит от природы независимой переменной:

1. Для количественной независимой переменной функция возвращает массив, где каждая строка соответствует отношению между двумя классами переменной отклика (синтаксис A|B означает "*вероятность* A *по сравнению с вероятностью* B"). Знак коэффициента показывает направление зависимости между категорией отклика и независимой переменной. Отношение шансов (*Odds ratio*) – простой способ интерпретации результата, который показывает, насколько меняется соотношение вероятностей при увеличении на единицу независимой переменной.

Примечание: не удивителен тот факт, что частные *p*-значения этого теста будут несколько отличаться от общих *p*-значений, полученных с помощью `Anova(модель)`. Во-первых, общий тест не сводится к сумме отдельных тестов, а, во-вторых, это разные тесты: на глобальном уровне оценивается максимальное правдоподобие, а на частном уровне осуществляется тест Вальда. Второй тест менее мощный, чем первый (см. п. **17**).

ПРИМЕР. Для модели, где отклик имеет три класса (A/B/C), ковариата имеет значимое влияние. Подробно ее влияние состоит в следующем :

```
       Coeff      SE Odds.ratio       z Pr(>|z|)
A|C -0.93509 0.57383     0.3926 -1.6296  0.10319
B|C  1.56595 1.05528     4.7872  1.4839  0.13783
B|A  2.50104 1.20247    12.1952  2.0799  0.03753 *
--
Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Вывод: ковариата влияет только на соотношение между условиями A и B (*p*-значение меньше 0.05). `Odds ratio` показывает, что вероятность того, что объект имеет категорию B, а не A, увеличивается-в 12.2 раза, если значение ковариаты увеличивается на единицу.

Примечание: значимость отличий коэффициента от 0 равносильна значимости отличий отношения шансов от 1.

2. Если независимая переменная – фактор с двумя уровнями, то функция возвращает таблицу, структурированную как для ковариаты. Отношения шансов и значения оценок коэффициентов при соотношении вероятностей указываются как для одного, так и для другого уровня независимого фактора, который определен в заголовке таблицы.

ПРИМЕР. Для модели, где отклик имеет три класса (A/B/C), фактор при двух уровнях (**`Самцы|Самки`**) имеет значимое влияние. Подробности этого влияния выглядят в протоколе вычислений следующим образом :

```
$‘Самцы|Самки’
      Coeff     SE Odds.ratio       z Pr(>|z|)
A|C  0.1702 1.8533    1.18550  0.09182  0.92684
B|C -1.9660 0.8645    0.14002 -2.27415  0.02296 *
B|A -5.1361 4.0226    0.00588 -1.27682  0.20166
--
Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Вывод: фактор влияет на соотношение вероятностей между условиями B и C. `Odds ratio` показывает, что вероятность того, что самцы определяют категорию B, а не C только в 0.14 раз больше, чем самки (т. е. самки имеют в 1/0.14 = 7.14 раз больше шансов быть B, а не C по сравнению с самцами).

3. Если независимая переменная – фактор с более чем двумя уровнями, то функция возвращает таблицу для каждого уровня этой переменной. Очевидно, что эта ситуация – просто расширение предыдущего случая, который труднее интерпретировать.

Прогноз на основе модели

Смысл моделей – не только показать роль различных независимых переменных в изменчивости объясняемой переменной, но и предсказать значение, которое бы принял отклик при известных значений независимых переменных. Поэтому, чтобы оценить набор вероятностей отклика, требуется зафиксировать значение всех переменных.

Для прогнозирования могут быть использованы два метода, и оба они основаны на функции `predict()`:

- значение каждой из независимых переменных дается непосредственно в теле функции в виде списка (с одной или несколькими записями): `predict(модель,newdata=list(переменные))`, где `переменные` – это последовательность `var1= значение,var2=значение` и т.д.;

- создать таблицу, содержащую столбцы по каждой независимой переменной (имена столбцов должны строго совпадать с именами переменных модели), и заполнить каждую строку значениями, для которых должно быть сделано предсказание. Таким образом, делается прогноз по каждой строке таблицы: `predict(модель, newdata = таблица)`.

Функция `predict()` возвращает для каждого прогноза метку класса, наиболее вероятного для переменной отклика, с учетом значений независимых переменных, которые задавались для прогнозирования. Чтобы получить детализацию вероятностей для каждого из классов переменной отклика, необходимо добавить аргумент `type="probs"`. Результат – обычно матрица, в которой каждая строка – это прогноз, а каждый столбец соответствует классу переменной отклика.

ПРИМЕРЫ

Имеем модель, содержащую `фактор` с двумя уровнями (`A` и `B`), ковариату `ковариата` в пределах от 0 до 30, и их взаимодействия :

```
> модель <- multinom(отклик ~ фактор*ковариата,
                abstol=1e-15,reltol=1e-15,maxit=1000)
```

Можно предсказать вероятность отклика таким образом :

```
> predict(модель,newdata= list(фактор=" A", ковариата =10),
                                        type="response").
```

Или для нескольких прогнозов :

```
> predict(модель, newdata=list(фактор =c("A","B"),
                      ковариата =c(10,10), type="response")
```

Или предварительно создать таблицу следующего вида:

```
> таблица
  Фактор ковариата
1      A        10
2      B        10
> predict(модель, newdata= таблица, type="response")
```

ОТКЛИК – ПОРЯДКОВАЯ ПЕРЕМЕННАЯ

Для анализа таких переменных лучше избегать использования простых непараметрических тестов, основанных на рангах (Манна-Уитни-Вилкоксона, Крускала-Уоллиса, Фридмана и т.д.). Если порядковая переменная содержит большое число идентичных значений (иногда фиксируется лишь несколько значений при работе с рейтинговыми данными), то это полностью искажает результаты ранговых тестов.

52. Анализ порядковых переменных, основанный на моделях

Обсуждаемые здесь процедуры анализа используют основные понятия построения моделей, изложенные в п. **39- 42**.

Отклик

Порядковая переменная обычно отражает место объекта в упорядоченной последовательности (рейтинге), тем более, если она кодируется в цифровой форме (т. е. 1/2/3/...). Можно было бы назвать ее количественной переменной, но это не так, поскольку это – качественная переменная, имеющая порядковый номер (см. п. **11**). Действительно, запись (1/2/3) вполне может быть заменена на буквы (A/B/C) или фразы (хорошо/средне/плохо), без изменения смысла самой переменной. Иногда такие переменные называют "полуколичественными", поскольку они предполагают возможность количественного упорядочивания, но сами по себе количественными не являются. Именно по этой причине параметр положения (см. п. **35**) таких случайных величин оценивается как медиана, а не среднее.

Значения переменной размещаются в объекте особого типа, в частности, факторе со специальной организацией уровней. Чтобы создать этот объект, используют функцию:
`фактор <- factor(выборка, levels=классификатор, ordered=TRUE)`,
где `выборка` – исходный набор данных и `классификатор` – значения градаций шкалы рейтинга в порядке возрастания (в кавычках).

ПРИМЕРЫ

Если градации кодируются в цифровом выражении 1/2/3/4, где 1 соответствует высшему рангу:

```
> фактор <- factor(выборка, levels=c("4","3","2","1"),
                                          ordered=TRUE)
```

При выводе на экран вектора созданного фактора, его уровни даются в такой форме:

```
Levels: 4 < 3 < 2 < 1
```

Если градации кодируются выражениями хорошее/среднее/плохое ("хорошее" соответствует максимальному рангу) :

```
> переменная <- factor(выборка,
        levels=c("плохой","средний","хороший"), ordered=TRUE)
```

При выводе на экран вектора созданного фактора, его уровни даются в такой форме:

```
Levels: плохой < средний < хороший
```

Типы модели

Хотя принципы построения моделей остаются одними и теми же, некоторые технические аспекты анализа отличаются в зависимости от того, связан ли отклик с другими факторами, определяющими некоторые важные для исследования условия эксперимента. Если связанный фактор отсутствует, то используется модель пропорционального риска (*Proportional Odds Model* или POM). Если выборка исходных данных включает, по крайней мере, один связанный случайный фактор, то используют модель пропорционального риска со смешанными эффектами (*Proportional Odds Mixed Model* или POMM).

Эти модели являются своего рода расширением логистических моделей, позволяющих проанализировать отклик в двоичном формате (см. п. **48**), однако не вписываются в семейство обобщенных линейных моделей (GLM). В случае порядкового отклика смысл анализа состоит в том, чтобы смоделировать процесс накопления вероятностей в ряду градаций от первого до последнего уровня и для каждого перехода в этом ряду рассчитать отношение этих вероятностей.

Примечание: POM/POMM на самом деле являются частным случаем моделей кумулятивных ссылок CLM/CLMM (*Cumulative Link (Mixed) Model*). Мы рассматриваем одну из версий этих моделей – кумулятивный логит (*Cumulative Logit*).

Построение модели

Для создания модели используют набор функций из пакета `ordinal`:

- без связанных случайных факторов `модель<- clm(формула);`
- с связанными качественными переменными `модель<-clmm(формула).`

См. п. **40** для подробного объяснения построения формулы. В целом можно напомнить, что:

- включение фактора в качестве независимой переменной позволяет проверить, различаются ли градации отклика между уровнями этого фактора;
- включение ковариаты позволяет проверить, существует связь между этой ковариатой и объясняемой переменной;
- включение взаимодействия двух независимых переменных позволяет проверить, зависит ли один эффект от другого (это же рассуждение – для наблюдений, приуроченных к двум или более отрезкам времени). Особый случай – анализ взаимодействия между ковариатой и фактором, который позволяет проверить, зависит ли отношение между независимой переменной и откликом от уровней фактора.

Проверка адекватности модели

Поскольку объясняемая переменная не является количественной, проверка модели обычным образом здесь не выполняется. Однако мы часто хотим реально знать, насколько оцененные параметры модели зависят от состава обрабатываемых данных. Если параметры очень чувствительны, то малейшее изменение в данных приводит к тому, что модель становится вырожденной. Это может означать (среди прочего), что некоторые параметры непредсказуемы и модель может быть упрощена.

Значение, которое может оценить эту тенденцию – обусловленность матрицы Гессе. Она представлена в объекте `модель` значением `cond.H`. Для ее значения нет абсолютного критического порога, но обычно считается, что, если обусловленность превышает 10^5, то это является признаком того, что модель вырождена.

Тестирование

Влияние независимых переменных проверяется с помощью дисперсионного анализа `Anova(модель)` из пакета `car` (см. п. **42** с подробным описанием проверки гипотез). Функция выполняет анализ отношения максимального правдоподобия (*Likelihood Ratio Test* или LR Test) и тестирует отдельно каждый член модели (т.е. по строкам возвращаемый таблицы с оценками коэффициентов).

Примечание: для тестирования модели, созданной `clm()` или `clmm()`, с использованием функции `Anova()` необходимо загрузить пакет `RVAideMemoire`.

Если один фактор (или взаимодействие с участием фактора) имеет значимый эффект, необходимо осуществить множественные сравнения, чтобы определить, какими конкретно уровнями или их комбинациями эти отличия определяются. См. п. **43** с методикой проведения этих сравнений.

Если ковариата имеет значимое влияние, направление эффекта определяется знаком параметра. Полную информацию обо всех параметрах модели можно получить с

помощью команды `summary(модель)`. Значения самих коэффициентов называются `Estimate` и находятся в таблице `Coefficients`. Если знак коэффициента для количественной переменной отрицательный, то вероятность отклика уменьшается, когда значение ковриаты увеличивается; а если он положительный, то вероятность увеличивается, когда значение ковариаты увеличивается.

Прогноз на основе модели

Смысл моделей – не только показать роль различных независимых переменных в изменчивости объясняемой переменной, но и предсказать значение, которое принял бы отклик при известных значений независимых переменных. Поэтому, чтобы оценить набор вероятностей отклика, требуется зафиксировать значение всех переменных.

Для прогнозирования могут быть использованы два метода, и оба они основаны на функции `predict()`:

- значение каждая из независимых переменных дается непосредственно в теле функции, в виде списка (с одной или несколькими записями): `predict(модель,newdata=list(переменные))`, где `переменные` – это последовательность `var1= значение, var2=значение` и т.д.;
- создать таблицу, содержащую столбцы по каждой независимой переменной (имена столбцов должны строго совпадать с именами переменных модели), и заполнить каждую строку значениями, для которых должно быть сделано предсказание. Таким образом, делается прогноз по каждой строке таблицы: `predict(модель, newdata = таблица)`.

Специфика POM(M) (и, в частности, CLM(M)) состоит в том, что они моделируют несколько вариантов значений отклика одновременно, т.е. вероятности или их соотношения, связанные с каждым уровнем рейтинга. Таким образом, функция `predict()` для каждого прогноза возвращает столько значений прогнозируемых вероятностей, сколько установлено этих уровней, и с учетом величин независимых переменных, которые задавались для прогнозирования. Списокк прогнозов, возвращаемых `predict()`, реорганизуется в матрицу, каждая строка которой – прогноз, а каждый столбец – оценки вероятности каждой градации (снизу вверх по рейтингу). Возможны два других вида представления результатов прогнозирования:

- если назначить для функции `predict()` аргумент `type="cum.prob"`, то возвращаются накопленные вероятности прогноза. В этом случае также выводится таблица с прогнозами в строках и градациями в столбцах. Однако каждый столбец содержит уже не вероятности для соответствующего уровня, а вероятность того, что прогнозируемый уровень ниже или равен каждой текущей градации. Поэтому последний столбец (т.е. наивысшая оценка в рейтинге), содержит обязательно значение 1.
- если назначить для функции `predict()` аргумент `type="class"`, то возвращаются наиболее вероятные оценки. Функция формирует на этот раз вектор, содержащий метки прогнозируемых уровней с учетом заданных значений независимых переменных.

ПРИМЕРЫ

Имеем модель, содержащую `фактор` с двумя уровнями (A и B), ковариату `ковариата` в пределах от 0 до 30 и их взаимодействия :

```
> модель <- clm(отклик ~ фактор * ковариата)
```

Можно предсказать вероятность ответа таким образом :

```
> predict(модели,newdata= list (фактор=" A", ковариата =10),
                                    type="response"),
```

Или для нескольких прогнозов :

```
> predict(модель,newdata=list(фактор =c("A","B"),
                    ковариата =c(10,10), type="response")
```

Или предварительно создать таблицу следующего вида:

```
> таблица
  Фактор ковариата
1      A        10
2      B        10
> predict(модель, newdata= таблица, type="response")
```

Графики

Влияние фактора. Для того чтобы проиллюстрировать влияние одного фактора, строится, как правило, гистограмма (см. п. **34**). Но из-за сложности объясняемой переменной приходится формировать несколько графиков. Также имеет значение выбор той информации, которую мы хотим представлять: эмпирическая частота, наблюдаемая для различных градаций в наборе данных или вероятность того, что будет назначен этот уровень, исходя из подогнанной модели. Часто более интересно представлять скорректированные вероятности, потому что они демонстрируют долю объясненной вариации анализируемого фактора с учетом влияния остальных независимых переменных в модели. Более того, только скорректированные вероятности учитывают влияние случайных факторов (если они есть в анализе).

Можно рассчитать и представить на графиках следующие показатели:

А) *Частоты / вероятности каждой градации*.

- Эмпирические частоты наблюдений могут быть получены с помощью функции `rating.prob(отклик, фактор)*` (фактор может быть также взаимодействием между двумя факторами, например, `фактор1:фактор2`).
- Для получения скорректированных вероятностей необходимы два шага. Сначала нужно вычислить эти вероятности и хранить их в объекте типа LSM: `LSM <- lsmeans(модель,~фактор|cut, mode="linear.predictor")` (см. п. **43** для объяснения синтаксиса функции `lsmeans()` из пакета `lsmeans`; важно, чтобы формула заканчивалась `|cut`). Потом нужно восстановить вероятности, которые не представлены непосредственно функцией `lsmeans()`, для чего можно воспользоваться функцией `rating.lsmeans(LSM)*`.

Б) *Совокупные частоты / вероятности*, накопленные каждой градацией (т.е. вероятность того, что оценка отклика будет меньше или равна каждой градации, считая снизу вверх по рейтингу).

- Накопленные частоты наблюдений могут быть получены с помощью функции `rating.prob(отклик,фактор,тип="cumprob")*`.
- Скорректированные и восстановленные вероятности (см. объект `LSM` в п. **43**) могут быть получены с помощью функции `rating.lsmeans(LSM, type="cumprob")*`.

В) *Наиболее распространенная / вероятная градация*. Гистограмма в этом случае не может быть использована, поскольку отклик имеет качественный, а не количественный характер. Частотная оценка самой распространенной градации может быть получена с помощью функции `rating.prob(отклик, фактор, тип="class")*`, а вероятностная оценка, основанная на объекте `LSM` – с использованием функции `rating.lsmeans(LSM, type="class")*`.

ОТКЛИК – СЧЕТНЫЕ ДАННЫЕ

Подсчет объектов без разделения на категории

Этот тип отклика в простых тестах часто называют "эффектом". Но на самом деле "эффект" не сводится только к подсчету числа индивидуумов.

53. Соответствие совокупностей счетных данных теоретическому распределению

Так как серии фактически наблюдаемых счетных данных редко полностью соответствуют постулируемому теоретическому закону распределения, необходимо оценить уровень их согласованности. Проверка степени приближения осуществляется, прежде всего, графически, где всегда присутствует доля субъективности, которой лишены статистические тесты.

Графический тест

Визуально степень согласия эмпирического и теоретического распределений оценивается с помощью квантиль-квантильного графика, который формируется функцией `qqPlot(серия,dist="loi",par)` из пакета `car`, где `серия` является вектором, содержащим ряд данных, `loi` обозначает выбранный теоретический закон (в кавычках) и `par` – его параметры, разделенные запятой (см. п. **19-22**).

ПРИМЕРЫ

Приближение биномиальным законом (см. п. **20**):

```
>    qqPlot(серия,dist="binom",n,p)
```

Приближение распределением Пуассона (см. п. **21**):

```
> qqPlot(серия,dist="pois",lambda)
```

Распределение выборки согласуется с выбранным теоретическим распределением, если точки графика располагаются примерно в районе главной диагонали. Любое другое расположение точек (кривизна, много удаленных точек и т.д.) указывает на обратное. Точки могут быть не полностью выровнены относительно прямой, но, если они остаются в доверительном интервале, то приближение считается корректным.

Статистический тест

Тест Крамера – фон Мизеса (непараметрический) реализуется с использованием пакетов `dgof` и `RVAideMemoire` следующим образом:

```
cvm.test(серия, cdf.discrete(серия,"loi",par)).
```

ПРИМЕРЫ

Приближение биномиальным законом (см. п. **20**):

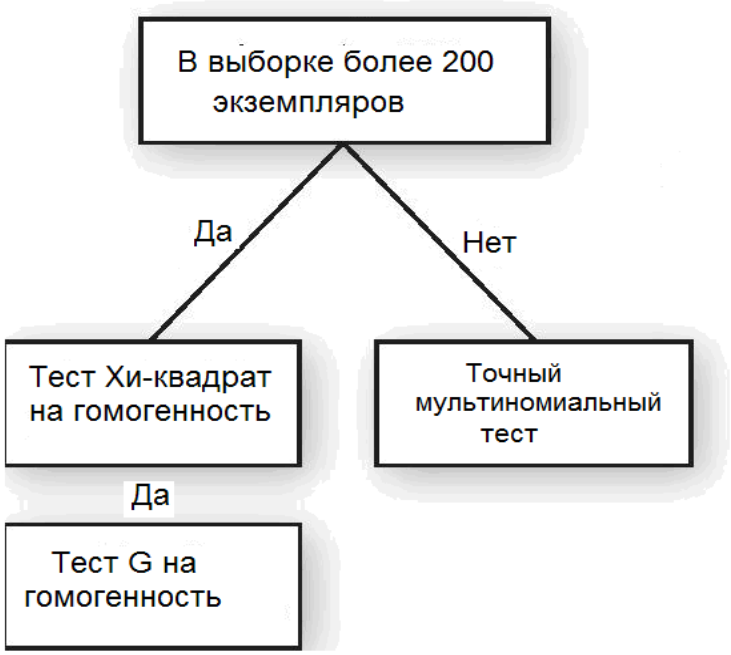
```
>  cvm.test(серия, cdf.discrete(серия, "binom", n, p))
```

Приближение распределением Пуассона (см. п. **21**):

```
>  cvm.test(серия, cdf.discrete(серия, "pois", lambda))
```

54. Проверка однородности счетных данных

Выбор соответствующего теста



Точный мультиномиальный тест по-прежнему является самым надежным, но время его вычисления значительно увеличивается с количеством объектов. Если общее количество экземпляров достаточно велико, то приближение, сделанное тестами χ^2 и G, является достаточно удовлетворительным, чтобы их можно было использовать. Результаты этих двух тестов очень похожи; поэтому выбор между ними осуществляется скорее по привычке, чем по статистическим причинам.

Точный мультиномиальный тест (непараметрический). Чтобы выполнить тест, можно использовать функцию: `multinomial.test(численность)*`, где `численность` – вектор счетных данных.

Статистически значимое *p*-значение указывает, что, по крайней мере, два счетных значения отличаются друг от друга, не уточняя, какие из них. В этом случае необходимо провести парные сравнения для определения конкретных наблюдений, о которых идет речь, с использованием функции `multinomial.multcomp(численность)*`.

Может случиться так, что парные сравнения не показывают никаких существенных различий, в отличие от глобального теста. В этом случае, наиболее разумным решением будет считать, что мы не можем знать, какие конкретно счетные выборочные значения привели к отказу от нулевой гипотезы в общем тесте.

Тест χ^2 на однородность (непараметрический), Чтобы провести тест, используют функцию `chisq.test(численность)`.

Если *p*-значение статистически значимо, то можно выполнить парные сравнения с помощью `chisq.multcomp(численность)*`.

Тест G *на однородность* (непараметрический), Чтобы провести тест, используют функцию : `G.test(численность)*`.

Если *p*-значение статистически значимо, то можно выполнить парные сравнения с помощью `G.multcomp(численность,p=prop.theo)*`.

55. Анализ счетных данных, основанный на моделях

Обсуждаемые здесь процедуры анализа используют основные понятия построения моделей, изложенные в п. **39- 42**.

Отклик

Отклик может быть только произвольной целочисленной положительной переменной.

Типы модели

Хотя основные принципы остаются одними и теми же, некоторые технические аспекты анализа отличаются в зависимости от того, связан ли отклик с другими факторами, определяющими некоторые важные для исследования условия эксперимента. Если связанный фактор отсутствует, то используется Обобщенная Линейная Модель (*Generalized Linear Model* или GLM), в то время, как при наличии связанных факторов – Смешанная Обобщенная Линейная Модель (*Generalized Linear Mixed Model* или GLMM). Термин "смешанный" подразумевает наличие, по крайней мере, одного случайного фактора, которые используются чтобы точно идентифицировать серии наблюдений (см. п. **11**).

В отличие от обыкновенной Линейной Модели (или ее смешанного варианта, см. далее п. **76**), GLM- или GLMM-модели не основаны на нормальном законе распределения. В случае счетных данных обычно априори используют распределение Пуассона.

Примечание: на самом деле существуют несколько типов GLM, основанных на распределении Пуассона. Здесь описан наиболее часто встречающийся вариант – *лог-линейная модель*. Если все объясняющие переменные являются числовыми величинами (ковариатами), мы имеем частный случай *лог-линейной регрессии* или *регрессии Пуассона*.

Построение модели

Для создания модели используют следующие функции:

- без связанных случайных факторов
  `модель <- glm(формула, family="poisson")`
- со связанными качественными случайными переменными:
  `модель <- glmer(формула, family="poisson")` из пакета `lme4`.

См. п. **40** для подробного объяснения построения формулы. В целом можно напомнить, что:

- включение независимого фактора позволяет проверить, различаются ли градации отклика между уровнями этого фактора;
- включение ковариаты позволяет проверить, существует связь между этой ковариатой и объясняемой переменной;
- включение взаимодействия двух независимых переменных позволяет проверить, зависит ли один эффект от другого (это же рассуждение также для наблюдений, приуроченных к двум или более отрезкам времени). Особый случай – анализ взаимодействия между ковариатой и фактором, который позволяет проверить, зависит ли отношение между независимой переменной и откликом от уровней фактора.

Проверка адекватности модели

Избыточная дисперсия остатков. Первый этап валидации модели состоит в том, чтобы проверить, не содержат ли остатки избыточной дисперсии (*overdispersion*), т.е. их вариация не превышает ожидаемую, исходя из свойств распределения Пуассона. Для этого достаточно сравнить отклонение остатков (*residual deviance*) модели со

степенями свободы для остатков (*residual degrees of freedom*). Если это отношение больше 1, то для модели имеет место избыточная дисперсия. Чтобы получать эти значения для различных типов моделей:

- GLM: получить `summary(модель)` и проанализировать строку
  `Residual deviance: xx.xx, xx degrees of freedom`
- GLMM: использовать функцию `overdisp.glmer(модель)*`.

Если имеет место избыточная дисперсия, необходимо изменить закон распределения данных, на котором основана модель. Для GLM возможны два решения:

- заменить распределение Пуассона квази-пуассоновским законом,
  `модель <- glm(формула, family = "Quasipoisson")`.
- заменить распределение Пуассона отрицательным биномиальным распределением, используя функцию `glm.nb` из пакета `MASS`:
  `модель <- glm.nb(формула)`.

Использование закона отрицательного биномиального распределения часто дает очень хорошие результаты. Если этого не достаточно, может быть использовано квази-пуассоновское распределение, которое почти всегда решает проблему избыточной дисперсии остатков.

Для моделей GLMM доступно только отрицательное биномиальное распределение, для чего нужно построить модель с использованием функции из пакета `lme4`:

`модель <-glmer.nb (формула)`.

Адекватность по отношению к данным. Вторая необходимая проверка состоит в том, чтобы оценить, насколько хорошо модель описывает эмпирические данные. Этот этап фундаментален и выполняется для любой модели, так как выводы, основанные на плохо подогнанной модели, не имеют смысла и по-просту не надежны (или не действительны). Возвратитесь к п.**41** для подробного объяснения этой проверки.

Если подгонка модели выполнена не совсем хорошо, наиболее простой способ разрешения проблемы состоит в том, чтобы изменить шкалу представления объясняемой случайной величины. Для счетных данных классическими являются три опции преобразований: $\sqrt{x}$, $log(x)$ или $\sqrt[4]{x}$, если трансформации квадратного корня оказалось недостаточно, а нули в данных препятствуют логарифмированию. Однако трансформированный отклик больше уже не представляет собой численности объектов, а непрерывную переменную величину. Соответственно меняется тип модели и анализ становится тем, как это представлено в п.**76**.

Способность модели к объяснению. Можно оценить общую объяснясняющую ценность модели благодаря *коэффициенту детерминации* (R^2), который представляет собой долю от общей вариации отклика, которая объяснена независимыми переменными модели. Этот коэффициент может быть получен с использованием функции `r.squaredGLMM (модель)` из пакета `MuMIn`. Функция фактически возвращает два значения: *маргинальный* R^2 (`R2m`), который соответствует доле вариации, объясненной только ковариатами и фиксированными факторами, и *условный* R^2 (`R2c`), который соответствует доле вариации, объясненной всеми предикторами (т.е. как фиксированными, так и случайными факторами). В случае GLM оба значения идентичны, так случайные факторы отсутствуют.

Тестирование

Для всех упомянутых моделей, за исключением основанных на квази-пуассоновском распределении, значимость объясняющих переменных оценивается функцией:
`Anova (модель)` из пакета `car` (см. п. **42** для подробного объяснения процесса тестирования гипотез). Однако этот тест для разных моделей имеет характерные

особенности:

- для GLM функция выполняет анализ отношения максимального правдоподобия (*Likelihood Ratio Test* или LR Test) и тестирует отдельно каждый член модели (т.е. по строкам возвращаемый таблицы с оценками коэффициентов);
- для GLMM функция выполняет тест Вальда и анализирует отдельно каждый член модели (т. е. по строкам возвращаемый таблицы с оценками коэффициентов).

В случае GLM с использованием квази-пуассоновского распределения, необходимо использовать проверку по *F*-критерию, которая осуществляется командой `Anova (модель, test = "F")`.

Если один фактор (или взаимодействие с участием фактора) имеет статистически значимый эффект, необходимо осуществить множественное парное сравнение его уровней (или сочетаний этих уровней), чтобы выявить, в каких комбинациях имеются различия. Посмотрите п. **43** как выполнить эти сравнения.

Если ковариата имеет значимое влияние, то направление эффекта определяется знаком параметра. Полную информацию обо всех параметрах модели можно получить с помощью команды `summary(модель)`. Значения самих коэффициентов называются `Estimate` и находятся в таблице `Coefficients` для GLM и `Fixed effects` для GLMM. Если знак коэффициента для количественной переменной отрицателен, то вероятность отклика уменьшается, когда значение ковариаты увеличивается; а если он положительный, то вероятность увеличивается, когда значение ковариаты увеличивается.

Прогноз на основе модели

Смысл моделей – не только показать роль различных независимых переменных в изменчивости объясняемой переменной, но и предсказать значение, которое бы принял отклик при известных значений независимых переменных. Поэтому, чтобы оценить значение отклика, требуется зафиксировать значение всех переменных.

Для прогнозирования могут быть использованы два метода (для GLMM доступен только второй), и оба они основаны на функции `predict()`:

- значение каждой из независимых переменных дается непосредственно в теле функции, в виде списка (с одной или несколькими записями): `predict(модель,newdata=list(переменные), type="response")`, где `переменные` – это последовательность `var1= значение,var2=значение` и т.д.;
- создать таблицу, содержащую столбцы по каждой независимой переменной (имена столбцов должны строго совпадать с именами переменных модели), и заполнить каждую строку значениями, для которых должно быть сделано предсказание. Таким образом, делается прогноз по каждой строке таблицы:

`predict(модель, newdata = таблица, type = "response")`.

Примечание: в случае GLMM некоторые значения случайных факторов могут не иметь данных. Если это не произошло, то необходимо добавить в функцию `predict()` аргумент `re.form=NA`. Тогда при оценке прогноза будут приниматься во внимание все эффекты случайных факторов модели.

ПРИМЕРЫ

Имеем модель, содержащую `фактор` с двумя уровнями (`A` и `B`), ковариату `ковариата` в пределах от 0 до 30, и их взаимодействия :

```
> модель <- glm(отклик ~ фактор*ковариата,family="binomial")
```

Можно предсказать численность отклика таким образом :

```
> predict(модели,newdata= list (фактор="A", ковариата =10),
                                    type="response")
```

Или для нескольких прогнозов :

```
> predict(модель,newdata=list(фактор =c("A","B"),
                    ковариата =c(10,10), type="response")
```

Или предварительно создать таблицу следующего типа:

```
> таблица
  Фактор ковариата
1      A        10
2      B        10
> predict(модель, newdata= таблица, type="response")
```

Графики

Влияние фактора. Чтобы проиллюстрировать результат воздействия фактора, обычно рисуется диаграмма с "перекладинами", где представлена средняя численность для каждого уровня (условия). Могут быть представлены два типа средних:

- средние брутто (т.е. вычисляемые по исходным данным) и их стандартные ошибки (см. пп. **35** и **37**, как использовать функцию `tapply()`).

Внимание: в случае GLMM эти средние и стандартные ошибки не учитывают случайные факторы.

- средние значения, которые были скорректированы с учетом влияния других переменных модели, а также их стандартные ошибки : см. п. **43**.

После вычисления средних и стандартных ошибок может быть изображен и сам график (см. п. **34**).

Зависимость от ковариаты. Чтобы проиллюстрировать связь между объясняемой переменной и независимой ковариатой, требует три этапа:

1. Вывести на график точки наблюдения: `plot(отклик ~ ковариата)`.
2. Создать вектор, имеющий те же минимум и максимум, что `ковариата`, но с очень маленьким интервалом между его значениями: `x <- seq2(ковариата)*`.
3. Добавить кривую модельных значений на график.

Определение всех точек модельной кривой основано на прогнозе значений, которые принимает объясняемая переменная для каждого значения вектора `x`. Большое количество значений этого вектора и очень маленький интервал, который их отделяет, определяет гладкость кривой (которая на самом деле состоит из сотен точек, соединенных между собой). Если модель содержит другие независимые переменные, то к прогнозу необходимо добавить одинаковые значения, такие, чтобы величина отклика изменяла свою величину только при изменении анализируемой ковариаты. Кривая на график может быть добавлена функцией

```
lines(x, predict(модель, newdata= x, type="response")).
```

Примечание: функция `lines()` может быть использована несколько раз, чтобы начертить несколько кривых на одном графике, например, для нескольких уровней одного и того же фактора, включенного в модель. Чтобы изменить тип линий графика и цвета кривых, можно использовать аргументы `lty` и `col` соответственно. Наконец, чтобы добавить заголовки, используйте функцию `legend()`. Смотрите справку `?` для получения дополнительных сведений об используемых параметрах и знакомства с различными вариантами графики.

ПРИМЕРЫ

Рассмотрим следующую модель :

```
> модель <- glm(отклик ~ фактор*ковариата,family="poisson")
```

Шаг 1: нарисуем точки, соответствующие наблюдаемым данным:

```
> plot(отклик ~ ковариата)
```

Шаг 2: создадим вектор x :

```
> x <- seq2(ковариата)
```

Шаг 3: добавляем модельную кривую.

Выберем один из уровней фактора, включенного в модель и нанесем кривую на график:

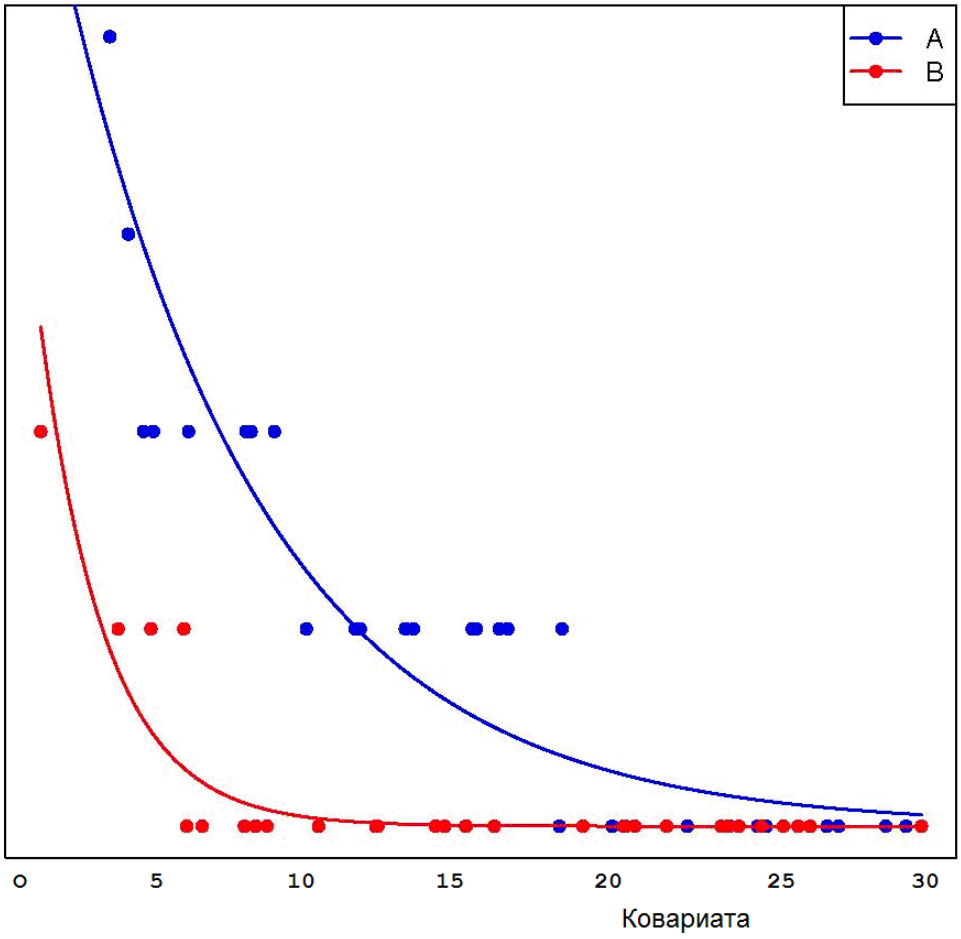
```
> lines(x,predict(модель, newdata=list(ковариата = x,
      фактор =rep (A,length(x))), type="response"), col = 2)
```

Функция rep() является очень полезной в этой ситуации, поскольку она позволяет очень просто создать вектор той же длины, что и x, содержащий одно и тоже повторяющееся значение A.

Чтобы добавить модельную кривую, соответствующую уровню B фактора :

```
> lines(x,predict(модель,newdata=list(ковариата = x,
          фактор =rep (B,length(x))), type="response"))
```

На полученном рисунке синим цветом представлена модельная кривая регрессии Пуассона для уровня A воздействующего фактора, а красным цветом – для уровня B.

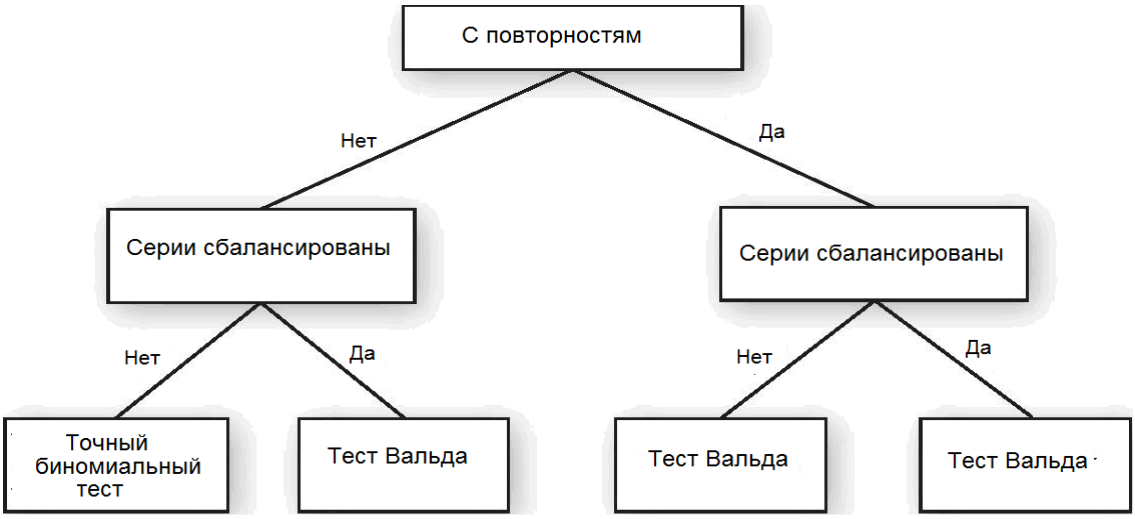


Анализ численности объектов, относящихся к 2 категориям

Этот тип отклика в простых тестах часто называют "пропорцией" или "долей". Но на самом деле "пропорция" не сводится только к подсчету числа индивидуумов.

56. Соответствие пропорции теоретическому значению

Выбор соответствующего теста



Нет повторностей

Точный биномиальный тест (непараметрический). Для выполнения анализа используется команда: `binom.test(n1, n, p=prop.theo)`, где `n1` – количество объектов в анализируемой категории, `n` – общая численность и `prop.theo` – теоретическая пропорция (0.5 по умолчанию).

ПРИМЕР

Мы хотим установить, является ли соотношение полов в выборке из 20 особей, содержащих 7 самок и 13 самцов сбалансированным (т. е. доля самок равна 0.5) :

```
> binom.test(7,20,p=0.5)
```

или проще, так как $p = 0.5$ по умолчанию :

```
> binom.test(7,20)
```

Тест Вальда (параметрический). Для выполнения анализа используется функция: `wald.ptheo.test(отклик, блок, p=prop.theo)*`, где `отклик` определяет, к какой категории принадлежит каждый объект (в числовой форме или фактор) и `блок` – случайный фактор, задающий группу каждого объекта (в том же порядке, что и `отклик`). Если `отклик` закодирован в форме 0/1, то проверяется доля группы 1; Если `отклик` является фактором, то проверяется доля 2-й модальности.

Есть повторности

Отклик должен быть матрицей из двух столбцов, где каждая строка представляет собой одну повторность. Для каждой из этих реплик указывается число отдельных объектов в каждой категории (по одному столбцу на категорию). Проверяемая пропорция определяется левым столбцом.

Тест Вальда со сбалансированными сериями (параметрический). Для выполнения анализа используется команда: `wald.ptheo.test(отклик,p=prop.theo)*`, где `отклик` является матрицей с двумя столбцами.

Тест Вальда с несбалансированными сериями (параметрический). Используется функция: `wald.ptheo.test(отклик, блок, p=prop.theo)*`, где `блок` – случайный фактор, задающий группу каждой повторности (т. е. имеется столько значений в векторе `блок`, сколько строк в матрице `отклик`).

57. Соответствие нескольких пропорций теоретическим значениям при 2-х категория

Данные должны быть представлены в виде матрицы из двух столбцов (называемой "таблицей сопряженности"), в которой строки соответствуют образцам/популяциям, а столбцы – двум анализируемым категориям:

	Категория 1	**Категория 2**
Образец/Популяция 1		
...		
Образец/Популяция *k*		

В каждой строке представлена численность объектов каждой категории. Тестируемые пропорции – в левом столбце (т.е. категория 1).

Критерий согласия χ^2 *на соответствие* (параметрический)

Условия: теоретические численности должны быть все ненулевыми и 80 % из них должны быть ≥ 5 ("правило Кохрана", см. ниже).

Эмпирические численности могут быть рассчитаны функцией: `chisq.exp (tab.cont, p=prop.theo)*`, где `tab.cont` – произвольная таблица сопряженности и `prop.theo` – вектор, задающий теоретическую пропорцию каждого образца (от 1 до *k*). Эмпирические численности позволяют проверить, соблюдается ли правило Кохрана.

Замечание: эмпирические пропорции *независимы* относительно образцов/популяций. Они не должны давать в сумме 1, так как образцы не сравниваются между собой, а каждый образец тестируется относительно его собственной теоретической пропорции.

Чтобы выполнить тест согласия χ^2, используется функция: `prop.test (tab.cont, p=prop.theo)*`.

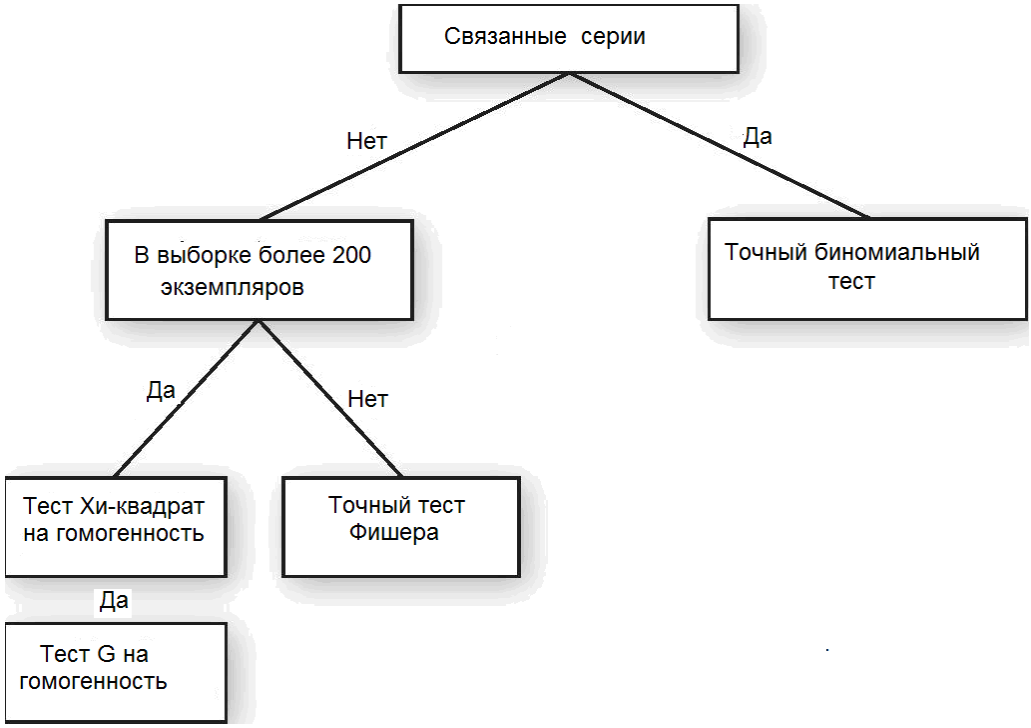
Замечание: функция `prop.test()` не использует правило Кохрана и выводит предупреждение, как только по крайней мере одна эмпирическая численность < 5. Если правило все же соблюдено, то не принимайте во внимание это предупреждение.

Статистически значимая величина *p* указывает на то, что, по крайней мере, одна пропорция будет отличаться от своего теоретического значения, не уточняя, какая именно. Чтобы определить, для каких именно образцов эмпирические доли отличаются от теоретических вероятностей, используют функцию: `prop.multcomp (tab.cont, p=prop.theo)*`.

Может случиться, что вопреки общему тесту множественные сравнения не показывают никаких статистически значимых различий. В этом случае, наиболее разумным решением будет считать, что мы не можем знать, какие именно пропорции привели к отказу от нулевой гипотезы в общем тесте.

58. Сравнение двух пропорций при 2-х категориях

Выбор соответствующего теста



Точный тест Фишера здесь всегда наиболее надежный, но время его расчета значительно увеличивается с ростом эмпирической численности. Если количество экземпляров достаточно велико, то приближение, сделанное тестами χ^2 и G, является достаточно удовлетворительным, чтобы их можно было использовать. Результаты этих двух тестов очень похожи; поэтому выбор между ними осуществляется скорее по привычке, чем по статистическим причинам.

Несвязанные серии

Данные должны быть представлены в виде матрицы (называемой "таблицей сопряженности") из двух столбцов, в которой строки соответствуют двум сравниваемым образцам, а столбцы – двум анализируемым категориям:

	Категория 1	**Категория 2**
Образец/Популяция 1		
Образец/Популяция 2		

В каждой строке представлена численность объектов каждой категории. Тестируемые пропорции – в левом столбце (т.е. категория 1).

Точный тест Фишера (непараметрический). Чтобы выполнить тест, можно использовать функцию `fisher.test(tab.cont)`, где `tab.cont` – таблица сопряженности.

Тест χ^2 на однородность (непараметрический), Чтобы провести тест, используют функцию: `prop.test(tab.cont)*`.

Тест G *на однородность* (непараметрический), Чтобы провести тест, используют функцию : `G.test(tab.cont)*`.

Связанные серии

В случае связанных серий, объекты (в статистическом смысле) соединены попарно. Это могут быть действительно различные объекты, либо один и тот же объект, наблюдаемый два раза (например, до и после воздействия).

Данные должны, как всегда, быть представлены таблицей сопряженности, но структурированной по-другому:

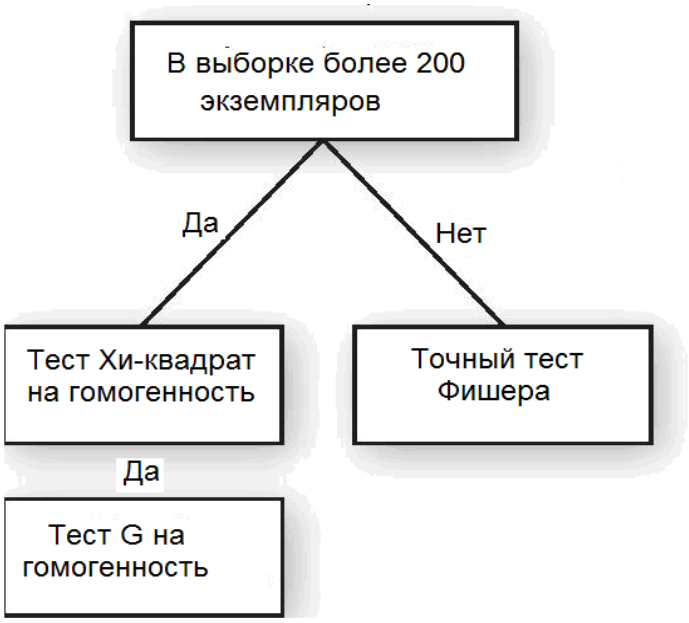
		2-й объект	
		Категория 1	**Категория** 2
1-й объект	**Категория** 1		
	Категория 2		

где строки соответствуют первым сравниваемым объектам каждой пары, а столбцы – вторым.

Точный биномиальный тест (непараметрический). Чтобы осуществить тест, выполняют команду: `binom.test(n1.2, nd)` где `n1.2` – количество индивидов в верхнем правом поле (т.е. число пар, первый индивид которых относится к категории 1, а второй – к категории 2), и `nd` – количество пар индивидов, у которых эти значения отличаются (т. е. поле в левом нижнем углу + поле в правом верхнем углу).

59. Сравнение более двух пропорций – 2 категории

Выбор соответствующего теста



Точный тест Фишера здесь всегда наиболее надежный, но время его расчета значительно увеличивается с ростом эмпирической численности. Если количество экземпляров достаточно велико, то аппроксимация, сделанная тестами χ^2 и G, является достаточно удовлетворительной, чтобы их можно было использовать. Результаты этих двух тестов очень похожи; поэтому выбор между ними осуществляется скорее по привычке, чем по статистическим причинам.

Данные должны быть представлены в виде матрицы из двух столбцов (называемой "таблицей сопряженности"), где строки соответствуют образцам/популяциям, а

столбцы – двум анализируемым категориям:

	Категория 1	**Категория 2**
Образец/Популяция 1		
...		
Образец/Популяция *к*		

В каждой строке представлена численность объектов в каждой категории. Тестируемые пропорции – в левом столбце (т.е. категория 1).

Точный тест Фишера (непараметрический). Чтобы выполнить тест, можно использовать функцию `fisher.test(tab.cont)`, где `tab.cont` – таблица сопряженности.

Статистически значимая величина *p* указывает, что, по крайней мере, две пропорции будут отличаться между собой, не уточняя, какие именно. Чтобы определить, для каких именно образцов доли отличаются между собой, используют функцию: `fisher.multcomp(tab.cont, p=prop.theo)*`.

Может случиться, что вопреки общему тесту множественные сравнения не показывают никаких статистически значимых различий. В этом случае, наиболее разумным решением будет считать, что мы не можем знать, какие именно пропорции привели к отказу от нулевой гипотезы в общем тесте.

Тест χ^2 *на однородность* (непараметрический). Чтобы провести тест, используют функцию: `prop.test (tab.cont)`. При необходимости проводятся множественные парные сравнения `pairwise.prop.test(tab.cont)*`.

Тест G *на однородность* (непараметрический). Чтобы провести тест, используют функцию : `G.test(tab.cont)`. При необходимости проводятся множественные парные сравнения `pairwise.G.test(tab.cont)*`.

60. Анализ численности объектов 2 категорий с использованием моделей

Обсуждаемые здесь процедуры анализа используют основные понятия построения моделей, изложенные в п. **39- 42**.

Отклик

Отклик представлен в виде численностей экземпляров, принадлежащих к двум категориям, одна из которых определяется как "Интересуемая категория". Предметом анализа является доля численности, представленная этой категорией и называемая в этом разделе "пропорцией".

Отклик должен быть матрицей из двух столбцов типа :

```
     Категория1 Категория2
[1,]         10         16
[2,]          8         24
[3,]         14         19
                ...
```

В этой матрице каждая строка соответствует одной серии наблюдений для каждой категории (столбцу). Общий объем серии является суммой численности экземпляров в обеих категориях. Общая численность может очень сильно варьироваться от одной строки к другой. Левый столбец (т.е. `Категория1`) определяет интересуемую

категорию.

Примечание: в статистическом смысле каждое наблюдение представлено строкой таблицы. Поскольку для каждого наблюдения указывается доля и общая численность, по которой рассчитывается эта доля, можно придать более или менее различный вес отдельным наблюдениям (поскольку доля, рассчитанная по более высокой численности, будет точнее).

Матрица отклика может быть получена командой `отклик <- cbind(Категория1, Категория2)`, где `Категория1` и `Категория2` являются векторами, соответствующими обоим столбцам, т.е. первые их значения соответствуют первой строке матрицы, а оба вектора находятся в одном и том же порядке. В формуле модели матрица `отклик` интерпретируется как объясняемая переменная.

Типы модели

Хотя основные принципы остаются одними и теми же, некоторые технические аспекты анализа отличаются в зависимости от того, связан ли отклик с другими факторами, определяющими некоторые важные для исследования условия эксперимента. Если связанный фактор отсутствует, то используется Обобщенная Линейная Модель (*Generalized Linear Model* или GLM), в то время, как при наличии связанных факторов – Смешанная Обобщенная Линейная Модель (*Generalized Linear Mixed Model* или GLMM). Термин "смешанный" подразумевает наличие, по крайней мере, одного случайного фактора, который используется, чтобы точно идентифицировать серии наблюдений (см. п. **11**).

В отличие от обыкновенной Линейной Модели (или ее смешанного варианта, см. далее п. **76**), GLM- или GLMM-модели не основаны на нормальном законе распределения. В случае пропорций обычно априори используют биномиальное распределение.

Примечание: на самом деле существуют несколько типов GLM, основанных на биномиальном распределении. Здесь описан наиболее часто встречающийся вариант – *логистическая модель*. Если все объясняющие переменные являются числовыми величинами (ковариатами), мы имеем частный случай *логистической регрессии*.

Построение модели

Для создания модели используют следующие функции:

- без связанных случайных факторов
  `модель<- glm(формула,family="binomial")`
- со связанными качественными случайными переменными:
  `модель<- glmer(формула,family="binomial")` из пакета `lme4`

См. п. **40** для подробного объяснения построения формулы. В целом можно напомнить, что:

- включение независимого фактора позволяет проверить, различаются ли пропорции между уровнями этого фактора;
- включение ковариаты позволяет проверить, существует связь между этой ковариатой и объясняемой переменной;
- включение взаимодействия двух независимых переменных позволяет проверить, зависит ли один эффект от другого (это же рассуждение – для наблюдений, приуроченных к двум или более отрезкам времени). Особый случай – анализ взаимодействия между ковариатой и фактором, который позволяет проверить, зависит ли отношение между независимой переменной и откликом от уровней фактора.

Примечание: поскольку статистические единицы являются строками матрицы отклика, в этой матрице столько значений объясняемой переменной, сколько строк.

Проверка адекватности модели

Избыточная дисперсия остатков. Первый этап валидации модели состоит в том, чтобы проверить, не содержат ли остатки избыточной дисперсии (*overdispersion*), т.е. их вариация не больше ожидаемой, исходя из свойств биномиального распределения. Для этого достаточно сравнить отклонение остатков (*residual deviance*) модели со степенями свободы для остатков (*residual degrees of freedom*). Если это отношение больше 1, то для модели имеет место избыточная дисперсия. Чтобы получать эти значения для различных типов моделей:

- GLM: получить `summary (модель)` и проанализировать строку
  `Residual deviance: xx.xx, xx degrees of freedom`
- GLMM: использовать функцию `overdisp.glmer (модель)*`.

Если имеет место избыточная дисперсия, необходимо изменить закон распределения данных, на котором основана модель. Для GLM можно заменить биномиальное распределение квази-биномиальным законом:
`модель <- glm(формула, family = "quasibinomial")`.

Для GLMM метод заключается в добавлении случайного фактора к модели, каждый уровень которого соответствует отдельному (статистическому) наблюдению. Это делается в два этапа:

1. Создется случайный фактор `obs <- factor(1:nrow(отклик))`.
2. Переформировывается модель с добавлением в конец формулы `+(1|obs)`.

Адекватность по отношению к данным. Вторая необходимая проверка состоит в том, чтобы оценить, насколько хорошо модель описывает эмпирические данные. Возвратитесь к п. **41** для подробного объяснения этой проверки.

Способность модели к объяснению. Можно оценить общую объяснясняющую ценность модели благодаря *коэффициенту детерминации* (R^2), который представляет собой долю от общей вариации отклика, которая объяснена независимыми переменными модели. Этот коэффициент может быть получен с использованием функции `r.squaredGLMM(модель)` из пакета `MuMIn`. Функция фактически возвращает два значения: *маргинальный* R^2 (`R2m`), который соответствует доле вариации, объясненной только ковариатами и фиксированными факторами, и *условный* R^2 (`R2c`), который соответствует доле вариации, объясненной всеми предикторами (т.е. как фиксированными, так и случайными факторами). В случае GLM оба значения идентичны, так случайные факторы отсутствуют.

Тестирование

Для всех упомянутых моделей, за исключением основанных на квази-биномиальном распределении, значимость объясняющих переменных оценивается функцией:
`Anova(модель)` из пакета `car` (см. п. **42** для подробного объяснения процесса тестирования гипотез). Однако этот тест для различных моделей имеет характерные особенности:

- для GLM функция выполняет анализ отношения максимального правдоподобия (*Likelihood Ratio Test* или LR Test) и тестирует отдельно каждый член модели (т.е. по строкам возвращаемый таблицы с оценками коэффициентов);
- для GLMM функция выполняет тест Вальда и анализирует отдельно каждый член модели (т.е. по строкам возвращаемый таблицы с оценками коэффициентов).

В случае GLM с использованием квази-биномиального распределения, необходимо использовать проверку по *F*-критерию, которая осуществляется командой
`Anova(модель, test = "F")`.

Если один фактор (или взаимодействие с участием фактора) имеет статистически значимый эффект, необходимо осуществить множественное парное сравнение его уровней (или сочетаний этих уровней), чтобы выявить, в каких комбинациях имеются различия. Посмотрите п. **43** как выполнить эти сравнения.

Если ковариата имеет значимое влияние, направление эффекта определяется знаком параметра. Полную информацию обо всех параметрах модели можно получить с помощью команды `summary(модель)`. Значения самих коэффициентов называются `Estimate` и находятся в таблице `Coefficients` для GLM и `Fixed effects` для GLMM. Если знак коэффициента для количественной переменной отрицательный, то пропорция интересуемой категории уменьшается, когда значение ковариаты увеличивается; а если он положительный, то пропорция увеличивается, когда значение ковариаты увеличивается.

Прогноз на основе модели

Смысл моделей – не только показать роль различных независимых переменных в изменчивости объясняемой переменной, но и предсказать значение, которое принял бы отклик при известных значений независимых переменных. Поэтому, чтобы оценить значение отклика, требуется зафиксировать значение всех переменных.

Для прогнозирования могут быть использованы два метода (для GLMM доступен только второй) и оба они основаны на функции `predict()` :

- значение каждой из независимых переменных дается непосредственно в теле функции, в виде списка (с одной или несколькими записями): `predict(модель,newdata=list(переменные),type="response")`, где `переменные` – последовательность `var1= значение,var2=значение` и т.д.;
- создать таблицу, содержащую столбцы по каждой независимой переменной (имена столбцов должны строго совпадать с именами переменных модели), и заполнить каждую строку значениями, для которых должно быть сделано предсказание. Таким образом, делается прогноз по каждой строке таблицы:

`predict(модель, newdata = таблица, type = "response")`.

Примечание: в случае GLMM некоторые значения случайных факторов могут не иметь данных. Если это не произошло, то необходимо добавить в функцию `predict()` аргумент `re.form=NA`. Тогда при оценке прогноза будут приниматься, во внимание все эффекты случайных факторов модели.

ПРИМЕРЫ

Имеем модель, содержащую `фактор` с двумя уровнями (A и B), ковариату `ковариата` в пределах от 0 до 30, и их взаимодействия:

```
> модель <- glm(отклик ~ фактор*ковариата,family="binomial")
```

Можно предсказать пропорцию для отклика таким образом :

```
> predict(модели,newdata= list (фактор="A", ковариата =10),
                                  type="response"),
```

Или для нескольких прогнозов :

```
> predict(модель,newdata=list(фактор =c("A","B"),
                    ковариата =c(10,10), type="response")
```

Или предварительно создать таблицу следующего типа:

```
> таблица
  Фактор ковариата
1      A        10
2      B        10
> predict(модель, newdata= таблица, type="response")
```

Графики

Влияние фактора. Чтобы проиллюстрировать результат воздействия фактора, обычно рисуется гистограмма, где представлена средняя пропорция для каждого уровня (условия). Могут быть представлены два типа средних:

- средние брутто (т. е. вычисляемые по исходным данным) и их стандартные ошибки (см. пп. **35** и **37**, как использовать функцию `tapply()`). *Внимание*: в случае GLMM эти средние и стандартные ошибки не учитывают случайные факторы.

Примечание: функция `tapply()` работает для вектора числовых значений. Здесь переменная, которую нужно объяснить, представляет собой матрицу с двумя столбцами, которая не подходит. Поэтому сначала необходимо вручную рассчитать пропорции: `пропорция <- Категория1/(Категория1 + Категория2)`. Именно на этом векторе пропорций должна использоваться функция `tapply()`.

- средние значения, которые были скорректированы с учетом других переменных модели, а также их стандартные ошибки : см. п. **43**.

После вычисления средних и стандартных ошибок может быть изображен и сам график (см. п. **34**).

Зависимость от ковариаты. Чтобы проиллюстрировать связь между объясняемой переменной и независимой ковариатой, требуется три этапа:

1. Вывести на график точки наблюдения: `plot(пропорция ~ ковариата)`.
2. Создать вектор, имеющий те же минимум и максимум, что `ковариата`, но с очень маленьким интервалом между его значениями : `x <- seq2(ковариата)*`
3. Добавить кривую модельных значений на график.

Определение всех точек модельной кривой основано на прогнозе значений, которые принимает объясняемая переменная для каждого значения вектора `x`. Большое количество значений этого вектора и очень маленький интервал, который их отделяет, определяет гладкость кривой (которая на самом деле состоит из сотен точек, соединенных между собой). Если модель содержит другие независимые переменные, то к прогнозу необходимо добавить одинаковые значения, такие, чтобы величина отклика изменяла свою величину только при изменении анализируемой ковариаты. Кривая на график может быть добавлена функцией
`lines(x,predict(модель,newdata= x, type="response")).`

Примечание: функция `lines()` может быть использована несколько раз, чтобы начертить несколько кривых на одном графике, например, для нескольких уровней одного и того же фактора, включенного в модель. Чтобы изменить тип линий графика и цвета кривых, можно использовать аргументы `lty` и `col` соответственно. Наконец, чтобы добавить заголовки, используйте функцию `legend()`. Смотрите справку `?` для получения дополнительных сведений об используемых параметрах и знакомства с различными вариантами графики.

ПРИМЕРЫ

Рассмотрим следующую модель :

```
> модель <- glm(отклик ~ фактор*ковариата,family="binomial")
```

Шаг 1: нарисуем точки, соответствующие наблюдаемым данным :

```
> plot(пропорция ~ ковариата)
```

Шаг 2: создадим вектор `x` :

```
> x <- seq2(ковариата)
```

Шаг 3: добавляем модельную кривую. Выбираем один из уровней фактора, включенного в модель:

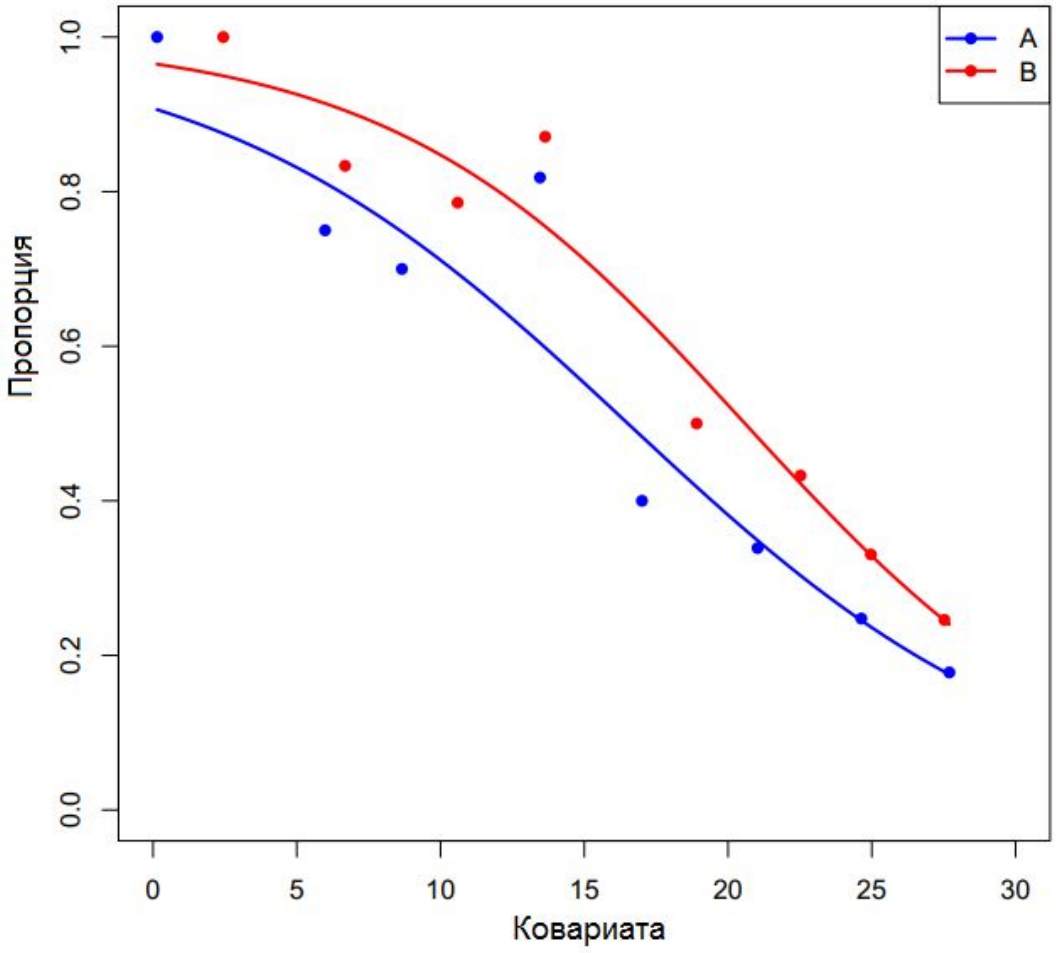
```
> lines(x,predict(модель,newdata=list(ковариата = x,
      фактор =rep (A,length(x))), type="response"), col = 2)
```

Функция `rep()` является очень полезной в этой ситуации, поскольку она позволяет очень просто создать вектор той же длины, что и `x`, содержащий одно и то же повторяющееся значение `A`.

Чтобы добавить модельную кривую, соответствующую уровню `B` фактора :

```
> lines(x,predict(модель,newdata=list(ковариата = x,
          фактор =rep (B,length(x))), type="response"))
```

На полученном рисунке синим цветом представлена модельная кривая логистической регрессии для уровня A воздействующего фактора, а красным цветом – уровня B.

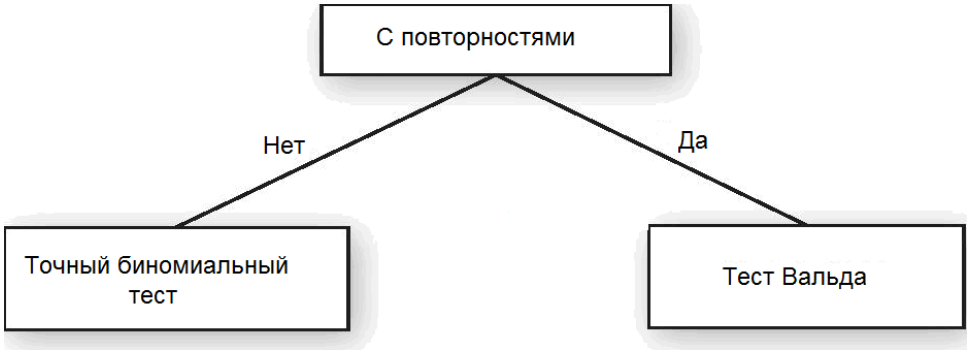


Отклик – счетные данные более чем 2-х категорий

61. Соответствие нескольких пропорций теоретическим значениям - более 2 категорий

Отклик должен быть матрицей, где каждая строка соответствует повторностям (которая может быть только одна и в этом случае отклик – вектор), а каждый столбец – категории. Таким образом, для каждой повторности есть подсчет отдельных экземпляров в каждой из категорий. Тестирование пропорций осуществляется отдельно для каждой категории.

Чтобы выбрать подходящий тест:



Точный мультиномиальный тест (непараметрический).
Чтобы выполнить тест, можно использовать функцию: `multinomial.theo.multcomp(отклик, p = prop.theo, prop=TRUE)*`, где `отклик` – вектор из *K* значений (для каждой из категорий), `prop.theo` –. вектор, определяющий теоретическую долю каждой категории (от 1 до *K*), сумма которых должна составлять 1. Точный биномиальный тест выполняется отдельно для каждой анализируемой категории.

Тест Вальда (параметрический). Для выполнения анализа используется функция: `wald.ptheo.multinom.test(отклик, p=prop.theo)*`, где `отклик` – матрица с *K* столбцами (для *K* категорий), содержащая столько строк, сколько имеется повторностей. Тест Вальда выполняется отдельно для каждой анализируемой категории.

Примечание: анализ учитывает тот факт, что общая численность каждой повторности может различаться.

ПРИМЕР

Имеем следующие данные:

```
> отклик
     Категория1    Категория2    Категория3
[1,]         10            16            16
[2,]          8            24            17
[3,]         14            19            20
```

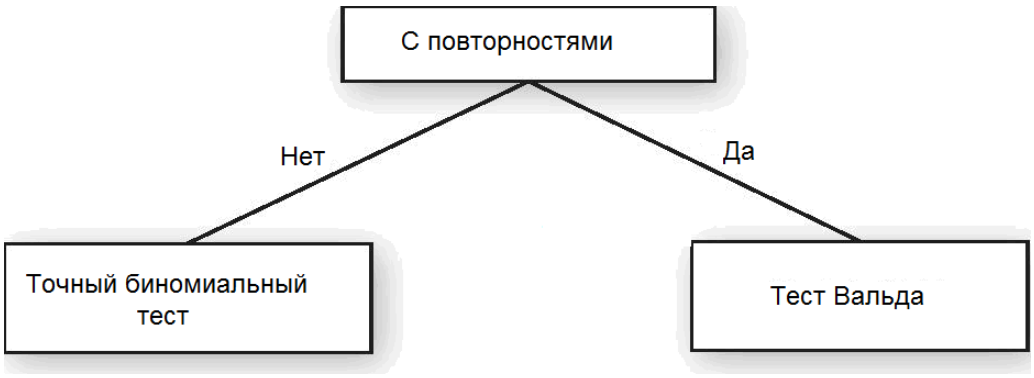
Пропорции сравниваются с теоретическими пропорциями 0.25 / 0.5 / 0.25 ::

```
> wald.ptheo.multinom.test(отклик, p=c(0.25,0.5,0.25) )*
```

62. Сравнение нескольких пропорций – более 2 категорий

Отклик должен быть матрицей, где каждая строка соответствует повторностям (которая может быть только одна и в этом случае отклик – вектор), а каждый столбец – категории. Таким образом, для каждой повторности есть подсчет отдельных экземпляров в каждой из категорий. Тестирование пропорций осуществляется отдельно для каждой категории.

Чтобы выбрать подходящий тест :



Точный мультиномиальный тест (непараметрический).
Чтобы выполнить тест, можно использовать функцию:
`multinomial.multcomp(отклик)*`, где `отклик` – вектор из *K* значений (для каждой из категорий). Функция фактически выполняет все возможные парные множественные сравнения между всеми анализируемыми категориями.

Тест Вальда (параметрический). Для выполнения анализа используется функция: `prop.multinom.test(отклик)*`, где `отклик` – матрица с *K* столбцами (для *K* категорий), содержащая столько строк, сколько имеется повторностей. Функция фактически выполняет все множественные сравнения между всеми анализируемыми категориями.

ПРИМЕР

Имеем следующие данные:

```
> отклик
     Категория1   Категория2   Категория3
[1,]         10           16           16
[2,]          8           24           17
[3,]         14           19           13
```

Рассчитывается доля каждой категории (и ее стандартная ошибка):

```
>     prop.multinom.test(отклик)*
$probs
Категория1   Категория2   Категория3
    0.2336       0.4307       0.3358
$se
Категория1   Категория2   Категория3
    0.0416       0.0325       0.0285
```

И эти доли можно сравнить между собой:

```
>     prop.multinom.test(отклик)*
```

63. Анализ счетных данных более чем 2 категорий с использованием моделей

Обсуждаемые здесь процедуры анализа используют основные понятия построения моделей, изложенные в п. **39- 42**.

Отклик

Задается отклик в виде подсчета численностей объектов, принадлежащих к более чем двум категориям. Необходимо проанализировать пропорции, представленные каждой категорией.

Таким образом, мы имеем для каждого наблюдения не одно уникальное значение отклика, как и в большинстве тривиальных ситуаций (например, анализ количества экземпляров в двух категориях, см. п. **60**), а одновременно несколько значений. Это серьезно усложняет интерпретацию результатов.

Для анализа такого отклика необходимо определить "*ссылочную категорию*", поскольку модель на самом деле будет оценивать соотношение между долей каждой анализируемой категории в сравнении с долей эталонной категории. Таким образом, для K категорий необходимо интерпретировать K - 1 соотношений по объясняемой переменной, которые расширяются до $K(K\text{-}1)/2$ соотношений для всех парных комбинаций K категорий. Очевидна сложность работы с такими откликами. Поэтому понятно стремление ограничиться очень простым экспериментальным планом (т.е. небольшим количеством объясняющих переменных и еще меньшим количеством взаимодействий) и ограниченным числом категорий (т.е. K). По аналогичным причинам лучше избегать качественных объясняющих переменных, варьирующих на более, чем двух уровнях (или нужно быть готовым к интерпретации, которая может быть и сложной, и утомительной).

Отклик должен быть задан матрицей из K столбцов типа :

```
     Категория1   Категория2   Категория3
[1,]         10           16           16
[2,]          8           24           17
[3,]         14           19           20
```

В этой матрице каждая строка соответствует одной повторности, а каждый столбец – одной категории. Для каждого повторности, таким образом, можно вычислить количество экземпляров во всех K категориях. Левый столбец (т. е. `Категория1`) определяется как ссылочная категория. Выбор этой категории не влияет на анализ.

Примечание: в статистическом смысле одно наблюдение представлено строкой таблицы. Таким образом, для каждой статистической единицы существует ряд пропорций, а также общая численность, на основе которой рассчитываются эти пропорции, что дает возможность как-то оценить вес отдельных наблюдений (поскольку доля, рассчитанная по высокой численности, более точна). Общая численность также может очень сильно варьироваться от одной строки к другой.

Матрица отклика может быть получена командой `отклик <- cbind(Категория1, Категория2,…, КатегорияК)`, где векторы `Категорияi` соответствуют столбцам, т.е. первые их значения соответствуют первой строке матрицы отклика, а все векторы сформированы в одном и том же порядке. В формуле модели матрица `отклик` интерпретируется как объясняемая переменная.

Типы модели

Используемая модель представляет собой мультиномиальную (или "неупорядоченную" политомическую) модель, которая является расширением обобщенной линейной модели GLM (*Generalized Linear Model*), анализирующей

соотношение численности отдельных экземпляров в двух категориях (см. п. **60**).

Примечание: существует несколько типов мультиномиальных моделей. Мы рассматриваем здесь наиболее частый случай – логистическую модель. Если все независимые переменные являются количественными, то мы имеем мультиномиальную логистическую регрессию.

Построение модели

Для создания модели используют функцию из пакета `multinom`:
`модель<-multinom(формула,abstol=1e-15,reltol=1e-15,maxit=1000)`.
См. п. **40** для подробного объяснения построения формул. Аргументы `abstol`, `reltol` и `maxit` – параметры, обеспечивающие более высокую точность результатов.

Примечание: построение мультиномиальной модели функцией `multinom()` – итерационный процесс, ход которого может отображаться на экране. Чтобы удалить все сообщения при построении модели, необходимо добавить аргумент `trace=FALSE`.

Проверка адекватности модели

Поскольку объясняемая переменная не является количественной, проверка модели обычным образом здесь не выполняется. Однако мы обычно хотим реально знать, насколько оцененные параметры модели зависят от состава обрабатываемых данных. Если параметры очень чувствительны, то малейшее изменение в данных приводит к тому, что модель становится вырожденной. Это – признак того (среди прочего), что некоторые параметры не предсказуемы и модель может быть упрощена.

Значение, которое может оценить эту тенденцию – обусловленность матрицы Гессе. Ее расчет осуществляется с помощью функции `cond.multinom(модель)*`. Здесь нет, в действительности, абсолютного критического порога, но обычно считается, что, если обусловленность превышает 10^5 , то это является признаком того, что модель плохо определена.

Тестирование

Влияние независимых переменных проверяется с помощью дисперсионного анализа `Anova(модель)` из пакета `car` (см. п. **42** с подробным описанием проверки гипотез). Функция выполняет анализ отношения максимального правдоподобия (*Likelihood Ratio Test* или LR Test) и тестирует отдельно каждый член модели (т.е. по строкам возвращаемый таблицы с оценками коэффициентов).

Если независимая переменная оказывается статистически значимой, это означает, что она оказывает влияние на соотношение, по крайней мере, между двумя категориями переменной отклика. Чтобы определить, на какие именно уровни отклика оказывается значительное влияние, используют функцию: `test.multinom(модель, переменная)*`, где `переменная` является независимой переменной, для которой требуется изучить детальное влияние.

Возвращаемый результат функции `test.multinom()` зависит от природы независимой переменной.

1. Для количественной независимой переменной функция возвращает массив, где каждая строка соответствует отношению между двумя классами переменной отклика (синтаксис A|B означает «*доля* A *в соотношении с долей* B»). Знак коэффициента показывает направление связи между категорией отклика и независимой переменной. Отношение шансов (*Odds ratio*) – простой способ интерпретации результата, который показывает, насколько меняется соотношение вероятностей при увеличении на единицу независимой переменной.

Примечание: не удивителен тот факт, что частные *p*-значения этого теста будут несколько отличаться от общих *p*-значений, полученных с помощью `Anova(модель)`. Во-первых, общий тест не сводится к сумме отдельных тестов, а, во-вторых, это разные тесты: на глобальном уровне оценивается максимальное правдоподобие, а на частном уровне выполняется тест Вальда. Второй тест менее мощный, чем первый (см. п. **17**).

ПРИМЕР. Для модели, где отклик имеет три категории (A/B/C), ковариата имеет значимое влияние. Подробно ее влияние состоит в следующем :

```
      Coeff      SE Odds.ratio       z Pr(>|z|)
A|C -0.93509 0.57383     0.3926 -1.6296  0.10319
B|C  1.56595 1.05528     4.7872  1.4839  0.13783
B|A  2.50104 1.20247    12.1952  2.0799  0.03753 *
--
Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Вывод: ковариата влияет только на соотношение между категориями A и B (*p*-значение меньше 0.05). Отношение шансов (*Odds ratio*) показывает, что вероятность того, что объект имеет категорию B, а не A, увеличивается в 12.2 раза, если значение ковариаты увеличивается на единицу.

Примечание: значимость отличия коэффициента от 0 равносильна значимости отличий отношения шансов от 1.

2. Если независимая переменная – фактор с двумя уровнями, то функция возвращает таблицу, структурированную как для ковариаты. Отношения шансов и значения оценок коэффициентов при соотношении вероятностей указываются как для обоих сравниваемых уровней независимого фактора, которые указываются в заголовке таблицы.

ПРИМЕР. Для модели, где отклик имеет три класса (A/B/C), фактор при двух уровнях (**`Самцы|Самки`**) имеет значимое влияние. Подробности этого влияния представлены следующим фрагментом протокола:

```
      Coeff     SE Odds.ratio       z Pr(>|z|)
A|C  0.1702 1.8533    1.18550  0.09182  0.92684
B|C -1.9660 0.8645    0.14002 -2.27415  0.02296 *
B|A -5.1361 4.0226    0.00588 -1.27682  0.20166
--
Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Вывод: фактор влияет на соотношение между категориями B и C. `Odds ratio` показывает, что соотношение вероятностей того, что самцы определяют категорию B, а не C, оказывается 0.14 по сравнению с самками (т. е. самки имеют в 1/0.14 = 7.14 раз больше шансов быть B, а не C, по сравнению с самцами).

3. Если независимая переменная – фактор более чем с двумя уровнями, то функция возвращает таблицу соотношений между категориями отклика для каждого уровня этой переменной. Поэтому этот вариант – просто расширение предыдущего случая, который труднее интерпретировать.

Прогноз на основе модели

Смысл моделей – не только показать роль различных независимых переменных в изменчивости объясняемой переменной, но и предсказать значение, которое бы принял отклик при известных значений независимых переменных. Поэтому, чтобы оценить набор соотношений между категориями отклика, требуется зафиксировать значение

всех независимых переменных.

Для прогнозирования могут быть использованы два метода, и оба они основаны на функции `predict()`:

- значение каждой из независимых переменных записывается непосредственно в теле функции в виде списка (с одной или несколькими записями): `predict(модель,newdata=list(переменные),type="probs")`, где `переменные` – последовательность `var1= значение, var2=значение` и т.д.;
- создать таблицу, содержащую столбцы по каждой независимой переменной (имена столбцов должны строго совпадать с именами переменных модели) и заполнить каждую строку значениями, для которых должно быть сделано предсказание. Таким образом, делается прогноз по каждой строке таблицы:

`predict(модель, newdata = таблица, type = "probs")`.

Функция `predict()` возвращает для каждого прогноза долю каждой категории переменной отклика. Результат – обычно матрица, в которой каждая строка составляет прогноз, а каждый столбец соответствует доле категории отклика.

ПРИМЕРЫ

Имеем модель, содержащую `фактор` с двумя уровнями (`A` и `B`), ковариату `ковариата` в пределах от 0 до 30, и их взаимодействия :

```
> модель <- multinom(отклик ~ фактор + ковариата,
                abstol=1e-15,reltol=1e-15,maxit=1000)
```

Можно предсказать соотношение категорий отклика таким образом :

```
> predict(модели,newdata= list (фактор=" A", ковариата =10),
                                    type = "probs"),
```

Или для нескольких прогнозов :

```
> predict(модель,newdata=list(фактор =c("A","B"),
                      ковариата =c(10,10), type = "probs")
```

Или предварительно создать таблицу следующего типа:

```
> таблица
  Фактор ковариата
1      A        10
2      B        10
> predict(модель, newdata= таблица, type = "probs")
```

ОТКЛИК – НЕОГРАНИЧЕННАЯ НЕПРЕРЫВНАЯ ВЕЛИЧИНА

64. Стратегия анализа непрерывного отклика с ограниченным интервалом

Если на заданном интервале имеется относительно небольшой объем данных, всегда лучше попробовать использовать модель для неограниченного непрерывного отклика (см. п. **76**). Анализ остатков этой модели (см. п. **41**) подскажет, приемлем ли такой подход или нет. Если в заданных пределах достаточно много данных или модель неограниченного отклика недостаточно хорошо подгоняется, желательно использовать какую-то иную стратегию анализа. Этот раздел представляет один из возможных вариантов преобразования данных ограниченного диапазона на всю непрерывную шкалу, но мы не можем гарантировать, что он будет функционировать всегда.

Принцип состоит в том, чтобы трансформировать значения объясняемой переменной таким образом, чтобы можно затем было проанализировать их как неограниченный непрерывный отклик. Преобразование делается в два этапа:

- значения отклика делятся на максимально возможную величину (даже если она не представлена в наборе данных), в результате чего получаем преобразованную переменную на интервале [0; 1],
- выполняется логит-преобразование, применяемое к долям (т.е. в отношении непрерывных откликов, ограниченных 0 и 1), используя функцию из пакета `car`: `отклик2 <-logit (отклик)`.

Затем новый вектор `отклик2` используется в качестве объясняемой переменной величины для модели неограниченного непрерывного отклика (так, как это описано в п. **76**).

Простые тесты, касающиеся распределений

65. Соответствие распределения непрерывной переменной теоретическому закону

Поскольку редко бывает, что распределение реальных данных вполне соответствуют теоретическому закону, необходимо оценить степень отклонения от сделанных предположений. Уровень согласия теоретического и эмпирического распределений проверяется, прежде всего, *графически*, что весьма информативно, но всегда субъективно, либо с использованием статистических тестов.

Графический тест

Визуальный тест может быть осуществлен с использованием квантиль-квантильного графика, формируемого следующим образом:
`qqPlot (выборка, dist = "loi",par)` , где `выборка` – вектор, содержащий серию данных, `loi` - выбранное теоретическое распределение (в кавычках) со своими параметрами, отделяемыми запятой (см. п. **23 - 28**).

ПРИМЕРЫ (с использованием функции из пакета `car`)
Приближение экспоненциальным законом (см. п. **25**):

- `qqPlot (выборка, dist = "exp",lambda)`

Приближение распределением χ^2 (см. п. **26**):

- `qqPlot (выборка, dist = "chisq", ddl)`

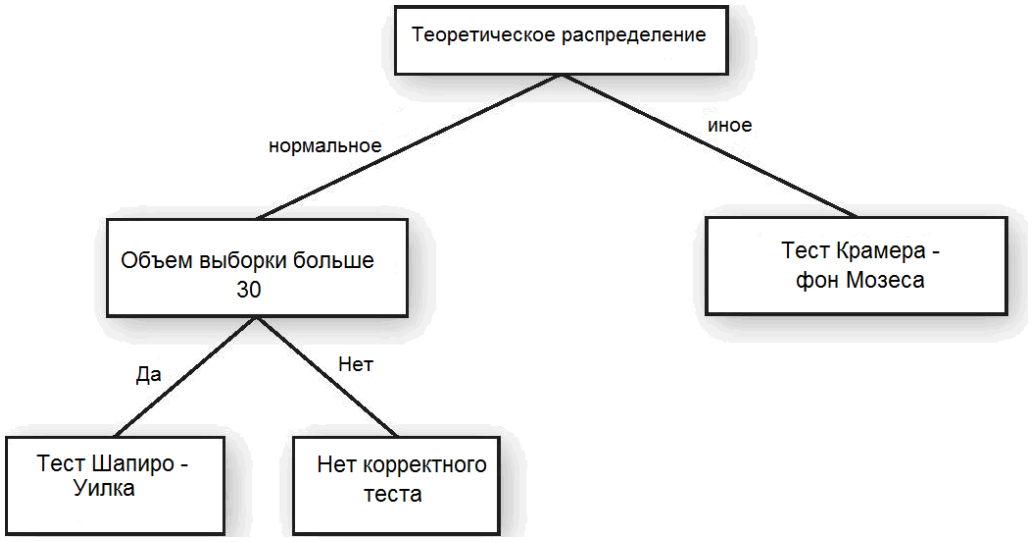
В случае нормального распределения, тот же результат можно получить непосредственно `qqPlot (выборка)`, поскольку нормальный закон используется по умолчанию. Чтобы сформировать график для каждого уровня фактора используют функцию: `byf.qqnorm(отклик~фактор)*`, где `отклик` и `фактор` – векторы, содержащие значение каждого элемента выборки для обеих переменных величин (в одном и том же порядке). Символ ~ означает "*в зависимости от*".

В случае приближения многомерным нормальным распределением можно использовать функции `mqqnorm()*` и `byf.mqqnorm()*` (см. `? mqqnorm` и `? byf.mqqnorm` для детальной информации).

Выборочное распределение согласуется с выбранным теоретическим законом, если точки на графике выровнены по главной диагонали. Любое другое структурирование точек (кривизна, многочисленные выбросы) указывает на противоположное. Точки могут быть не полностью выровнены относительно прямой, но, если они остаются в доверительном интервале, то приближение считается корректным.

Статистический тест

Чтобы выбрать подходящий тест:



Тест Шапиро-Уилка (непараметрический) используется для оценки согласия с нормальным законом распределения и реализуется следующим образом: `shapiro.test(выборка)`. Чтобы проверить нормальность распределения переменной для каждого уровня сопутствующего фактора, используют функцию: `byf.shapiro(отклик~фактор)*`.

В случае оценки приближения к многомерному нормальному распределению используйте функции `mshapiro.test()*` и `byf.mshapiro()*` (см. `? mshapiro.test` и `? byf.mshapiro` для детальной информации).

Тест Крамера – фон Мизеса (непараметрический) используется для оценки согласия с законом, отличным от нормального, и реализуется следующим образом:
`cvm.test(выборка, "loi",par)*`,
где `loi` - выбранное теоретическое распределение и его параметры.

ПРИМЕРЫ

Приближение экспоенциальным законом (см. п. **25**):

```
>   cvm.test(выборка, pexp,lambda)
```

Приближение распределением χ^2 (см. п. **26**):

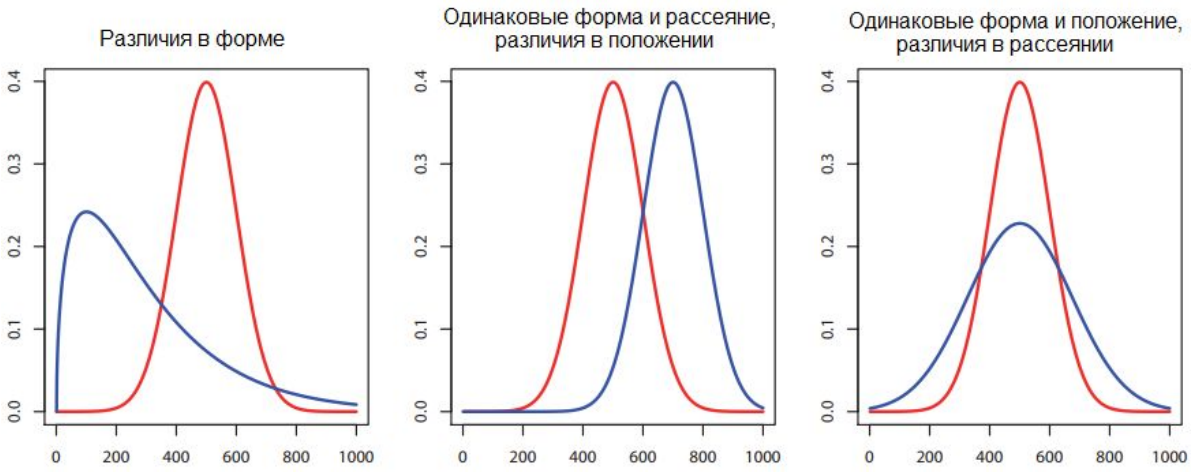
```
>   cvm.test(выборка, pchisq,ddl)
```

66. Сравнение двух выборочных распределений

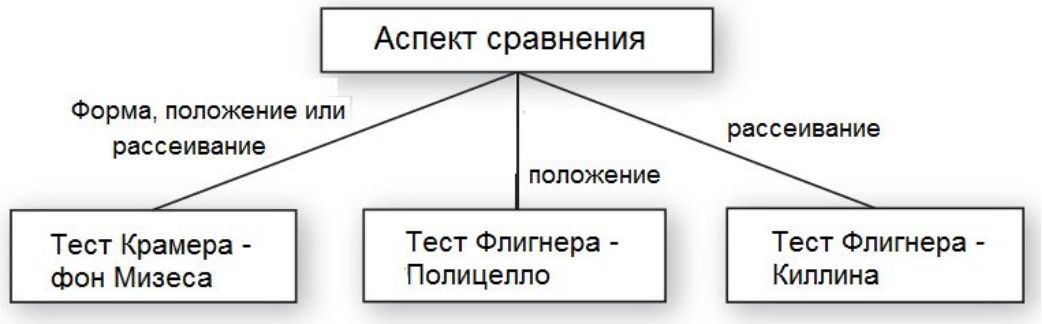
Когда сравниваются два выборочных распределения, мы можем интересоваться тремя аспектами их различий:

- различия в *форме.*
- различия в *положении.* Центр положения может иногда резюмироваться в таком параметре, как медиана (см. п. **35**).
- Различия в степени *рассеивания.* Степень разброса обычно резюмируется в таком параметре, как дисперсия (см. п. **36**).

Замечание: тесты, чувствительные к различиям в форме, также чувствительны к различиям в положении и/или рассеивании. Они названы *омнибусами,* так как они тестируют три аспекта одновременно.



Чтобы выбрать подходящий тест:



Тест Крамера - фон Мизеса (непараметрический). Чтобы осуществить тест, можно использовать функцию: `CvM.test (выборка1, выборка2)*`, где `выборка1` и `выборка2` – векторы, содержащие обе серии данных (которые не обязаны быть одинаковой длины).

Тест Флигнера - Полицелло (непараметрический).
Дополнительное условие: распределение данных должно быть симметричным в обеих группах (но неважно, если их форма отличается). Чтобы осуществить тест, можно использовать функцию: `fp.test(выборка1, выборка2)*`. Обе серии данных не обязаны быть одинаковой длины.

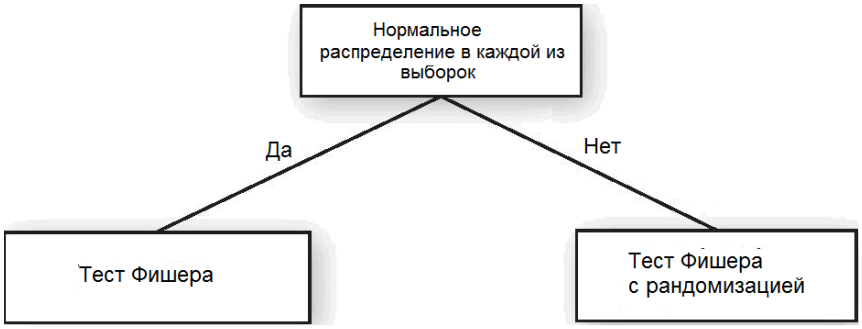
Этот тест не чувствителен ни к форме, ни к рассеиванию.

Тест Флигнера - Киллина (непараметрический). Чтобы осуществлять тест, используют функцию: `fligner.test(выборка1, выборка2)`. Обе серии данных не обязаны быть одинаковой длины. Этот тест не чувствителен к положению. Он хорошо функционирует, если данные приблизительно нормально распределены, но очень консервативен, если это не так.

Простые тесты, касающиеся дисперсий

67. Сравнение двух дисперсий

Чтобы выбрать подходящий тест:



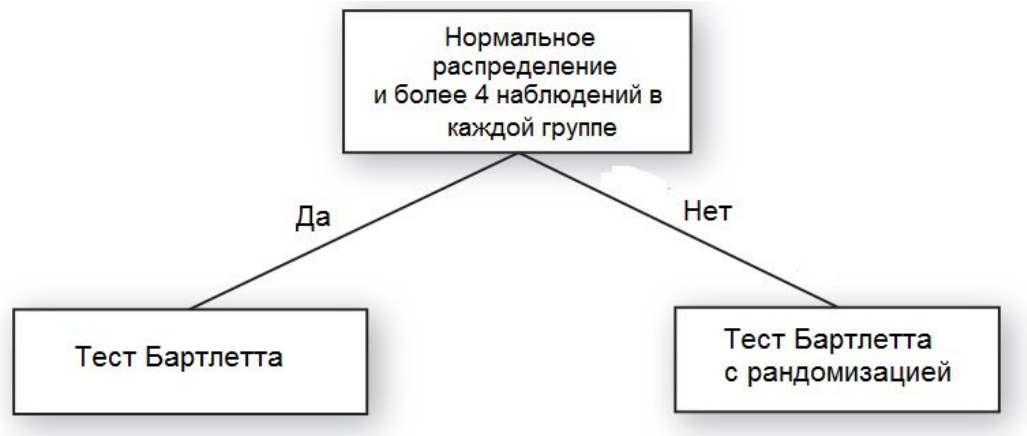
Примечание: феномен, когда несколько дисперсий не отличаются между собой, называется "однородностью вариации" или "гомоскедастичностью".

Тест Фишера (параметрический). Чтобы осуществить тест, используют команду: `var.test(отклик~фактор)` где `отклик` – вектор, содержащий значения анализируемой переменной величины, и `фактор` – вектор, содержащий сопутствующие условия (например, порядковый номер серии) для каждого наблюдения в том же порядке, что и `отклик`. Символ `~` означает "*в зависимости от*".

Тест Фишера с рандомизацией (непараметрической). Чтобы осуществить тест, используется функция: `perm.var.test (отклик~фактор)*`.

68. Сравнение более чем двух дисперсий

Чтобы выбрать подходящий тест:



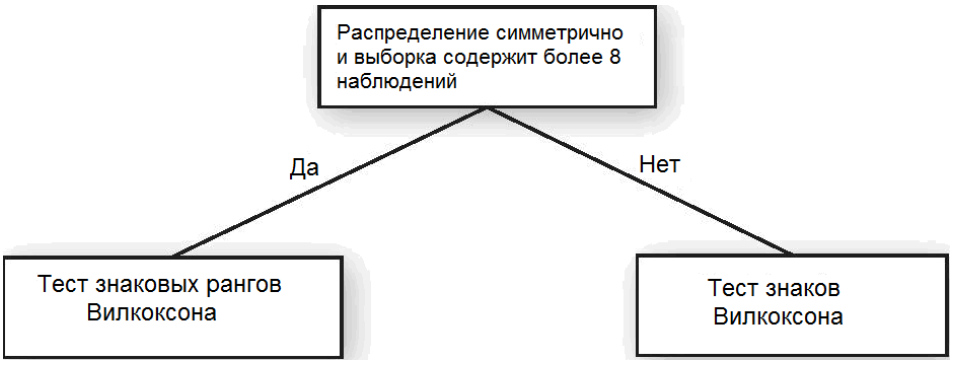
Тест Бартлетта (параметрический). Чтобы осуществить тест, используют команду: `bartlett.test (отклик~фактор)`, где `отклик` – вектор, содержащий значения анализируемой переменной величины, и `фактор` – вектор, содержащий сопутствующие условия (например, порядковый номер серии) для каждого наблюдения в том же порядке, что и `отклик`. Символ `~` означает "*в зависимости от*".

Тест Бартлетта с рандомизацией (непараметрической). Чтобы осуществить тест, используется функция: `perm.bartlett.test (отклик~фактор)*`.

Простые тесты, касающиеся медиан

69. Соответствие медианы теоретическому значению

Чтобы выбрать подходящий тест:



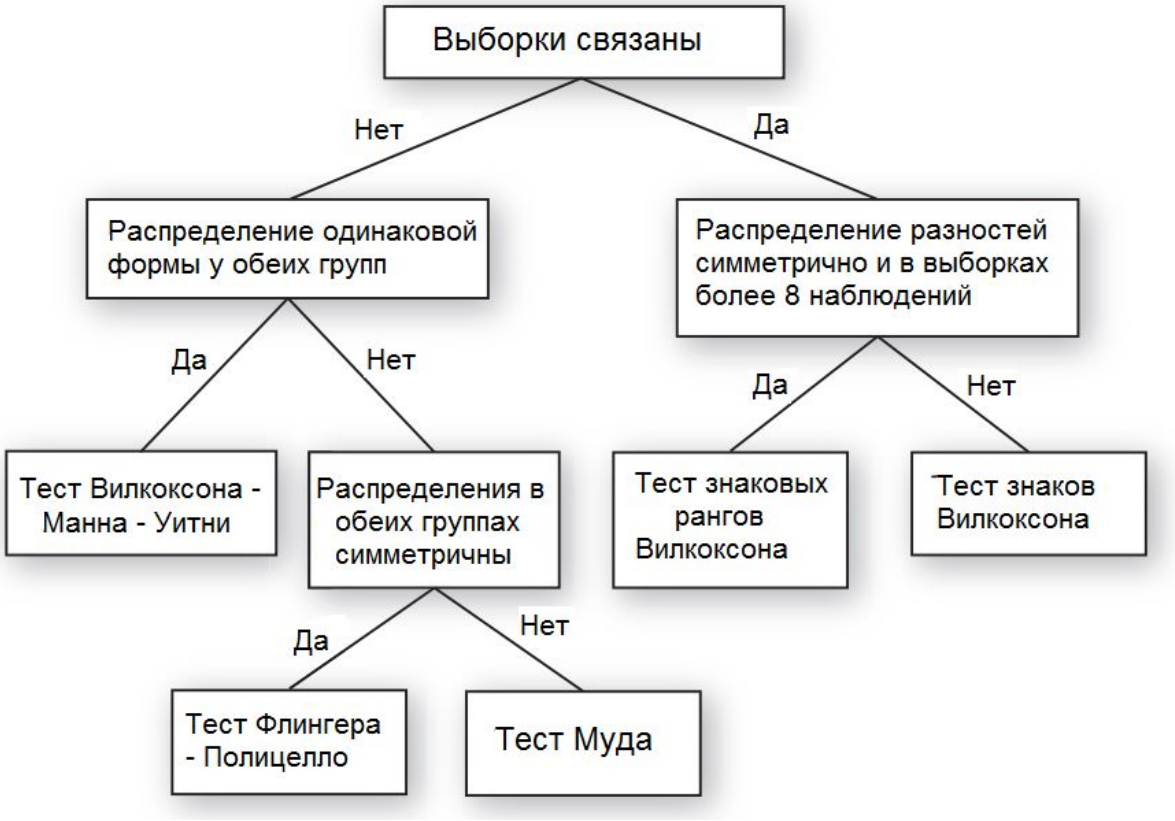
Замечание: симметричный характер распределения проверяется с использованием гистограммы (см. п. **31**). При этом неважно, является ли распределение, одно- или полимодальным.

Тест знаковых рангов Вилкоксона (непараметрический). Чтобы выполнить тест, подают команду: `wilcox.test(выборка, mu=m.theo)`, где `выборка` – вектор, содержащий серию данных и `m.theo` – теоретическая медиана.

Тест знаков Вилкоксона (непараметрический). Чтобы выполнить тест, используется функция: `wilcox.signtest(выборка, mu=m.theo)*`.

70. Сравнение двух медиан

Чтобы выбрать подходящий тест:



Замечание: общая форма и симметричный характер распределения проверяются с использованием гистограммы (см. п. **31**). Для теста Вилкоксона-Манна-Уитни распределение обеих выборок должно иметь почти одну и ту же форму (в том числе, одинаковую дисперсию, см. п. **66** для иллюстрации). Для теста Флингера - Полицелло неважно, одинаковы ли оба типа распределения; лишь бы только они были симметричны для каждой из групп. Для тестов со связанными выборками важна только форма распределения *разностей между связанными значениями*. Эти разности могут быть получены командой

```
differences <- отклик[as.numeric(фактор)==1] -
                        отклик[as.numeric(фактор)==2],
```

где `отклик` – вектор, содержащий значения переменной величины, и `фактор` – вектор, определяющий номер выборки для каждого наблюдения. Связанные значения должны быть в одном и том же порядке для обеих серий.

Несвязанные выборки

Тест Вилкоксона-Манна-Уитни (непараметрический). Для выполнения теста используется команда: `wilcox.test(отклик~фактор)*`. Символ `~` означает "*в зависимости от*".

Тест Флигнера - Полицелло (непараметрический). Чтобы выполнить тест, используется функция: `fp.test(отклик~фактор)*`.

Тест Муда (непараметрический). Чтобы выполнить тест, используется функция: `mood.medtest(отклик~фактор)*`.

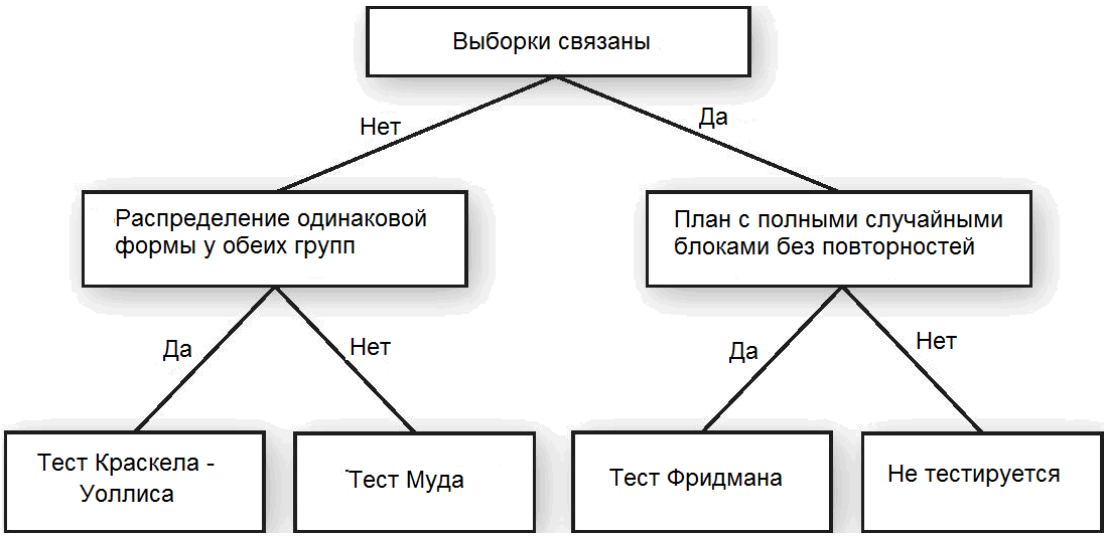
Связанные выборки

Тест знаковых рангов Вилкоксона (непараметрический). Чтобы выполнить тест, подают команду: `wilcox.test (отклик~фактор, paired=TRUE)`.

Тест знаков Вилкоксона (непараметрический). Чтобы выполнить тест, используется функция: `wilcox.signtest(отклик~фактор)*`.

71. Сравнение более чем двух медиан

Чтобы выбрать подходящий тест:



Замечание: общая форма и симметричный характер распределения проверяются с использованием гистограммы (см. п. **31**). Для теста Краскела-Уоллиса распределение обеих выборок должно иметь почти одну и ту же форму (в том числе, одинаковую дисперсию, см. п. **66** для иллюстрации). Посмотрите п. **13** для пояснения того, чем является план эксперимента с полными случайными блоками без повторностей.

Несвязанные выборки

Тест Краскела-Уоллиса (непараметрический). Для выполнения теста используется команда: `kruskal.test(отклик~фактор)`, где `отклик` – вектор, содержащий значения анализируемой величины, и `фактор` – вектор, содержащий условия выполнения каждого наблюдения. Символ ~ означает "*в зависимости от*".

Если получено статистически значимая величина *p*, то это указывает, что, по крайней мере, медианы двух выборок различаются между собой (не уточняется, какие именно). В этом случае при необходимости можно осуществить парные сравнения, чтобы идентифицировать уровни фактора, оказывающие влияние на отклик, используя тест Данна, реализованный функцией: `dunn.test(отклик~фактор)*`.

Тест Муда (непараметрический). Чтобы выполнить тест, используется функция: `mood.medtest(отклик~фактор)*`. Если *p*-значение статистически значимо, то можно выполнить парные сравнения, используя функцию `pairwise.mood.medtest(отклик, фактор)*`.

Связанные выборки

Тест Фридмана (непараметрический). Для выполнения теста используется команда: `friedman.test(отклик~факт.фикс|факт.случ)`, где `факт.фикс` и `факт.случ` – векторы, содержащие значения фиксированного и случайного факторов

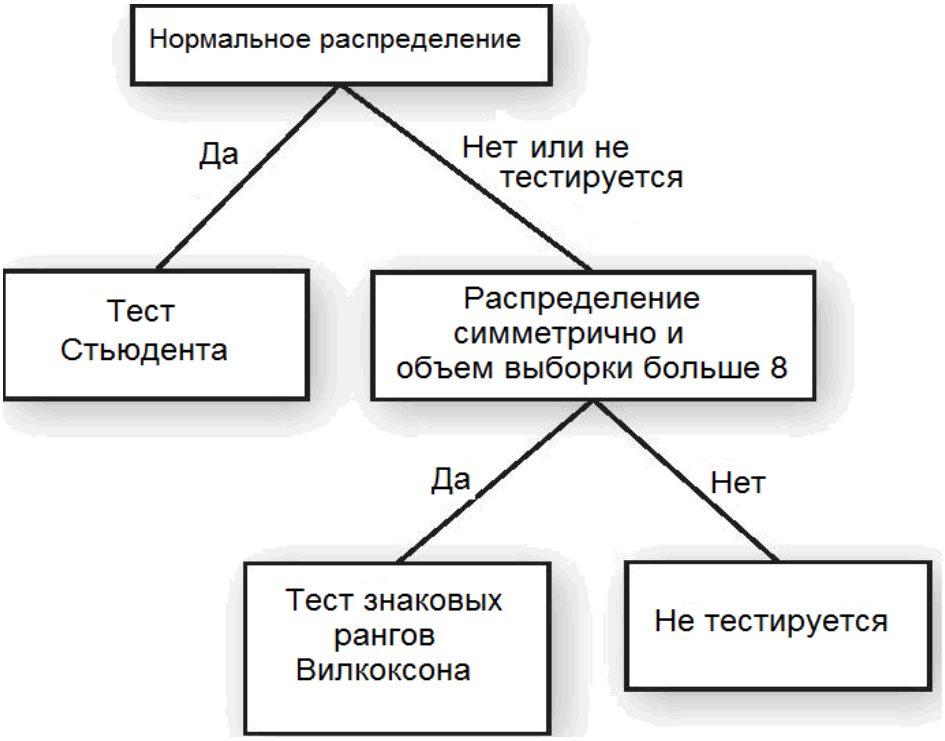
(в том же порядке, что и отклик), которые мы хотим сравнить, сопоставляя серии взаимосвязанных наблюдений.

Если *p*-значение статистически значимо, то можно выполнить парные сравнения, используя функцию
`wilcox.paired.multcomp (отклик~факт.фикс|факт.случ)*`.

Простые тесты, касающиеся средних

72. Соответствие среднего значения теоретическому значению

Чтобы выбрать подходящий тест:



Замечания: нормальность распределения проверяется с использованием описанных методов (см. п. **65**), а его симметричный характер оценивается путем построения гистограммы (см. п. **31**). Для теста знаковых рангов Вилкоксона неважно, является ли выборочное распределение одно- или полимодальным.

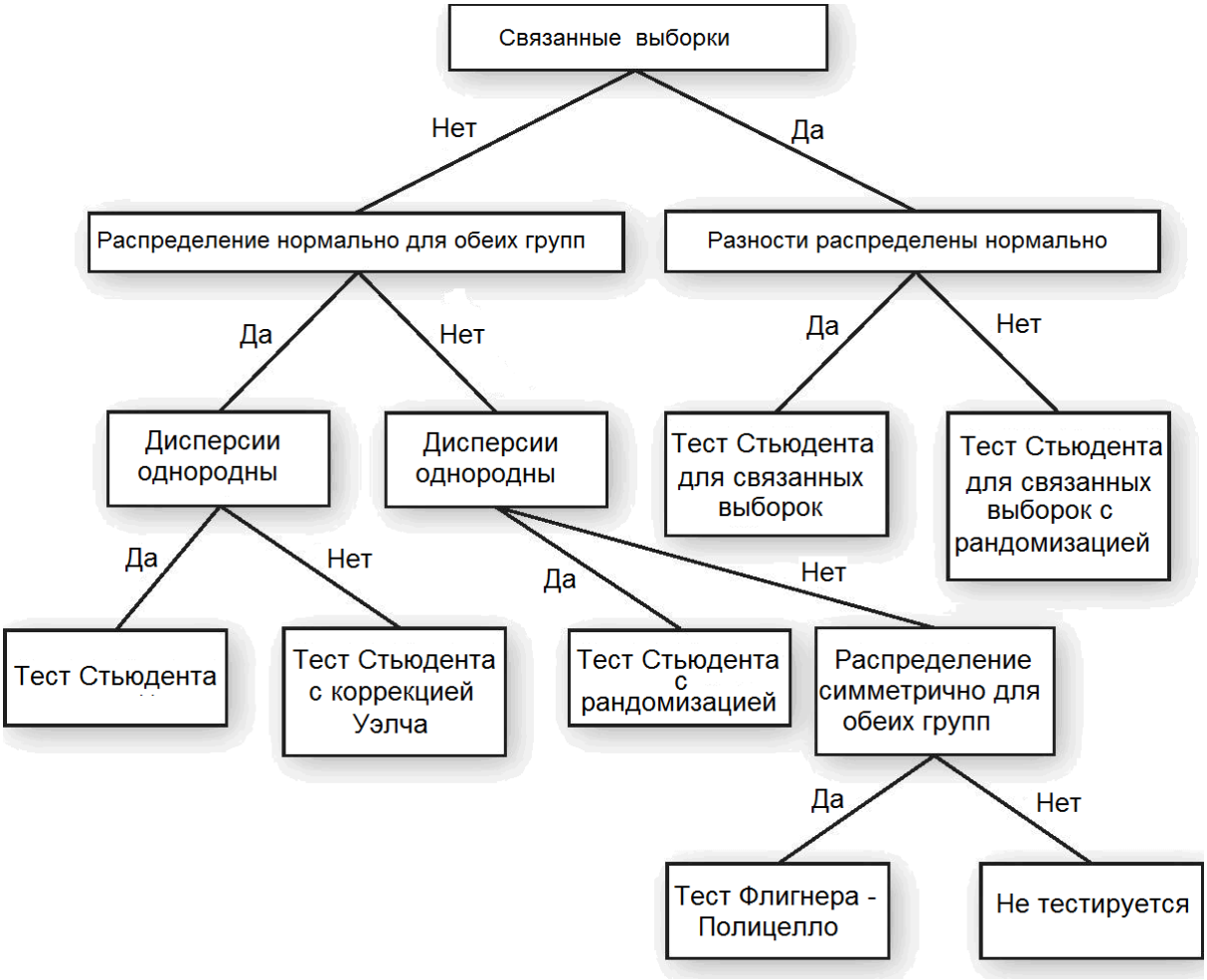
Тест Стьюдента (параметрический). Чтобы выполнить тест, подают команду: `t.test(выборка, mu=m.theo)`, где `выборка` – вектор, содержащий серию данных и `m.theo` – теоретическое среднее.

Тест знаковых рангов Вилкоксона (непараметрический). Чтобы выполнить тест, подают команду: `wilcox.test (выборка, mu=m.theo)`.

Замечание: этот тест фактически напрямую не использует понятия среднего. Но, если распределение данных симметрично, заключение теста может быть применено к среднему.

73. Сравнение двух средних

Чтобы выбрать подходящий тест:



Замечание: нормальность распределения проверяется с использованием описанных методов (см. п. **65**) , а его симметричный характер оценивается путем построения гистограммы (см. п. **31**). Для теста Флингера - Полицелло неважно, одинаковы ли оба типа распределения; лишь бы только они были симметричны в каждой из групп. Для тестов со связанными выборками важна только форма распределения *разностей между связанными значениями*. Эти разности могут быть получены командой

```
differences <- отклик[as.numeric(фактор)==1] -
                        отклик[as.numeric(фактор)==2],
```

где `отклик` – вектор, содержащий значения переменной величины, и `фактор` – вектор, определяющий номер выборки для каждого наблюдения. Связанные значения должны быть в одном и том же порядке для обеих серий.

Несвязанные выборки

Тест Стьюдента (параметрический). Для выполнения теста используется команда: `t.test(отклик~фактор, var.equal=TRUE)`. Символ `~` означает "*в зависимости от*".

Тест Стьюдента с коррекцией Уэлча (параметрический). Для выполнения теста используется команда: `t.test (отклик~фактор, var.equal=FALSE)`, или проще `t.test(отклик~фактор)`, т.е. коррекция Уэлча выполняется по умолчанию.

Тест Стьюдента с рандомизацией (непараметрический). Чтобы осуществить тест, можно использовать функцию: `perm.t.test(отклик~фактор)*`.

Тест Флигнера - Полицелло (непараметрический). Чтобы осуществить тест, можно использовать функцию: `fp.test (отклик~фактор)*`.

Замечание: этот тест фактически сравнивает *медианы* обеих выборок. Но, если распределение данных симметрично в каждой из групп, то медиана и среднее очень близки и заключение теста может быть применено к среднему.

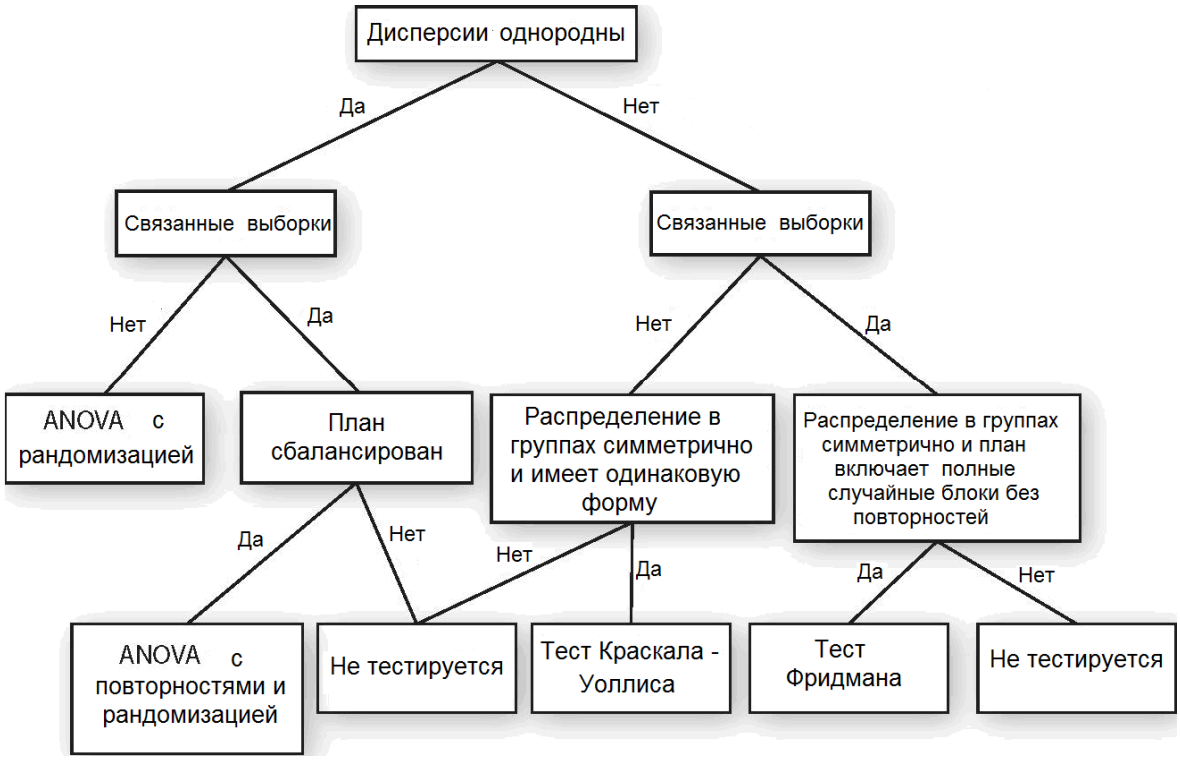
Связанные выборки

Тест Стьюдента для связанных выборок (параметрический). Для выполнения теста используется команда: `t.test(отклик~фактор, paired=TRUE)`.

Тест Стьюдента для связанных выборок с рандомизацией (непараметрический). Чтобы осуществить тест, можно использовать функцию: `perm.t.test(отклик~фактор)*`.

74. Сравнение более чем двух средних (однофакторный дисперсионный анализ)

Чтобы выбрать подходящий тест:



Замечание: общая форма и симметричный характер распределения проверяются с использованием гистограммы (см. п. **31**). Для теста Краскела-Уоллиса распределение обеих выборок должно иметь почти одну и ту же форму (в том числе, одинаковую дисперсию, см. п. **66** для иллюстрации). Для ANOVA с повторными измерениями и использованием рандомизации "сбалансированный план" означает одинаковое число наблюдений в каждом сочетании уровней фиксированного и случайного факторов. Посмотрите п. **13** для пояснения того, чем является план эксперимента с полными случайными блоками без повторностей.

Дисперсионный анализ ANOVA (англ. *analysis of variance*) на самом деле является тестом на основе модели. В этом качестве он представлен в п.**76**.Тестирование на основе модели.

Несвязанные выборки

Тест ANOVA с рандомизацией (непараметрический). Для выполнения теста можно использовать функцию: `perm.anova(отклик~фактор)*`, где `отклик` – вектор, содержащий значения переменной величины, и `фактор` – вектор, содержащий условия выполнения каждого наблюдения (в том же порядке).

Если получено статистически значимая величина *p*, то это указывает, что, по крайней мере, средние двух выборок различаются между собой (не уточняется, какие именно). При необходимости в этом случае можно осуществить парные сравнения, чтобы идентифицировать уровни фактора, оказывающие влияние на отклик, используя функцию: `pairwise.perm.t.test(отклик~фактор)*`.

Может случиться, что парные сравнения не указывают никаких значимых различий, вопреки общему тесту. В этом случае, наиболее осторожное решение состоит в том, чтобы считать, что не можем знать, какие средние ответственны за отклонение от нулевой гипотезы в общем тесте.

Тест Краскела-Уоллеса (непараметрический). Для выполнения теста используется команда: `kruskal.test(отклик~фактор)`. При значимом *p* можно осуществить парные сравнения, используя тест Данна, реализованный функцией:
`dunn.test(отклик~фактор)*`.

Замечание: тест Краскела-Уоллеса и тест Данна фактически сравнивают *медианы* анализируемых выборок. Но, если распределение данных симметрично в каждой из групп, то медиана и среднее очень близки и заключение теста может быть применено и к сравнению групповых средних.

Связанные выборки

Тест ANOVA с повторными измерениями и использованием рандомизации (непараметрический). Чтобы осуществить тест, можно использовать функцию: `perm.anova(отклик~факт.фикс|факт.случ)*`, где `факт.фикс` и `факт.случ` – векторы, содержащие значения фиксированного и случайного факторов (в том же порядке, что и `отклик`), которые мы хотим сравнить, сопоставляя серии взаимосвязанных наблюдений.

Если *p*-значение статистически значимо, то можно выполнить парные сравнения, используя функцию
`pairwise.perm.t.test(отклик~факт.фикс|факт.случ)*`.
Обратите внимание, что связанные значения должны быть в одном и том же порядке для всех уровней `факт.фикс`.

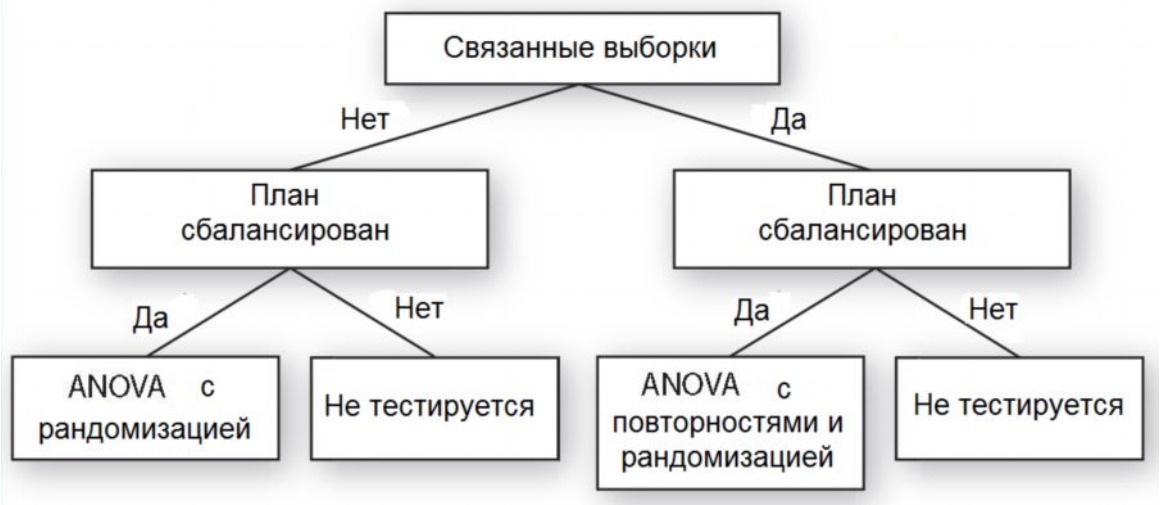
Тест Фридмана (непараметрический). Для выполнения теста используется команда:
`friedman.test (отклик~факт.фикс|факт.случ)`.

Если *p*-значение статистически значимо, то можно выполнить парные сравнения, используя функцию
`wilcox.paired.multcomp(отклик~факт.фикс|факт.случ)*`.

Замечание: тест Фридмана и тест ранговых знаков Вилкоксона фактически сравнивают *медианы* анализируемых выборок. Но, если распределение данных симметрично в каждой из групп, то медиана и среднее очень близки и заключение теста может быть применено и к сравнению групповых средних.

75. Двухфакторный дисперсионный анализ

Чтобы выбрать подходящий тест:



Замечание: "сбалансированный план" включает одно и то же количество наблюдений для каждого сочетания всех факторов, т.е. двух факторов для ANOVA с рандомизацией и трех (включая случайный фактор) для ANOVA с повторными измерениями и использованием рандомизации.

Несвязанные выборки

Тест ANOVA с рандомизацией (непараметрический). Для выполнения теста можно использовать функцию: `perm.anova(формула)*`, где `формула` может принимать следующие выражения:

- `отклик ~ фактор1 + фактор2`
- `отклик ~ фактор1 * фактор2`
- `отклик ~ фактор1/фактор2` – для модели, где оба фактора фиксированы;
- `отклик ~ фактор1/фактор2` – добавляя аргумент `nest.f2 ="Random"` для модели, где `фактор2` – случайный фактор (см. п. **40** для подробного объяснения конструкции формул).

Если получена статистически значимая величина *p*, то это указывает, что, по крайней мере, средние двух групп различаются между собой (не уточняется, какие именно). При необходимости в этом случае можно осуществить парные сравнения, чтобы идентифицировать комбинации переменных, оказывающих влияние на отклик, используя функцию: `pairwise.perm.t.test(отклик, фактор)*`, где `фактор` – отдельные факторы или их взаимодействия, статистически значимые в соответствии с общим тестом.

Может случиться, что парные сравнения не показывают никаких значимых различий, вопреки общему тесту. В этом случае, наиболее осторожное решение состоит в том, чтобы считать, что не можем знать, различия каких групповых средних определяют отклонение нулевой гипотезы в общем тесте.

Связанные выборки

Тест ANOVA с повторными измерениями и использованием рандомизации (непараметрический). Чтобы осуществить тест, можно использовать функцию `perm.anova(формула)*`, где `формула` может принимать следующие выражения:

- `отклик ~ факт.фикс1 + факт.фикс2|факт.случ`
- `отклик ~ факт.фикс1 * факт.фикс2|факт.случ`

Если *p*-значение статистически значимо, то можно выполнить парные сравнения функцией `pairwise.perm.t.test(отклик~фактор, paired=TRUE)*`, где `фактор` – отдельные факторы или их взаимодействия, статистически значимые в соответствии с общим тестом. Обратите внимание, что связанные значения должны быть в одном и том же порядке для всех уровней обоих факторов `факт.фикс`.

Построение моделей

76. Анализ неограниченной непрерывной переменной – линейная зависимость

Обсуждаемые здесь процедуры анализа используют основные понятия построения моделей, изложенные в п. **39- 42**.

Типы модели

Хотя основные принципы остаются одними и теми же, некоторые технические аспекты анализа отличаются в зависимости от того, связан ли отклик с другими факторами, определяющими некоторые важные для исследования условия эксперимента. Если связанный фактор отсутствует, то используется Линейная Модель (*Linear Model* или LM), в то время, как при наличии связанных факторов - Смешанная Линейная Модель (*Linear Mixed Model* или LMM). Термин "*смешанный*" подразумевает наличие, по крайней мере, одного случайного фактора, который используется, чтобы точно идентифицировать серии наблюдений (см. п. **11**).

Примечание: если все объясняющие переменные являются числовыми величинами (ковариатами), мы имеем частный случай *линейной регрессии* (с использованием обыкновенного метода наименьших квадратов).

Построение модели

Для создания модели используют следующие функции:

- без связанных случайных факторов `модель <- lm(формула)`
- с связанными качественными переменными:
  `модель <- lmer(формула)` из пакета `lme4`

См. п. **40** для подробного объяснения построения формулы. В целом можно напомнить, что:

- включение независимого фактора позволяет проверить, имеется ли вариация отклика между уровнями этого фактора;
- включение ковариаты позволяет проверить, существует связь между этой ковариатой и объясняемой переменной;
- включение взаимодействия двух независимых переменных позволяет проверить, зависит ли один эффект от другого (это же рассуждение - для наблюдений, приуроченных к двум и более отрезкам времени). Особый случай – анализ взаимодействия между ковариатой и фактором, который позволяет проверить, зависит ли отношение между независимой переменной и откликом от уровней фактора.

Проверка адекватности модели

Прежде чем проводить дальнейший анализ, необходимо проверить, насколько хорошо модель подогнана к анализируемым данным. Этот этап фундаментален для любой модели, так как любой тест, проводимый на недостаточно адекватной модели, просто ненадежен (или не соответствует действительности). См. п. **41** для подробного

объяснения методики этой проверки.

Если подгонка модели выполнена не совсем хорошо, наиболее простой способ разрешения проблемы состоит в том, чтобы изменить шкалу представления объясняемой случайной величины. Для непрерывных данных классическими являются три опции преобразований: $\sqrt{x}$, $log(x)$ или $\sqrt[4]{x}$, если трансформации квадратного корня оказалось недостаточно, а нули в данных препятствуют логарифмированию.

Способность модели к объяснению

Можно оценить общую объяснясняющую ценность модели благодаря *коэффициенту детерминации* (R^2), который представляет собой долю от общей вариации отклика, которая объяснена независимыми переменными модели.

Для линейной модели он включается в общий протокол: `summary(модель)$r.squared`. Для смешанной линейной модели этот коэффициент может быть получен с использованием функции `r.squaredGLMM(модель)` из пакета `MuMIn`. Функция фактически возвращает два значения: *маргинальный* R^2 (`R2m`), который соответствует доле вариации, объясненной только ковариатами и фиксированными факторами, и *условный* R^2 (`R2c`), который соответствует доле вариации, объясненной всеми предикторами (т.е. как фиксированными, так и случайными факторами).

Тестирование

Для обеих моделей значимость объясняющих переменных оценивается функцией: `Anova(модель)` из пакета `car` (см. п. **42** для подробного объяснения процесса тестирования гипотез). Однако этот тест для разных моделей имеет характерные особенности:

- для линейной модели LM функция выполняет тест по *F*-критерию, который, если все объясняющие переменные являются факторами, состоит в анализе групповых дисперсий ANOVA (*analysis of variance*);
- для смешанной модели LMM функция выполняет тест Вальда и анализирует отдельно каждый член модели (т. е. по строкам формируемой таблицы с оценками коэффициентов).

Примечание: хотя термин "ANOVA" часто и повсеместно используется, он фактически соответствует единственному очень частному случаю: *F*-тест для линейной модели с использованием обыкновенных наименьших квадратов, где все объясняющие переменные величины являются факторами.

Если один из факторов (или взаимодействие с участием фактора) имеет статистически значимый эффект, необходимо осуществить множественное парное сравнение его уровней (или сочетаний этих уровней), чтобы выявить, в каких комбинациях имеются различия. Посмотрите п. **43** как выполнить эти сравнения.

Если ковариата имеет значимое влияние, направление эффекта определяется знаком коэффициента, который соответствует *углу наклона* зависимости между независимой переменной и откликом. Если знак коэффициента для количественной переменной отрицателен, то значение отклика уменьшается, когда значение ковариаты увеличивается; а если он положителен, то отклик увеличивается, когда значение ковариаты увеличивается.

Полную информацию о всех параметрах модели можно получить с помощью команды `summary(модель)`. Значения самих коэффициентов называются `Estimate` и находятся в таблице `Coefficients` для LM и `Fixed effects` для LMM. Стандартная ошибка всех коэффициентов дана в столбце `Std. Error`.

Замечание: в случае простой линейной регрессии (т.е. только с одной ковариатой в качестве объясняющей переменной), представляет интерес, какую величину принимает ордината при нулевом значении предиктора модели. Она дано в строке `(Intercept)`.

Прогноз на основе модели

Смысл моделей – не только показать роль различных независимых переменных в изменчивости объясняемой переменной, но и предсказать значение, которое принял бы отклик при известных значений независимых переменных. Поэтому, чтобы оценить значение отклика, требуется зафиксировать значение всех переменных.

Для прогнозирования могут быть использованы два метода (для LMM доступен только второй), и оба они основаны на функции `predict()`:

- значение каждой из независимых переменных дается непосредственно в теле функции в виде списка (с одной или несколькими записями): `predict(модель,newdata=list(переменные))`, где `переменные` – это последовательность `var1= значение, var2=значение` и т.д.;
- создать таблицу, содержащую столбцы по каждой независимой переменной (имена столбцов должны строго совпадать с именами переменных модели), и заполнить каждую строку значениями, для которых должно быть сделано предсказание. Таким образом, делается прогноз по каждой строке таблицы:

```
predict(модель, newdata = таблица).
```

Примечание: в случае LMM некоторые значения случайных факторов могут не иметь данных. Если этого не произошло, то необходимо добавить в функцию `predict()` аргумент `re.form=NA`. Тогда при оценке прогноза будут приниматься, во внимание все эффекты случайных факторов модели.

ПРИМЕРЫ

Рассмотрим модель, включающую `фактор`, варьируемый на двух уровнях (`A` и `B`) , непрерывную величину `ковариата`, изменяющуюся от 0 до 30, а также их взаимодействие:

```
>    модель <- lm(отклик~фактор*ковариата)
```

Можно предсказать отклик для конкретных условий таким образом:

```
>    predict (модель, newdata=list (фактор = " A ",
                ковариата =10))
```

Или для нескольких значений предикторов

```
>    predict (модель, newdata=list (фактор =c (" A ", " B "),
     ковариата =c (10,10)))
>    predict (модель, newdata=list (фактор =c (" A ", " B "),
     ковариата =rep (10,2)))
```

Или создать таблицу нужного типа и использовать ее в дальнейшем:

```
 > таблица
     фактор ковариата
 1      A         10
 2      B         15
 > predict (модель, newdata= таблица)
```

Графики

Влияние фактора. Чтобы проиллюстрировать результат воздействия фактора, могут быть представлены два типа средних:

- средние брутто (т.е. вычисляемые по исходным данным) и их стандартные ошибки (см. пп. **35** и **37**, как использовать функцию `tapply()`).

Внимание: в случае LMM эти средние и стандартные ошибки не учитывают случайные факторы.

- средние значения, которые были скорректированы с учетом других переменных модели, а также их стандартные ошибки (см. п. **43**).

После вычисления средних и стандартных ошибок может быть изображен и сам график (см. п. **34**).

Зависимость от ковариаты. Чтобы проиллюстрировать связь между объясняемой переменной и независимой ковариатой, требуется три этапа:

1. Вывести на график точки наблюдений `plot(отклик ~ ковариата)`.

2. Создать вектор, имеющий те же минимум и максимум, что `ковариата`, но с очень маленьким интервалом между его значениями `x <- seq2(ковариата)`*.

3. Добавить кривую модельных значений на график.

Определение всех точек модельной кривой основано на прогнозе значений, которые принимает объясняемая переменная для каждого значения вектора `x`. Большое количество значений этого вектора и очень маленький интервал, который их отделяет, определяет гладкость кривой (которая на самом деле состоит из сотен точек, соединенных между собой). Если модель содержит другие независимые переменные, то к прогнозу необходимо добавить одинаковые значения, такие, чтобы величина отклика изменяла свою величину только при изменении анализируемой ковариаты.

Кривая на график может быть добавлена функцией

`lines(x, predict(модель,newdata= x))` .

Примечание: функция `lines()` может быть использована несколько раз, чтобы начертить несколько кривых на одном графике, например, для нескольких уровней одного и того же фактора, включенного в модель. Чтобы изменить тип линий графика и цвет кривых, можно использовать аргументы `lty` и `col` соответственно. Наконец, чтобы добавить заголовки, используйте функцию `legend()`. Смотрите справку `?` для получения дополнительных сведений об используемых параметрах и знакомства с различными вариантами графики.

ПРИМЕРЫ

Рассмотрим модель, представленную выше :

```
>    модель <-lm (отклик~фактор*ковариата)
```

Шаг 1 : нарисуем точки, соответствующие наблюдаемым данным :

```
> plot(отклик ~ ковариата)
```

Шаг 2 : создадим вектор `x` :

```
> x <- seq2(ковариата)
```

Шаг 3 : добавляем модельную кривую.

Выбираем один из уровней фактора `A`, включенного в модель:

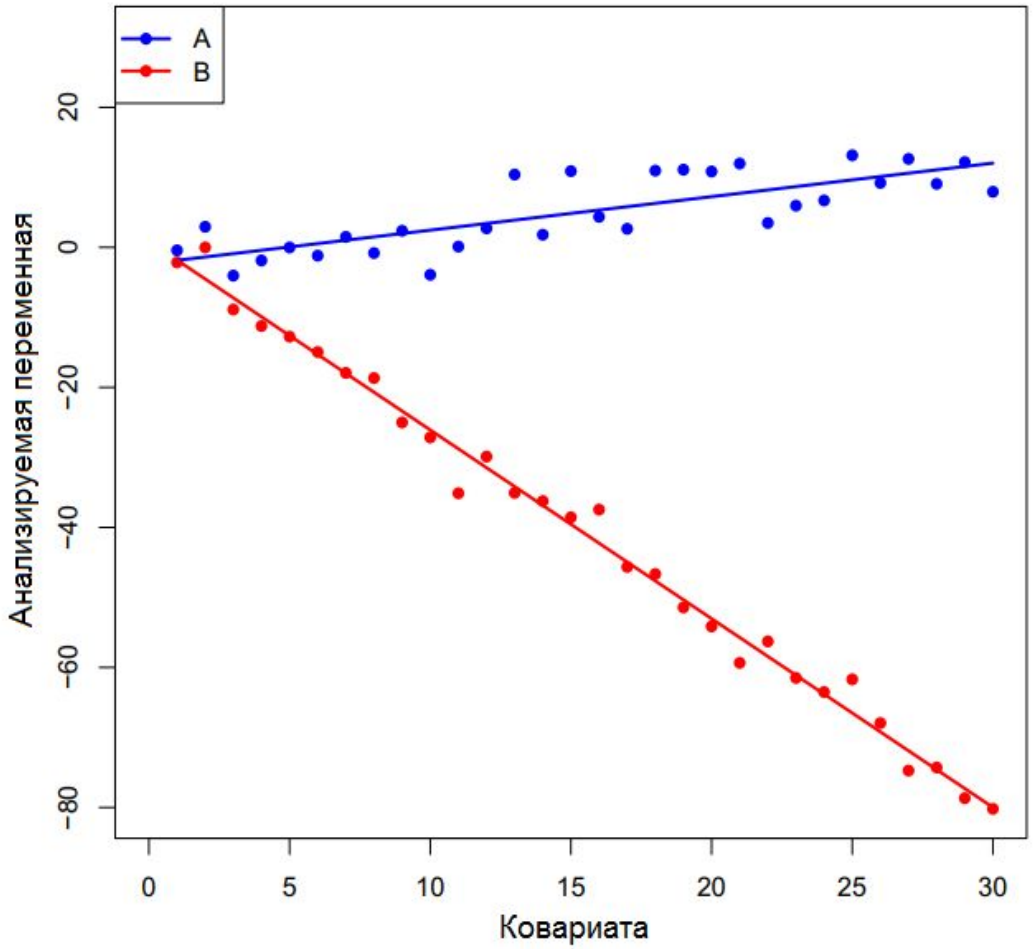
```
> lines(x,predict(модель,newdata=list(ковариата = x,
     фактор =rep (A,length(x)))), col = 2)
```

Функция `rep()` является очень полезной в этой ситуации, поскольку она позволяет очень просто создать вектор той же длины, что и `x`, содержащий одно и тоже повторяющееся значение `A`.

Чтобы добавить модельную кривую, соответствующую уровню `B` фактора :

```
> lines(x,predict(модель,newdata=list(ковариата = x,
          фактор =rep (B,length(x)))))
```

На полученном рисунке синим цветом представлена модельная прямая линейной регрессии для уровня `A` воздействующего фактора, а красным цветом – уровня `B`.



77. Анализ неограниченной непрерывной переменной – нелинейная зависимость

Типы модели

Используемая модель – Нелинейная Модель (*Non Linear Model* или NLM). Она используется, как правило, с непрерывными объясняющими переменными величинами и этот частный случай называется *нелинейной регрессией*. В модель может быть добавлен фактор, чтобы определить различные параметры условий наблюдения.

Построение модели

Созданиние NLM требует двух условий:

- необходимо иметь предположения о структуре уравнения, которое связывает объясняющую переменную величину с откликом;
- знать *априори* ориентировочные оценки параметров этого уравнения (большая часть времени может быть затрачена, чтобы составить себе такое представление, начиная с построения графиков).

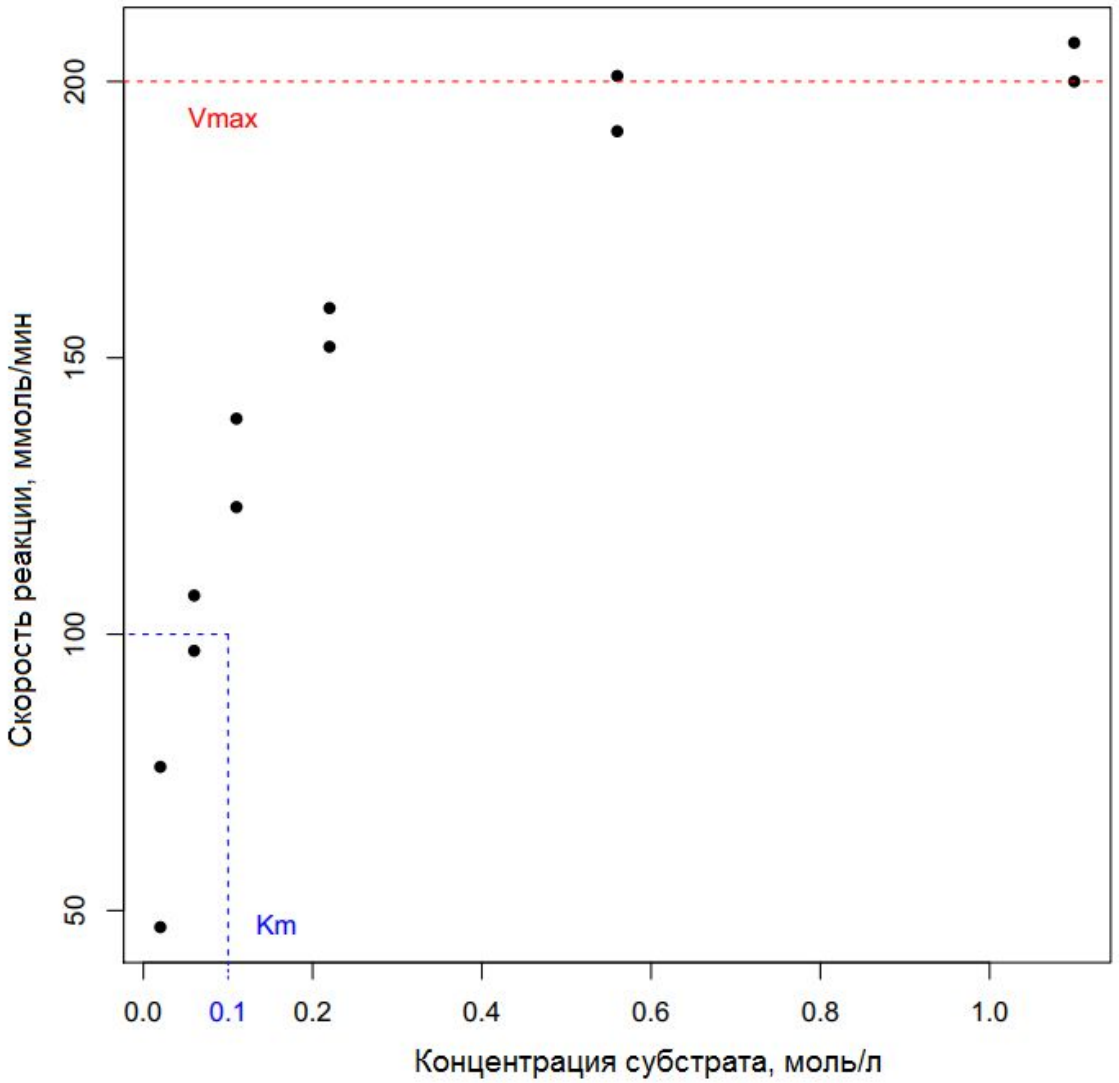
Модель может быть получена с использованием команды:
`модель <- nls(отклик~уравнение, start=list(parametres))`, где `уравнение` – некоторая явная нелинейная функция, определяющая связь предикторов с откликом, и `parametres` – список параметров уравнения с указанием для каждого из них начального приближенного значения. Символ `~` означает "*в зависимости от*".

ПРИМЕР

Уравнение Михаэлиса–Ментен определяет зависимость скорости биохимических реакций (далее `Vitesse`), катализируемой ферментами (в нашем случае энзимами `enzyme`), от концентрации субстрата (`concentration`). Модель имеет два параметра: V_{max}максимальную скорость реакции и K_M – концентрацию субстрата, при которой скорость реакции составляет половину от максимальной (константа Михаэлиса). В этих обозначениях уравнение имеет форму:

$$Vitesse = \frac{V_{\max} \cdot concentration}{K_M + concentration}.$$

Были получены следующие экспериментальные значения:



С использованием графика оцениваем V_{max} = 200 ммоль/мин и K_M = 0,1 моль/л. Следовательно, команда для построения модели запишется:

```
> модель <- nls(Vitesse~Vmax* concentration/
          (Km+concentration), start=list(Vmax = 200, Km=0.1))
```

Некоторые распространенные зависимости описаны в **R** специальными функциями, которые позволяют одновременно однозначно определять структуру уравнения, а также (и главным образом) автоматически *априори* оценивать стартовые значения его параметров:

Тип регрессии	Уравнение	Функция R
Асимптотический		
Михаэлиса–Ментен	$y = \frac{V_m \cdot x}{K + x}$	`SSmicmen(x,Vm,K)`
Экспонента с 2 параметрами	$y = Asym(1 - e^{-e^{lrc}x})$	`SSasympOrig(x,Asym, lrc)`
Экспонента с 3 параметрами	$y = Asym + (R_0 - Asym)e^{-e^{lrc}x}$	`SSasymp(x,Asym,RO,lrc)`
Сигмоидный		
Логистический с 3 параметрами	$y = \frac{Asym}{1+e^{\frac{xmid-x}{scal}}}$	`SSlogis(x,Asym,xmid, seal)`
Логистический с 4 параметрами	$y = A + \frac{B-A}{1+e^{\frac{xmid-x}{scal}}}$	`SSfpl(x,A,B,xmid, seal)`
Вейбулла	$y = Asym - Drop \cdot e^{-e^{lrc}x^{pwr}}$	`SSweibull(x,Asym,Drop, lrc,pwr)`
Гомпертца	$y = Asym \cdot e^{-b2 \cdot b3^x}$	`SSgompertz(x,Asym,b2, b3)`
Прочие		
Биэкспоненциальная	$y = A1 \cdot e^{-e^{lrc1}x} + A2 \cdot e^{-e^{lrc2}x}$	`SSbiexp(x,Al,lrcl, A2,lrc2)`

Использование этих функций значительно упрощает конструкцию модели. Для построения той же модели Михаэлиса – Ментен эквивалентна запись:

```
> modele <- nls(vitesse~SSmicmen(concentration, Vmax, Км))
```

Чтобы оценивать различные параметры для каждого уровня заданного фактора, вместо `nls()` используют функцию из пакета `nlme`: `nlsList()`. Обе функции основаны на одном и том же синтаксисе, но во втором случае добавляется член `уравнение|фактор`, где `фактор` – вектор, содержащий условия каждого наблюдения (в том же порядке, что и остальные переменные).

ПРИМЕР

Пусть по той же модели Михаэлиса – Ментен был осуществлен опыт с двумя различными энзимами и мы хотим оценить различные параметры для каждого из них:

```
> модель <- nlsList(vitesse~Vmax*concentration/
     (Km+concentration)|enzyme, start=list(Vmax=200, Km=0.1))
```

или:

```
> модель <- nlsList(vitesse~
                    SSmicmen(concentration,Vmax,Km)|enzyme)
```

Проверка адекватности модели

Прежде чем проводить дальнейший анализ, необходимо проверить, насколько хорошо модель подогнана к анализируемым данным. Этот этап фундаментален для любой модели, так как любой тест, проводимый на недостаточно адекватной модели, просто ненадежен (или не соответствует действительности). См. п. **41** для подробного объяснения методики этой проверки.

Оценка значимости коэффициентов и тестирование

Значения всех параметров модели получены командой `summary (модель)`. Они оказываются в таблице `Parameters` для модели без фактора (т.е. созданной `nls()`),

или Coefficients для модели с фактором (т.е. созданной nlsList()). В обоих случаях таблица включает значение (Estimate) и стандартную ошибку (Std. Error) каждого параметра, и тест его уровня значимости. Если *p*-значение не является значимым, то это указывает на то, что параметр может быть исключен из уравнения модели.

Прогноз на основе модели

Смысл моделей – не только показать роль различных независимых переменных в изменчивости объясняемой переменной, но и предсказать значение, которое бы принял отклик при известных значений независимых переменных. Поэтому, чтобы оценить значение отклика, требуется зафиксировать значение всех переменных.

Для прогнозирования могут быть использованы два метода и оба они основаны на функции predict():

- значение каждой из независимых переменных дается непосредственно в теле функции в виде списка (с одной или несколькими записями): predict(модель,newdata=list(переменные)), где переменные – это последовательность var1= значение,var2=значение и т.д.;
- создать таблицу, содержащую столбцы по каждой независимой переменной (имена столбцов должны строго совпадать с именами переменных модели), и заполнить каждую строку значениями, для которых должно быть сделано предсказание. Таким образом, делается прогноз по каждой строке таблицы:

predict(модель, newdata = таблица).

ПРИМЕР

Пусть по модели Михаэлиса–Ментен был осуществлен опыт с двумя различными энзимами (A и B)

```
> модель <- nlsList(vitesse~SSmicmen(concentration,Vmax,
                                                Km)|enzyme)
```

Можно предсказать значение скорости реакции таким образом:

```
> predict(модель,newdata=list(enzyme="A",concentration=0.5))
```

Или для нескольких значений:

```
> predict(модель,newdata=list(enzyme=c("A","B"),
                              concentration=c(0.5,0.5)))
```

Или еще:

```
> predict(модель,newdata=list(enzyme=c("A","B"),
                              concentration=rep(0.5,2)))
```

Или создадим и используем таблицу :

```
>    таблица
enzyme concentration
1      A           0.5
2      B           0.5
>  predict(модель,newdata= таблица)
```

Графики

Чтобы проиллюстрировать связь между объясняемой переменной и независимой переменной ковариата, требуется три этапа:

1. Вывести на график точки наблюдения : plot(отклик ~ ковариата) .
2. Создать вектор, имеющий те же минимум и максимум, что ковариата, но с очень маленьким интервалом между его значениями : x <- seq2(ковариата)*.
3. Добавить кривую модельных значений на график.

Определение всех точек модельной кривой основано на прогнозе значений, которые принимает объясняемая переменная для каждого значения вектора x. Большое количество значений этого вектора и очень маленький интервал, который их отделяет, определяет гладкость кривой (которая на самом деле состоит из сотен точек, соединенных между собой). Если модель содержит другие независимые переменные, то к прогнозу необходимо добавить одинаковые значения, такие, чтобы величина отклика изменяла свою величину только при изменении анализируемой ковариаты. Кривая на график может быть добавлена функцией

```
lines(x,predict(модель,newdata= x)).
```

Примечание: функция `lines()` может быть использована несколько раз, чтобы начертить несколько кривых на одном графике, например, для нескольких уровней одного и того же фактора, включенного в модель. Чтобы изменить тип линий графика и цвет кривых, можно использовать аргументы `lty` и `col` соответственно. Наконец, чтобы добавить заголовки, используйте функцию `legend()`. Смотрите справку `?` для получения дополнительных сведений об используемых параметрах и знакомства с различными вариантами графики.

ПРИМЕРЫ

Рассмотрим модель, представленную выше:

```
>    модель <- nlsList(vitesse~SSmicmen(concentration,Vmax,
                                        Km)|enzyme)
```

Шаг 1 : нарисуем точки, соответствующие наблюдаемым данным :

```
> plot(vitesse ~ concentration)
```

Шаг 2 : создадим вектор x :

```
> x <- seq2(concentration)
```

Шаг 3 : добавляем модельную кривую.

Выбираем один из уровней A фактора, включенного в модель:

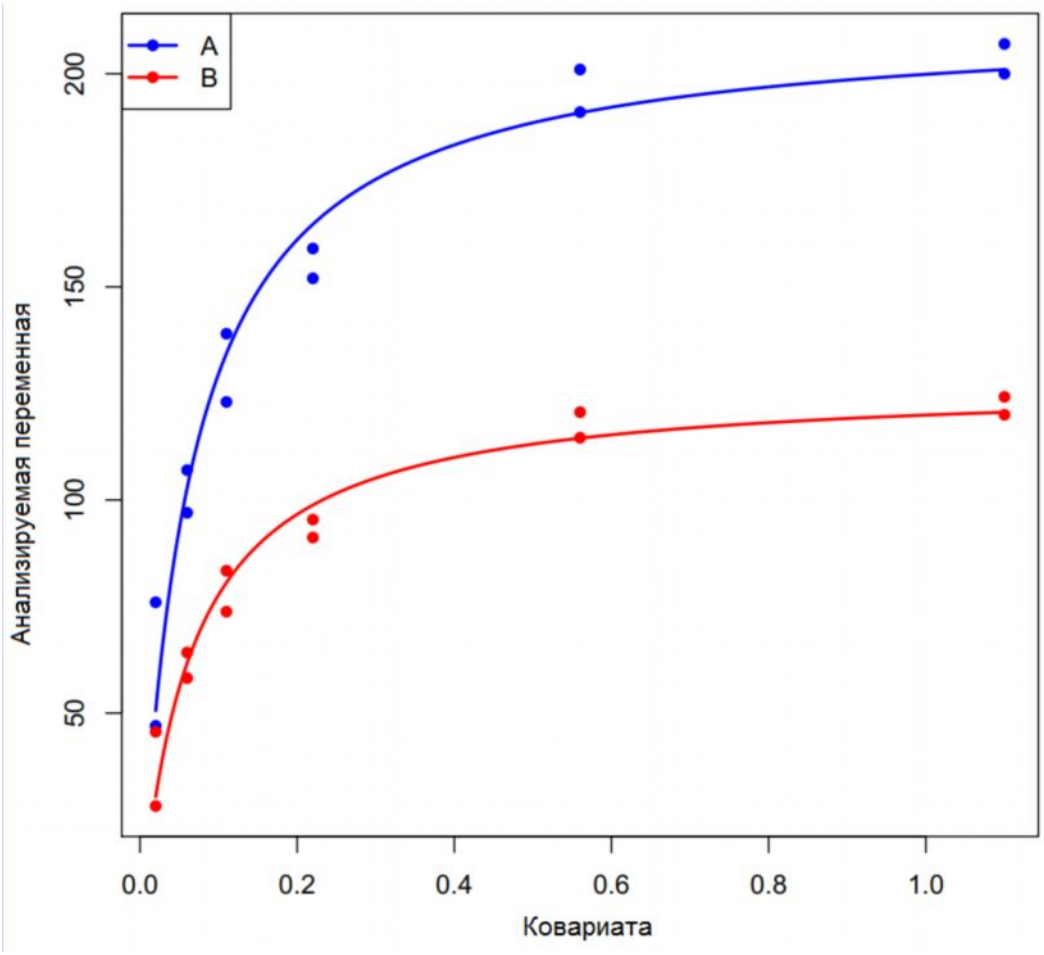
```
> lines(x,predict(модель,newdata=list(concentration = x,
     enzyme =rep ("A",length(x)))), col = 2)
```

Функция `rep()` является очень полезной в этой ситуации, поскольку она позволяет очень просто создать вектор той же длины, что и x, содержащий одно и тоже повторяющееся значение A.

Чтобы добавить модельную кривую, соответствующую уровню B фактора:

```
> lines(x,predict(модель,newdata=list(concentration = x,
          enzyme =rep ("B",length(x)))))
```

На полученном рисунке синим цветом представлена модельная кривая логистической регрессии для уровня A воздействующего фактора, а красным цветом – уровня B.


A
B
Анализируемая переменная
50
100
150
200
0.0
0.2
0.4
0.6
0.8
1.0
Ковариата

ОТКЛИК – ВРЕМЯ ДО ВОЗНИКНОВЕНИЯ СОБЫТИЯ

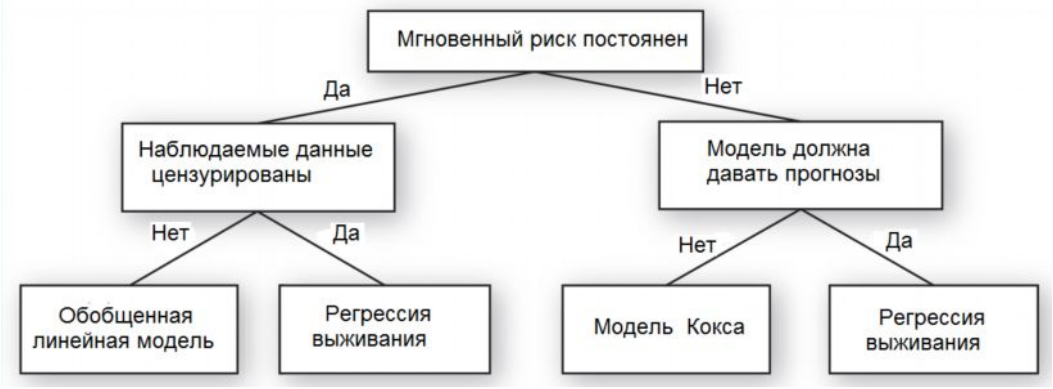
Время до возникновения события мы, для удобства, будем называть в последующих разделах "*время выживания*" (а само событие – "смерть"), поскольку модели, разработанные для этого типа отклика изначально использовались для анализа времени выживания. Но это – всего лишь вопрос лексики, а сам подход одинаков вне зависимости от природы события.

78. Выбор модели анализа времени выживания

Для анализа времени выживания необходимо предварительно определить следующие понятия:

- *Цензурирование*: данные об объектах считаются цензурированными, если начальная "точка отсчета" времени до смерти неизвестна (цензурирование *слева*), либо смерть не была зарегистрирована до конца периода наблюдения, потому что этот период уже завершился или наблюдение за объектом по каким-то причинам было потеряно (цензурирование *справа*). В этом справочнике рассматриваются только объекты, цензурированные справа. Существенное условие для анализа цензурированных данных: они должны быть *независимыми* от условий опыта.
- *Мгновенный риск*: это плотность вероятности гибели объекта в момент времени *t* при условии, что он до этого момента дожил. Риск может быть постоянным, или, напротив, увеличиваться или уменьшаться во времени.

Чтобы проанализировать время выживания, могут быть использованы несколько моделей, довольно различных по своему типу. Для выбора наиболее соответствующих из них:



Замечание: подгонка кривой выживания регрессионной моделью вполне может быть использовано для интерпретации данных, но также иметь своей целью выполнить экстраполяцию результатов (т.е. прогнозирование исхода). Модель Кокса, напротив, ограничена по своей прогнозирующей способности.

Чтобы оценить, постоянен ли мгновенный риск, можно сформировать график кривой выживания объектов, используя функцию:
`plotsurvivors(гибель, цензура)*` ,' где `гибель` – вектор, содержащий время до смерти каждого объекта, и `цензура` – вектор, значения которого указывают, цензурирован объект или нет (0 – цензурирован, т.е. смерть не была зарегистрирована, или 1 – не цензурирован), в том же порядке, что и `гибель`. Мгновенный риск считается постоянным, если точки кривой выживания выровнены почти по одной прямой.

79. Анализ времени выживания – обобщенная линейная модель

Обсуждаемые здесь процедуры анализа используют основные понятия построения моделей, изложенные в п. **39- 42**.

Отклик

Отклик, названный в этом разделе "временем выживания" (см. п. **78**), представлен только произвольными положительными числами.

Типы модели

Хотя основные принципы остаются одними и теми же, некоторые технические аспекты анализа отличаются в зависимости от того, связан ли отклик с другими факторами, определяющими некоторые важные для исследования условия эксперимента. Если связанный фактор отсутствует, то используется Обобщенная Линейная Модель (*Generalized Linear Model* или GLM), в то время, как при наличии связанных факторов – Смешанная Обобщенная Линейная Модель (*Generalized Linear Mixed Model* или GLMM). Термин "смешанный" подразумевает наличие, по крайней мере, одного случайного фактора, который используется, чтобы точно идентифицировать отдельные серии наблюдений (см. п. **11**).

В отличие от обыкновенной Линейной Модели (или ее смешанного варианта, см. далее п. **76**), GLM- или GLMM-модели не основаны на нормальном законе распределения. В случае времени выживания обычно используют Гамма-распределение.

Построение модели

Для создания модели используют следующие функции:

- без связанных случайных факторов

```
модель <- glm(формула,family="Gamma")
```

- со связанными качественными случайными переменными:

`модель <- glmer(формула,family="Gamma")` из пакета `lme4`

См. п. **40** для подробного объяснения построения формулы. В целом можно напомнить, что:

- включение независимого фактора позволяет проверить, имеется ли вариация отклика между уровнями этого фактора;
- включение ковариаты позволяет проверить, существует связь между этой ковариатой и объясняемой переменной;
- включение взаимодействия двух независимых переменных позволяет проверить, зависит ли один эффект от другого (это же рассуждение – для наблюдений, приуроченных к двум или более отрезкам времени). Особый случай – анализ взаимодействия между ковариатой и фактором, который позволяет проверить, зависит ли отношение между независимой переменной и откликом от уровней фактора.

Проверка адекватности модели

Прежде чем проводить дальнейший анализ, необходимо проверить, насколько хорошо модель подогнана к анализируемым данным. Этот этап фундаментален для любой модели, так как любой тест, проводимый на недостаточно адекватной модели, просто ненадежен (или не соответствует действительности). См. п. **41** для подробного объяснения методики этой проверки.

Способность модели к объяснению

Можно оценить общую объясняющую ценность модели на основе *коэффициента детерминации* (R^2), который представляет собой долю от общей вариации отклика, которая объяснена независимыми переменными модели. Этот

коэффициент может быть получен с использованием функции `r.squaredGLMM (модель)` из пакета `MuMIn`. Функция фактически возвращает два значения: *маргинальный* R^2 (`R2m`), который соответствует доле вариации, объясненной только ковариатами и фиксированными факторами, и *условный* R^2 (`R2c`), который соответствует доле вариации, объясненной всеми предикторами (т.е. как фиксированными, так и случайными факторами). В случае GLM оба значения идентичны, так как случайные факторы отсутствуют.

Тестирование

Для всех упомянутых моделей значимость объясняющих переменных оценивается функцией: `Anova (модель)` из пакета `car` (см. п. **42** для подробного объяснения процесса тестирования гипотез). Однако этот тест для разных моделей имеет характерные особенности:

- для GLM функция выполняет анализ отношения максимального правдоподобия (*Likelihood Ratio Test* или LR Test) и тестирует отдельно каждый член модели (т.е. по строкам возвращаемый таблицы с оценками коэффициентов);
- для GLMM функция выполняет тест Вальда и анализирует отдельно каждый член модели (т.е. по строкам возвращаемый таблицы с оценками коэффициентов).

Если один фактор (или взаимодействие с участием фактора) имеет статистически значимый эффект, то желательно осуществить множественное парное сравнение его уровней (или сочетаний этих уровней), чтобы выявить, в каких комбинациях имеются различия. Посмотрите п. **43** как выполнить эти сравнения.

Если ковариата имеет значимое влияние, направление эффекта определяется знаком параметра. Полную информацию обо всех параметрах модели можно получить с помощью команды `summary(модель)`. Значения самих коэффициентов называются `Estimate` и находятся в таблице `Coefficients` для GLM и `Fixed effects` для GLMM. Если знак коэффициента для количественной переменной отрицательный, то время выживания уменьшается, когда значение ковариаты увеличивается; а если он положительный, то время увеличивается, когда значение ковариаты увеличивается.

Прогноз на основе модели

Смысл моделей – не только показать роль различных независимых переменных в изменчивости объясняемой переменной, но и предсказать значение, которое бы принял отклик при известных значений независимых переменных. Поэтому, чтобы оценить время выживания, требуется зафиксировать значение всех переменных.

Для прогнозирования могут быть использованы два метода (для GLMM доступен только второй), и оба они основаны на функции `predict()`:

- значение каждой из независимых переменных дается непосредственно в теле функции в виде списка (с одной или несколькими записями): `predict(модель,newdata=list(переменные),type="response")`, где `переменные` – это последовательность `var1= значение,var2=значение` и т.д.;
- создать таблицу, содержащую столбцы по каждой независимой переменной (имена столбцов должны строго совпадать с именами переменных модели), и заполнить каждую строку значениями, для которых должно быть сделано предсказание. Таким образом, делается прогноз по каждой строке таблицы:
`predict(модель, newdata = таблица, type = "response")`.

Примечание: в случае GLMM некоторые значения случайных факторов могут не иметь данных. Если этого не произошло, то необходимо добавить в функцию `predict()` аргумент `re.form=NA`. Тогда при оценке прогноза будут приниматься во внимание все эффекты случайных факторов модели.

ПРИМЕРЫ

Имеем модель, содержащую фактор с двумя уровнями (A и B), ковариату ковариата в пределах от 0 до 30, и их взаимодействия:

```
> модель <- glm(отклик ~ фактор*ковариата,family="Gamma")
```

Можно предсказать время выживания таким образом :

```
> predict(модели,newdata= list (фактор="A", ковариата =10),
                                    type="response"),
```

Или для нескольких прогнозов:

```
> predict(модель,newdata=list(фактор =c("A","B"),
                    ковариата =c(10,10), type="response")
```

Или предварительно создать таблицу следующего типа:

```
> таблица
  Фактор ковариата
1      A        10
2      B        10
> predict(модель, newdata= таблица, type="response")
```

Графики

Влияние фактора. Чтобы проиллюстрировать результат воздействия фактора, обычно рисуются кривые выживания (по методике Каплан-Мейера). Их построение осуществляется в три этапа:

1. Создать *объект выживания*, который является особой формой представления времен выживания с использованием функции из пакета `survival`:

`выживание <- Surv(отклик)`, где `отклик` – данные о времени выживания.

2. Создать объект, содержащий данные для графика

`кривые <- survfit(выживание~фактор)`

3. Начертить график: `plot(кривые)`. Можно добавить на график доверительный интервал (например, 95 %) для каждой кривой, указав аргумент `conf.int=TRUE` (остерегайтесь увеличивать число кривых, что приведет к трудно читаемому графику). Чтобы изменять тип линии и цвет кривых, используйте аргументы `lty` и `col` (для информации об этих и других графических параметров наберите `?par`). Наконец, можно добавить легенду, используя функцию `legend()` (используйте `? legend` для большей информации).

Зависимость от ковариаты. Чтобы проиллюстрировать связь между временем выживания и независимой ковариатой, требуется три этапа:

1. Нанести на график точки наблюдения: `plot(отклик ~ ковариата)`.

2. Создать вектор, имеющий те же минимум и максимум, что `ковариата`, но с очень маленьким интервалом между его значениями : `x <- seq2(ковариата)*`.

3. Добавить кривую модельных значений на график.

Определение всех точек модельной кривой основано на прогнозе значений, которые принимает объясняемая переменная для каждого значения вектора `x`. Большое количество значений этого вектора и очень маленький интервал, который их отделяет, определяет гладкость кривой (которая на самом деле состоит из сотен точек, соединенных между собой). Если модель содержит другие независимые переменные, то к прогнозу необходимо добавить одинаковые значения, такие, чтобы величина отклика изменяла свою величину только при изменении анализируемой ковариаты. Кривая на график может быть добавлена функцией

`lines(x,predict(модель,newdata= x, type="response"))`.

Примечание: функция `lines()` может быть использована несколько раз, чтобы начертить несколько кривых на одном графике, например, для нескольких уровней

одного и того же фактора, включенного в модель. Чтобы изменить тип линий графика и цвет кривых, можно использовать аргументы `lty` и `col` соответственно. Наконец, чтобы добавить перечень обозначений, используйте функцию `legend()`. Смотрите справку `?` для получения дополнительных сведений об используемых параметрах и знакомства с различными вариантами графики.

ПРИМЕРЫ

Рассмотрим следующую модель:

```
> модель <- glm(отклик ~ фактор*ковариата,family="Gamma")
```

Шаг 1 : нарисуем точки, соответствующие наблюдаемым данным :

```
> plot(отклик ~ ковариата)
```

Шаг 2 : создадим вектор `x` :

```
> x <- seq2(ковариата)*
```

Шаг 3 : добавляем модельную кривую.

Выбираем один из уровней фактора, включенного в модель:

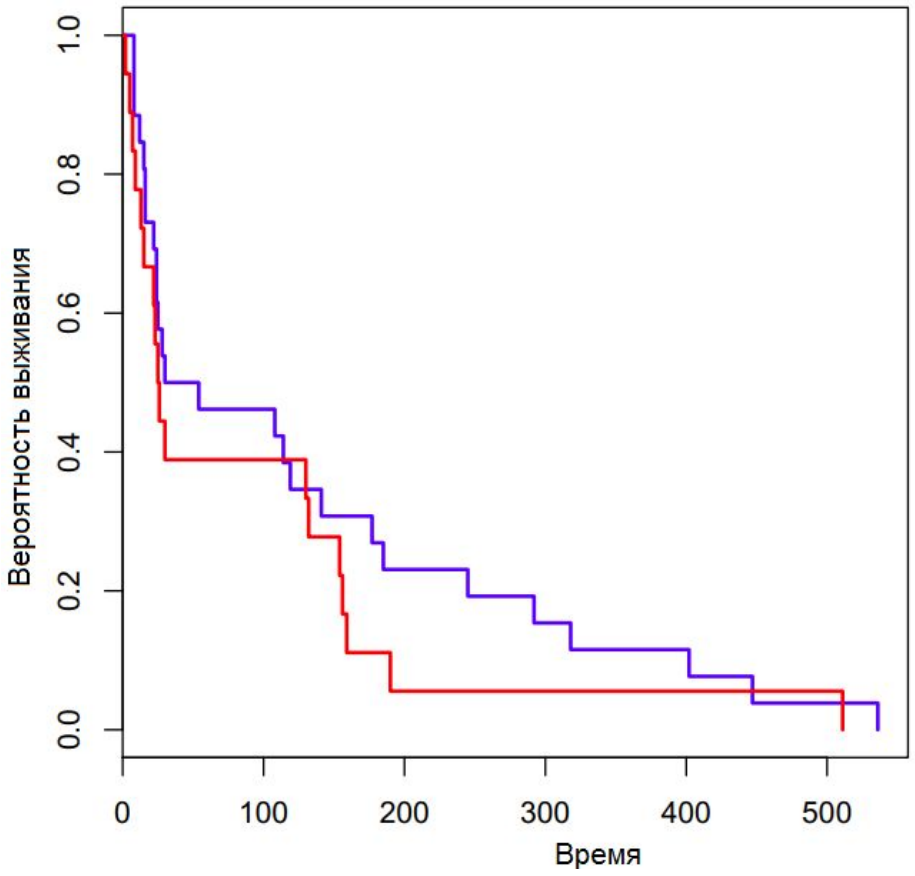
```
> lines(x,predict(модель,newdata=list(ковариата = x,
      фактор =rep("A",length(x))), type="response"), col = 2)
```

Функция `rep()` является очень полезной в этой ситуации, поскольку она позволяет очень просто создать вектор той же длины, что и `x`, содержащий одно и тоже повторяющееся значение `A`.

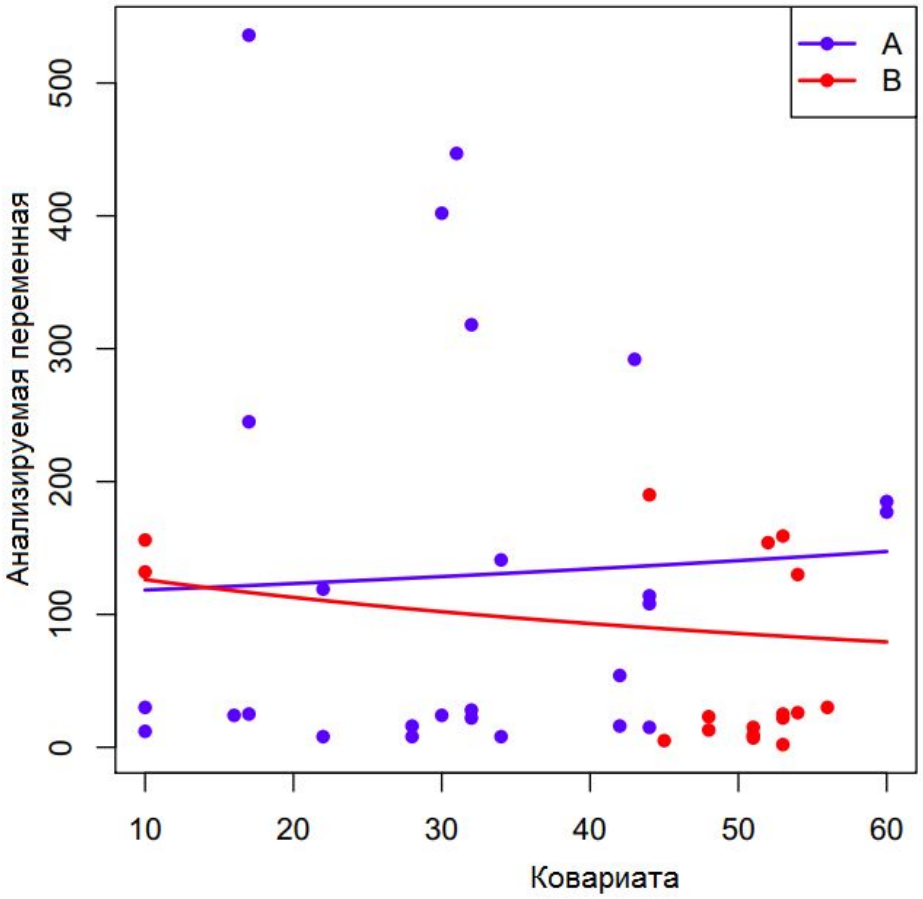
Чтобы добавить модельную кривую, соответствующую уровню `B` фактора :

```
> lines(x,predict(модель,newdata=list(ковариата = x,
           фактор =rep("B",length(x))), type="response"))
```

На полученном рисунке представлен график оценки вероятности выживания для двух наблюдаемых объектов:



На другом полученном рисунке синим цветом представлена линия регрессии выживания для уровня A воздействующего фактора, а красным цветом – для уровня B.



80. Анализ времени выживания – регрессионная модель

Обсуждаемые здесь процедуры анализа используют основные понятия построения моделей, изложенные в п. **39- 42**.

Отклик

В регрессионных моделях выживания объясняемая переменная величина может не всегда являться непосредственно временем до гибели объекта. Поэтому для использования в функциях пакета `survival`, которые рассматриваются ниже, формируется специальный объект, называемый *объектом выживания*. Он одновременно учитывает время выживания и возможную ссылку на цензурированность (т.е. выделяются объекты, смерть которых не была зарегистрирована до конца наблюдения; см. п. **78**). Чтобы создать объект `выживание`, используют функцию:

- `выживание <- Surv(гибель)`, если нет ни одного цензурированного объекта, а `гибель` – вектор, содержащий время до смерти каждого из них.
- `выживание <- Surv((гибель, цензура)`, где `гибель` – вектор, содержащий время до смерти каждого объекта, и `цензура` – вектор, значения которого указывают, цензурирован объект или нет (0 – цензурирован, т.е. смерть не

была зарегистрирована, или 1 – не цензурирован), в том же порядке, что и `гибель`.

В формуле регрессионной модели объясняемая переменная указывается как `выживание`, т.е. делается ссылка на объект выживания.

Построение модели

Для создания модели используют следующие функции:

- если мгновенный риск постоянный, то используется экспоненциальный закон распределения:

```
модель<- survreg (формула, dist = "Exponential");
```

- если мгновенный риск не постоянен, то распределение Вейбулла служит, чтобы смоделировать дрейф риска в течение времени (то увеличение, то уменьшение):

```
модель<- survreg (формула, dist = "Weibull").
```

См. п. **40** для подробного объяснения построения формулы. В целом можно также напомнить, что:

- включение независимого фактора позволяет проверить, имеется ли вариация отклика между уровнями этого фактора;
- включение ковариаты позволяет проверить, существует связь между этой ковариатой и объясняемой переменной;
- включение взаимодействия двух независимых переменных позволяет проверить, зависит ли один эффект от другого (это же рассуждение – для наблюдений, приуроченных к двум или более отрезкам времени). Особый случай – анализ взаимодействия между ковариатой и фактором, который позволяет проверить, зависит ли отношение между независимой переменной и откликом от уровней фактора.

В случае модели с переменным риском, изменение этого риска показано командой `summary(модель)`. Если параметр `Scale < 1`, риск сокращается со временем, если `Scale > 1`, риск увеличивается.

Проверка адекватности модели

Прежде чем проводить дальнейший анализ, необходимо проверить, насколько хорошо модель подогнана к анализируемым данным. Этот этап фундаментален для любой модели, так как любой тест, проводимый на недостаточно адекватной модели, просто ненадежен (или не соответствует действительности). См. п. **41** для подробного объяснения методики этой проверки.

Тестирование

Для упомянутой модели значимость объясняющих переменных оценивается функцией: `Anova(модель)` из пакета `car` (см. п. **42** для подробного объяснения процесса тестирования гипотез). Функция выполняет анализ отношения максимального правдоподобия (*Likelihood Ratio Test* или LR Test) и тестирует отдельно каждый член модели (т. е. по строкам возвращаемый таблицы с оценками коэффициентов).

Если один фактор (или взаимодействие с участием фактора) имеет статистически значимый эффект, необходимо осуществить множественное парное сравнение его уровней (или сочетаний этих уровней), чтобы выявить, в каких комбинациях имеются различия. Посмотрите п. **43** как выполнить эти сравнения.

Если ковариата имеет значимое влияние, направление эффекта определяется знаком параметра. Полную информацию о всех параметрах модели можно получить с помощью команды `summary(модель)`. Значения самих коэффициентов называются `Value`. Если знак коэффициента для количественной переменной отрицательный, то время выживания уменьшается, когда значение ковриаты увеличивается; а если он положительный, то время увеличивается, когда значение ковариаты увеличивается.

Прогноз на основе модели

Смысл моделей – не только показать роль различных независимых переменных в изменчивости объясняемой переменной, но и предсказать значение, которое бы принял отклик при известных значениях независимых переменных. Поэтому, чтобы оценить время выживания, требуется зафиксировать значение всех переменных.

Для прогнозирования могут быть использованы два, и оба они основаны на функции `predict()`:

- значение каждой из независимых переменных дается непосредственно в теле функции, в виде списка (с одной или несколькими записями): `predict(модель,newdata=list(переменные))`, где `переменные` – это последовательность `var1= значение,var2=значение` и т.д.;
- создать таблицу, содержащую столбцы по каждой независимой переменной (имена столбцов должны строго совпадать с именами переменных модели), и заполнить каждую строку значениями, для которых должно быть сделано предсказание. Таким образом, делается прогноз для каждой строки таблицы:

`predict(модель, newdata = таблица)`.

ПРИМЕРЫ

Имеем модель, содержащую `фактор` с двумя уровнями (`A` и `B`), ковариату `ковариата` в пределах от 0 до 30, и их взаимодействия :

```
> модель <- survreg (выживание ~ фактор*ковариата,
          dist = "Exponential")
```

Можно предсказать время выживания таким образом :

```
> predict(модели,newdata= list (фактор="A", ковариата =10)
```

Или для нескольких прогнозов :

```
> predict(модель,newdata=list(фактор =c("A","B"),
                    ковариата =c(10,10))
```

Или предварительно создать таблицу следующего типа:

```
> таблица
  Фактор ковариата
1      A        10
2      B        10
> predict(модель, newdata= таблица, type="response")
```

Графики

Влияние фактора. Чтобы проиллюстрировать результат воздействия фактора, обычно рисуются кривые выживания (по методике Каплан-Мейера). Их построение осуществляется в два этапа:

1. Создаем объект, содержащий данные для графика

`кривые <-survfit (выживание~фактор)`

2. Начертим график: `plot(кривые)`.

Если объекты подвергались цензурированию, они отмечаются крестом `"+"`. Можно добавить на график доверительный интервал (например, 95 %) для каждой кривой, указав аргумент `conf.int=TRUE` (остерегайтесь увеличивать число кривых, что приведет к трудно читаемому графику). Чтобы изменять тип линии и цвет кривых, используйте аргументы `lty` и `col` (для информации об этих и других графических параметров наберите `?par`). Наконец, можно добавить легенду, используя функцию `legend ()` (используйте `? legend` для подробной информации).

Зависимость от ковариаты. Чтобы проиллюстрировать связь между временем выживания и независимой ковариатой, требуется три этапа:

1. Вывести на график точки наблюдения: `plot(гибель ~ ковариата)`.

2. Создать вектор, имеющий те же минимум и максимум, что `ковариата`, но с очень маленьким интервалом между его значениями : `x <- seq2(ковариата)*`.

3. Добавить кривую модельных значений на график.

Определение всех точек модельной кривой основано на прогнозе значений, которые принимает объясняемая переменная для каждого значения вектора `x`. Большое количество значений этого вектора и очень маленький интервал, который их отделяет, определяет гладкость кривой (которая на самом деле состоит из сотен точек, соединенных между собой). Если модель содержит другие независимые переменные, то к прогнозу необходимо добавить одинаковые значения, такие, чтобы величина отклика изменяла свою величину только при изменении анализируемой ковариаты.

Кривая на график может быть добавлена функцией `lines(x,predict(модель,newdata= x, type="response"))`.

Примечание: функция `lines()` может быть использована несколько раз, чтобы начертить несколько кривых на одном графике, например, для нескольких уровней одного и того же фактора, включенного в модель. Чтобы изменить тип линий графика и цвет кривых, можно использовать аргументы `lty` и `col` соответственно. Наконец, чтобы добавить заголовки, используйте функцию `legend()`. Смотрите справку `?` для получения дополнительных сведений об используемых параметрах и знакомства с различными вариантами графики.

ПРИМЕРЫ

Рассмотрим следующую модель :

```
> модель <- survreg(выживание ~ фактор*ковариата,
          dist = "Exponential")
```

Шаг 1 : нарисуем точки, соответствующие наблюдаемым данным :

```
> plot(гибель ~ ковариата)
```

Шаг 2 : создадим вектор `x` :

```
> x <- seq2(ковариата)
```

Шаг 3 : добавляем модельную кривую. Мы выбираем один из уровней фактора, включенного в модель:

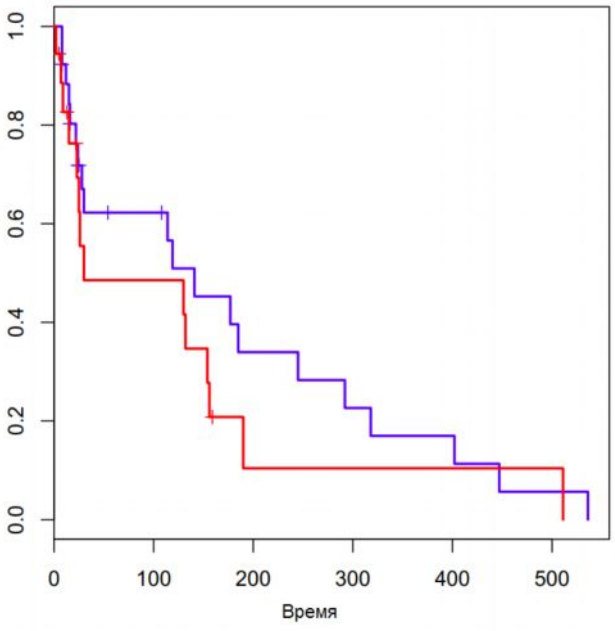
```
> lines(x,predict(модель,newdata=list(ковариата = x,
      фактор =rep (A,length(x))), type="response"), col = 2)
```

Функция `rep()` является очень полезной в этой ситуации, поскольку она позволяет очень просто создать вектор той же длины, что и `x`, содержащий одно и то же повторяющееся значение `A`.

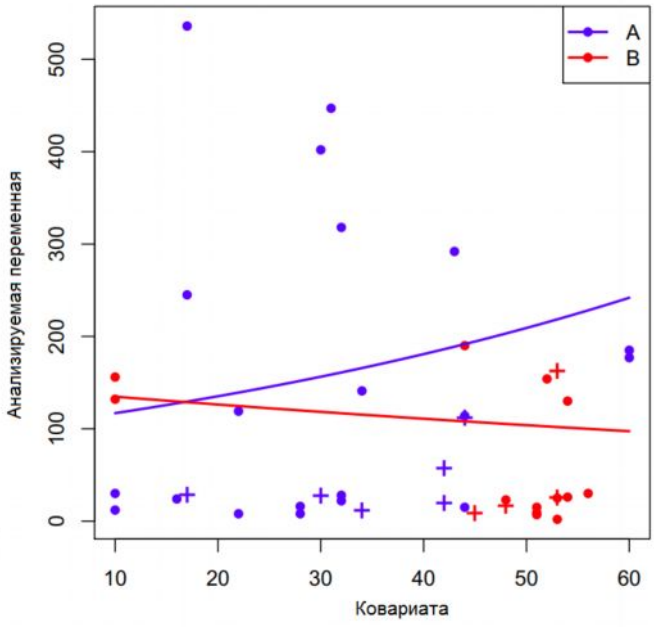
Чтобы добавить модельную кривую, соответствующую уровню `B` фактора :

```
> lines(x,predict(модель,newdata=list(ковариата = x,
           фактор =rep (B,length(x))), type="response"))
```

График вероятности выживания для двух объектов



Синим цветом представлена линия регрессии выживания для уровня A воздействующего фактора, а красным цветом – уровня B.



81. Анализ времени выживания – модель Кокса

Обсуждаемые здесь процедуры анализа используют основные понятия построения моделей, изложенные в п. **39- 42**.

Отклик

В модели выживания Кокса объясняемая переменная величина не всегда может являться непосредственно временем до гибели объекта. Поэтому для использования в функциях пакета `survival` формируется специальный объект, называемый *объектом выживания*. Он одновременно учитывает время выживания и возможную ссылку на цензурированность (т.е. выделяются объекты, смерть которых не была зарегистрирована до конца наблюдения; см. п. **78**). Чтобы его создать, используют функцию:

- `выживание <- Surv(гибель)`, если нет ни одного цензурированного объекта, а `гибель` – вектор, содержащий время до смерти каждого из них.
- `выживание <- Surv((гибель, цензура)`, где `гибель` – вектор, содержащий время до смерти каждого объекта, и `цензура` – вектор, значения которого указывают, цензурирован объект или нет (0 – цензурирован, т.е. смерть не была зарегистрирована, или 1 – не цензурирован), в том же порядке, что и `гибель`.

В формуле регрессионной модели объясняемая переменная указывается как `выживание`, т.е. делается ссылка на объект выживания.

Построение модели

Для создания модели Кокса используют функцию:

```
Модель <- coxph(формула).
```

См. п. **40** для подробного объяснения построения формулы. В целом можно также напомнить, что :

- включение независимого фактора позволяет проверить, имеется ли вариация отклика между уровнями этого фактора;
- включение ковариаты позволяет проверить, существует связь между этой ковариатой и объясняемой переменной;
- включение взаимодействия двух независимых переменных позволяет проверить, зависит ли один эффект от другого (это же рассуждение – для наблюдений, приуроченных к двум или более отрезкам времени). Особый случай – анализ взаимодействия между ковариатой и фактором, который позволяет проверить, зависит ли отношение между независимой переменной и откликом от уровней фактора.

Определенная специфика модели Кокса состоит в том, что в отличие от других моделей анализа времени выживания, она непараметрическая (по другим точкам зрения – полу-параметрическая), и по этой причине ее нельзя использовать в целях прогнозирования.

Проверка адекватности модели

Прежде чем проводить дальнейший анализ, необходимо проверить, насколько хорошо модель подогнана к анализируемым данным. Этот этап фундаментален для любой модели, так как любой тест, проводимый на недостаточно адекватной модели, просто ненадежен (или не соответствует истине). Однако у модели Кокса есть особенные условия адекватности.

Во-первых, зависимость между каждой количественной объясняющей переменной (ковариатой) и мгновенным риском должна быть лог-линейной. Чтобы проверить эту гипотезу, можно использовать функцию: `cox.resid (модель)*`. Функция рисует график для каждой ковариаты, на котором красной линией отображается тенденция распределения облака точек. Гипотеза лог-линейности для ковариаты принимается, если соответствующая красная линия не отклоняется от горизонтальной прямой. В противном случае наиболее простое решение состоит в том, чтобы разделить ковариату на диапазоны, а затем использовать ее в модели как фактор.

ПРИМЕР.

Для модели, включающей фактор и ковариату

```
> модель <- coxph(выживание~фактор+ковариата)
```

необходимо проверить гипотезу лог-линейности между откликом и ковариатой:

```
> cox.resid(модель)
```

Если это предположение не соблюдено, разделяем ковариату на классы.

Пример с разбиением на два класса:

```
> covar.class <- cut(ковариата, breaks=2)
```

Замечание: чтобы добавить имена к уровням создаваемого фактора, используйте аргумент `label`. У обоих классов по умолчанию – одна и та же численность объектов. Поскольку для функции доступно много опций, для более полной информации посмотрите `? cut`.

Новая модель создана с ковариатой, превращенной в фактор:

```
> модель <-coxph (выживание~фактор + covar.class)
```

Во-вторых, отношение мгновенных рисков для любых объектов должно быть постоянным в течение времени наблюдений («гипотеза пропорциональных рисков»). Чтобы проверить эту гипотезу, используйте команду: `cox.zph(модель)`. Функция тестирует гипотезу с учетом каждой объясняющей переменной, а также для глобальной модели. Если *p*-значение является значимым, это указывает на то, что предположение не соблюдается и риски при использовании данной переменной зависят от времени. Тогда лучше включить ее в модель Кокса не как объясняющую переменную, а как страту (*strate*). Для количественных переменных это также можно осуществить путем разрезания на классы и преобразования их в фактор. Эффект для этой переменной больше не рассчитывается при подгонке модели, но будет учитываться путем определения различных значений мгновенных рисков в зависимости от страт. Чтобы включить страту в формулу модели, добавьте `+ strata(переменная)` после списка объясняющих переменных, из которых элиминированная ковариата должна быть удалена.

ПРИМЕР. Для той же модели, включающей фактор и ковариату

```
> модель <-coxph(выживание~фактор+ковариата)
```

необходимо проверить гипотезу пропорционального риска:

```
cox.zph(модель)
```

Если это предположение не соблюдается для переменной `фактор`, то формируем новую модель со стратификацией:

```
> модель <-coxph(выживание~ ковариата + strata(фактор))
```

Если гипотеза не соблюдается для ковариаты, то она должна быть факторизирована, а затем интегрирована как страта в модель:

```
> модель <-coxph(выживание~фактор + strata(covar.class))
```

Тестирование

В зависимости от того, содержит ли модель страту или нет, необходимо выполнить различные тесты:

- для модели без страты значимость объясняющих переменных оценивается функцией: `Anova(модель)`, которая выполняет анализ отношения максимального правдоподобия (*Likelihood Ratio Test* или LR Test) и тестирует отдельно каждый член модели (т. е. по строкам возвращаемой таблицы с оценками коэффициентов);
- для модели со стратой: `Anova(модель, test="Wald")`, где функция выполняет тест Вальда для каждого члена модели (т. е. по одному в каждой строке возвращаемой таблицы).

Если один фактор (или взаимодействие с участием фактора) имеет статистически значимый эффект, необходимо осуществить множественное парное сравнение его уровней (или сочетаний этих уровней), чтобы выявить, в каких комбинациях имеются различия. Посмотрите п. **43** для разъяснения, как выполнить эти сравнения.

Если ковариата имеет значимое влияние, направление эффекта определяется знаком параметра. Полную информацию обо всех параметрах модели можно получить с помощью команды `summary(модель)`. Значения самих коэффициентов называются `Value`. Если знак коэффициента для количественной переменной отрицательный, то время выживания уменьшается, когда значение ковриаты увеличивается; а если он положительный, то время увеличивается, когда значение ковариаты увеличивается.

Графики

Влияние фактора. Поскольку модель Кокса не является предикативной, на практике оценивается только влияние факторов на время выживания. Для этого обычно рисуются кривые выживания (по методике Каплан-Мейера). Их построение осуществляется в два этапа:

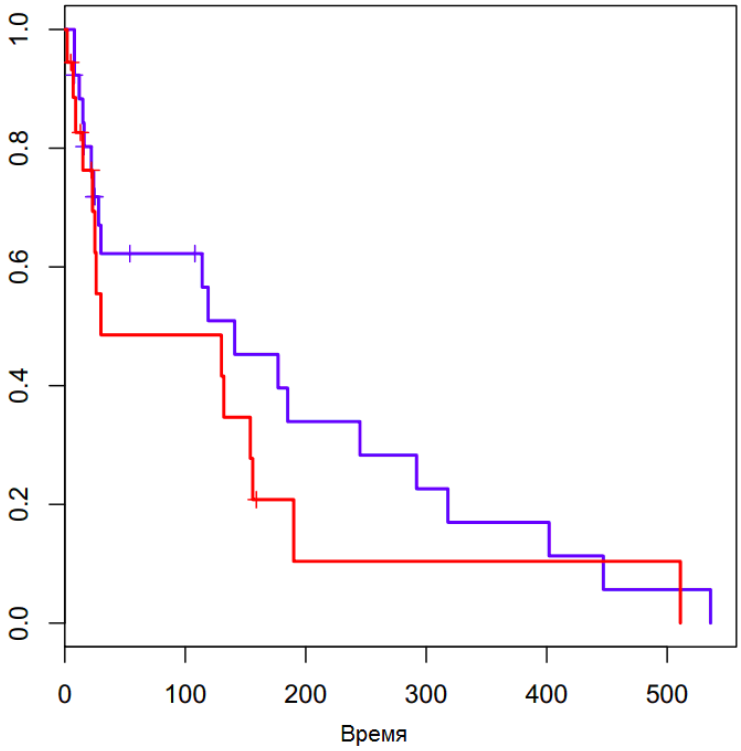
1. Создаем объект, содержащий данные для графика

```
кривые <-survfit (выживание~фактор).
```

Можно добавить в формулу страту в виде `+ strata(фактор)` для построения кривых для каждого уровня фактора и страты. Однако будьте осторожны, чтобы не умножать кривые, которые сделают ваш график трудно читаемым.

3. Начертим график: `plot(кривые)`. Можно добавить на график доверительный интервал (например, 95 %) для каждой кривой, указав аргумент `conf.int=TRUE` (остерегайтесь увеличивать число кривых, что приведет к трудно читаемому графику). Чтобы изменять тип линии и цвет кривых, используйте аргументы `lty` и `col` (для информации об этих и других графических параметров наберите `?par`). Наконец, можно добавить легенду, используя функцию `legend()` (используйте `? legend` для подробной информации).

На полученном рисунке представлен график выживания для двух объектов:



ДВУМЕРНЫЙ СТАТИСТИЧЕСКИЙ АНАЛИЗ

82. График "облака" точек

Этот тип графика позволяет представить значения каждого наблюдения двух связанных случайных переменных X и Y в виде $Y = f(X)$. Это позволяет получить общий обзор зависимости, которая может существовать между этими величинами. Построение графика основано на функции `plot()`, а исходными данными могут быть два вектора одинаковой длины, матрица или таблица из двух столбцов.

Если используются два вектора `x` и `y`, содержащие значение каждого наблюдения для этих переменных в одном и том же порядке, то подается команда: `plot(y~x)`. Символ ~ означает "*в зависимости от*".

Чтобы добавить название графика, используйте аргумент `main = "Название"`.

Если необходимо изменить наименование горизонтальной оси, задайте аргумент `xlab = "Легенда"`, а для определения вертикальной оси – `ylab = "Легенда"`. Чтобы познакомиться с большим числом других графических возможностей, наберите `? par`.

Чтобы добавить прямую линию типа $y = ax + b$: `abline(b, a)`.

Чтобы добавить прямую линию линейной регрессии по методу наименьших квадратов (см. п. **76**): `abline(lm(y~x))`.

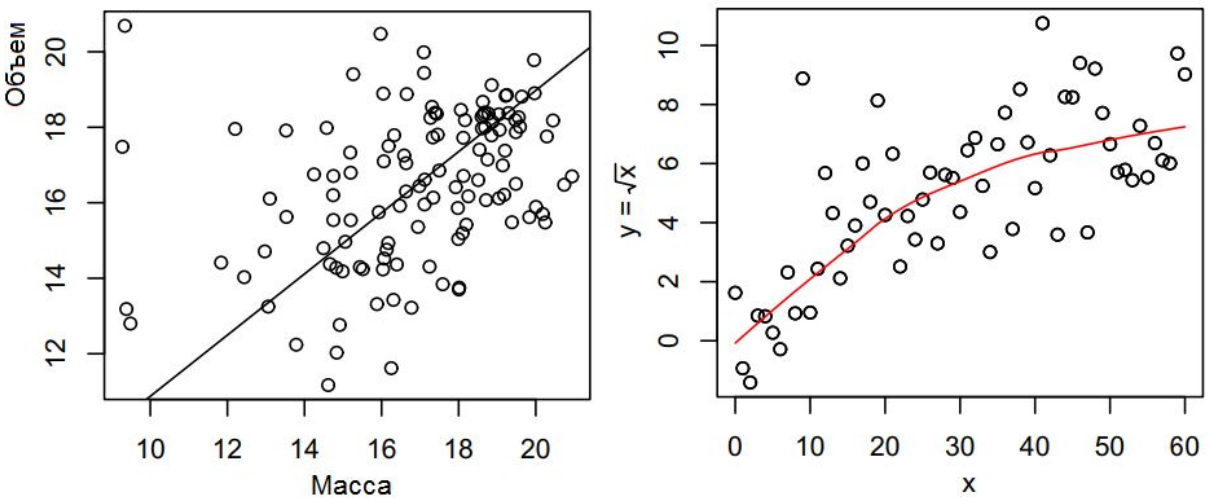
Чтобы добавить прямую линию линейной регрессии по методу наименьших прямоугольников (см. п. **87**):

`abline(least.rect(y~x))*`.

Для добавления сглаженной кривой, отображающей тенденцию изменения расположения точек, используйте функцию: `panel.smooth(x, y)`.

Добавление горизонтальной прямой линии: `abline(H=ордината)`.

Добавление вертикальной прямой линии: `abline(V=абсцисса)`.



83. Выраженность связи между двумя переменными

Описываемые ниже выборочные параметры позволяют оценить общую величину степени (тесноты) взаимосвязи двух наборов данных, которая (возможно) позволяет им изменяться синхронно. Выбор параметра зависит от типа обеих переменных, а иногда и от формы их зависимости. Последняя оценивается с использованием типичного графика "облака" точек (см. п. **82**) и может быть линейной, монотонной (т.е. только возрастающей или убывающей), либо какой-то иной.

Две количественные переменные величины

Линейную зависимость оценивают два важнейших параметра:

- ковариация Пирсона (параметрический), рассчитываемая функцией `cov(выборка1, выборка2)`, где `выборка1` и `выборка2` – векторы, содержащие значения каждого наблюдения для каждой переменной величины в одном и том же порядке;
- коэффициент корреляции Пирсона (параметрический), который находится как `cor(выборка1, выборка2)`.

Особый случай корреляции – частная корреляция, т.е. степень взаимосвязи двух переменных относительно друг друга, без учета влияния третьей переменной. Этот коэффициент рассчитывается с использованием функции `pcor(выборка1, выборка2, выборка3)*`, где `выборка3` – вектор, таблица или список, задающий переменные величины, корреляция с которыми должна быть исключена из оценки корреляции между `выборка1` и `выборка2`.

Во всех случаях коэффициент корреляции изменяется между -1 (полная обратная зависимость) и 1 (полная прямая зависимость), проходя через 0 (отсутствие зависимости).

Монотонную (*однообразную*) *зависимость* обнаруживают функции оценки непараметрических эквивалентов предыдущих параметров:

- ковариация Спирмена (непараметрический):
  `cov(выборка1, выборка2, method = " spearman ")`
- коэффициент корреляции Спирмена (непараметрический):
  `cor(выборка1, выборка2, method = " spearman ");`
- коэффициент частной корреляции – функция из пакета `RVAideMemoire`:
  `pcor(выборка1, выборка2, выборка3, method = "spearman")*.`

Какую-то иную зависимость никакой простой параметр не оценивает. Необходимо разделить диапазон варьирования переменных величин на классы, а затем их считать качественными. Смотрите для этого `? cut`.

Две порядковые качественные переменные величины

Монотонную (*однообразную*) *зависимость* могут оценивать непараметрические версии ковариации и коэффициента корреляции (например, Спирмена).

Две качественные переменные величины

Если каждый класс обеих переменных величин содержит по крайней мере 5 % от общей суммы количества объектов, можно рассчитать значение коэффициента ассоциации Крамера (непараметрический): `cramer (выборка1, выборка2)`.

Коэффициент меняется:

- между 0 (никакой ассоциации или *независимость*) и 1 (полная ассоциация), если хотя бы одна из переменных величин разделена более, чем на два класса.
- между -1 и 1, если каждая из обеих переменных разделена строго на два класса.

Чем больше коэффициент Крамера отдален от 0 (независимо от знака), тем больше ассоциация между переменными.

84. Корреляция между двумя количественными или порядковыми переменными величинами

Залача здесь состоит в том, чтобы определить доверительный интервал коэффициента корреляции и проверить его соответствие задаваемому теоретическому значению. См. п. **83** для объяснения, когда используется коэффициент Пирсона (параметрический) или коэффициент Спирмена (непараметрический).

Для всех функций, рассчитывающих доверительный интервал, ширина этого интервала может быть изменена благодаря аргументу `conf.level` (по умолчанию `conf.level=0.95`, то есть оценивается 95 %-й доверительный интервал).

Коэффициент корреляции Пирсона (параметрический)

Доверительный интервал и соответствие оценки параметра теоретическому значению, равному 0. Тест реализуется командой:
`cor.test(выборка1, выборка2)`, где векторы `выборка1` и `выборка2` содержат значения каждого наблюдения для каждой переменной (в одном и том же порядке). Функция возвращает в следующей последовательности: тест соответствия значению 0 (при справедливости теста корреляция считается статистически незначимой), вывод, сделанный относительно проверяемой гипотезы, доверительный интервал и значение коэффициента корреляции.

Для частной корреляции (т.е. когда элиминирует зависимость данных от одной или нескольких посторонних переменных, прежде чем оценивать и тестировать корреляцию) используют: `pcor.test(выборка1, выборка2, выборка3)*`, где `выборка3` – вектор, таблица или список, определяющий сопутствующие переменные (в том же порядке, что и `выборка1` и `выборка2`).

Тест соответствия теоретическому значению, отличному от 0. Чтобы выполнить тест, используется функция:
`cor.conf (выборка1, выборка2, theo=значение)*` , где `значение` – теоретический коэффициент корреляции.

Коэффициент корреляции Спирмена (непараметрический)

Доверительный интервал. Рассчитывается с использованием бутстреп–метода функцией: `spearman.ci(выборка1, выборка2)*`.

Тест соответствия теоретическому значению, равному 0. Чтобы осуществлять тест: `cor.test(выборка1, выборка2, method = "spearman")`. Функция возвращает величину коэффициента корреляции и *p*-значение теста на соответствие его нулю.
Для частной корреляции:
`pcor.test (выборка1, выборка2, выборка3, method = "spearman")*`.

Тест соответствия теоретическому значению, отличному от 0. Не существует теста, оценивающего значимость коэффициента корреляции Спирмена. Между тем достаточно рассчитать доверительный интервал коэффициента и увидеть, является ли теоретический коэффициент его частью. По определению, если 95 %-й доверительный интервал не включает теоретический коэффициент, то наблюдаемый коэффициент корреляции значимо отличается от теоретического коэффициента с риском ошибиться $\alpha = 5$ % (см. п. **15**).

85. Сравнение некоторых коэффициентов корреляции

См. п. **83** для объяснения, когда используется коэффициент корреляции Пирсона (параметрический) или коэффициент Спирмена (непараметрический).

Коэффициент корреляции Пирсона (параметрический)

Сравнение двух коэффициентов. Чтобы выполнить тест, можно по ситуации использовать одну из двух следующих функций:

- `cor.2comp(выборка1, выборка2, выборка3, выборка4)*` , где `выборка1` и `выборка2` – векторы, содержащие значение каждого наблюдения для обеих переменных, определяющих первую корреляцию (в одном и том же порядке), а `выборка3` и `выборка3` – векторы, содержащие аналогичные данные для вычисления второго коэффициента корреляции.
- `cor.multcomp(выборка1, выборка2, фактор)*` , где `выборка1` и `выборка2` – объединенные векторы обеих коррелируемых переменных, а `фактор` включает метки принадлежности к каждому из двух (или более) сравниваемых между собой комплексов наблюдений в составе векторов `выборка1` и `выборка2`.

Если оба коэффициента корреляции значимо не отличаются, обе функции возвращают значение общего коэффициента корреляции, его 95 %-й доверительный интервал и результат теста соответствия этого коэффициента нулевому значению (это теоретическое значение может быть изменено с использованием аргумента `theo=значение`, где `значение` – теоретический коэффициент корреляции).

Сравнение более чем двух коэффициентов корреляции. Чтобы выполнить тест, используйте функцию: `cor.multcomp(выборка1, выборка2, фактор)*`.

Если *p*-значение теста статистически значимо, это указывает, что по крайней мере два коэффициента корреляции попарно различаются между собой (не уточняется, какие именно). Функция тогда осуществляет все возможные парные сравнения.

Может случиться, что парные сравнения не указывают никаких значимых различий, вопреки общему тесту. В этом случае, наиболее осторожное решение состоит в том, чтобы считать, что мы не знаем, какие коэффициенты корреляции ответственны за отклонение от нулевой гипотезы в общем тесте.

Коэффициент корреляции Спирмена (непараметрический)

Не существует теста, оценивающего различие коэффициентов корреляции Спирмена. Между тем достаточно сравнить доверительные интервалы обоих коэффициентов. Действительно, если два 95%-х доверительных интервала не перекрывают друг друга, то оба коэффициента корреляции значимо отличаются на уровне доверия α = 5 % (см. п. **15**). Так как осуществляются множественные сравнения, необходимо скорректировать доверительные интервалы, как это было сделано для одного *p*-значения (см. п. **16**).

Чтобы рассчитать исправленные доверительные интервалы, используйте функцию: `spearman.cor.multcomp(выборка1, выборка2, фактор)*`. Функция возвращает таблицу из трех столбцов: коэффициенты корреляции (`r`) и нижние (`inf`) и верхние (`sup`) пределы доверительных интервалов для каждого уровня фактора.

ПРИМЕР

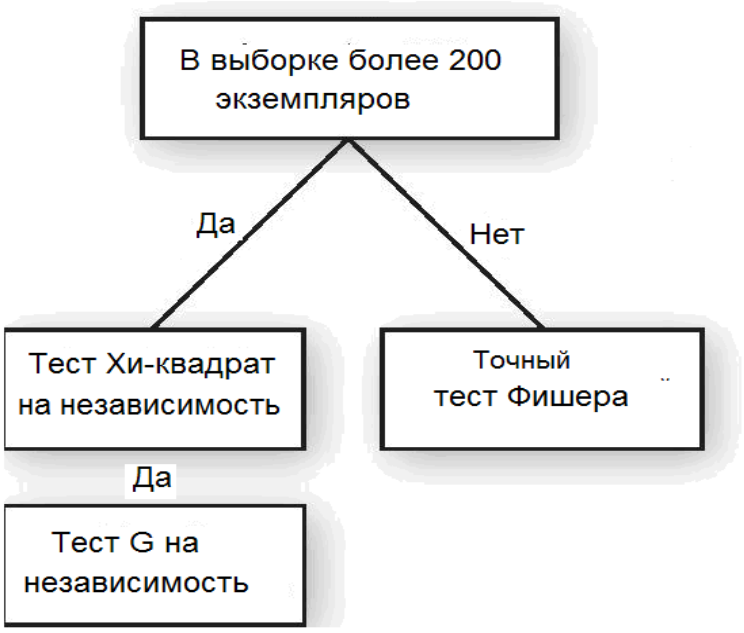
Получены три следующих расчета доверительных интервалов:

```
     inf      r    sup
A -0.978 -0.846 -0.419
B  0.241  0.789  0.957
C  0.593  0.904  0.985
```

Интервал группы A не перекрывает интервалы групп B и C, следовательно A значимо отличается от B и C. Интервалы групп B и C перекрывают друг на друга. Следовательно B и C значимо не отличаются.

86. Ассоциация между двумя качественными переменными

Выбор соответствующего теста



Точный тест Фишера здесь как всегда наиболее надежный, но время его расчета значительно увеличивается с ростом эмпирических численностей. Если количество экземпляров достаточно велико, то приближение, сделанное тестами χ^2 и G является достаточно удовлетворительным, чтобы их можно было использовать. Результаты этих двух тестов очень похожи; поэтому выбор между ними осуществляется скорее по привычке, чем по статистическим причинам.

Данные должны быть представлены в виде таблицы сопряженности, где строки соответствуют категориям переменной A, а столбцы – категориям переменной B:

		Переменная B		
		Класс 1	...	Класс *c*
Перемен-ная A	Класс 1			
	...			
	Класс *k*			

B в каждой строке представлено число объектов относящихся одновременно к соответствующим классам (категориям) A и B.

Эта таблица может быть получена следующим образом:
`tab.cont <- table(переменнаяA, переменнаяB)` где `переменнаяA` и `переменнаяB` – векторы, содержащие значение каждого наблюдения для каждой переменной величины (в том же порядке).

Точный тест Фишера (непараметрический). Чтобы выполнить тест, можно использовать функцию `fisher.test(tab.cont)`, где `tab.cont` – таблица сопряженности.

Если появляется предупреждение out of workspace, то следует увеличить значение аргумента `workspace` (по умолчанию `workspace=200000`). Если появляется другое предупреждение, то это связана с нехваткой вычислительных ресурсов из-за слишком сложной таблицы, которую надо анализировать. Тогда следует рассмотреть возможность использования теста χ^2 или теста G.

Статистически значимая величина *p* указывает, что обе переменные величины не являются независимыми, не уточняя классы, которые ложатся в основу этой связи. Чтобы определить, какие именно категории отличаются между собой, используют функцию: `fisher.multcomp(tab.cont)*`. Функция выполняет точный тест Фишера для каждой возможной таблицы сопряженности 2 x 2, начиная с `tab.cont`. Эти результаты следует проанализировать и выделить классы, которые систематически дают статистически значимую *p*-величину. Именно эти классы и являются связанными.

Может случиться, что вопреки общему тесту множественные сравнения не показывают никаких статистически значимых различий. В этом случае, наиболее разумным решением будет считать, что не можем знать, какие различающиеся категории ответственны за отклонение от нулевой гипотезы в общем тесте.

ПРИМЕР

Наблюдаемые данные – цвет волос (A) и глаз (B) у 116 человек:

```
> tab.cont
         голубой зеленый карий
блондин      25       6     9
брюнет       10      15    16
рыжий        12      14     9
> fisher.multcomp(tab.cont)*
   Pairwise comparisons by Fisher's exact test for count data
   data: tab.cont
                голубой:зеленый голубой:карий зеленый:карий
блондин:брюнет         0.02211       0.03435        0.7793
блондин: рыжий         0.03435       0.44655        0.4801
брюнет: рыжий          0.77935       0.44655        0.5362

P value adjustment method: fdr
```

Классы, которые здесь связаны между собой: `блондин` и `голубой`.

Тест χ2 на независимость (непараметрический). Чтобы провести тест, используют функцию: `chisq.test(tab.cont)`. При статистически значимой величине *p* выполняют множественные парные сравнения `fisher.multcomp(tab.cont)*`.

Замечание: тест χ^2 на независимость – это также тест на соответствие нулевому значению коэффициента ассоциации Крамера (см. п. **83**). Этот тест может быть осуществлен функцией `cramer.test(переменнаяA, переменнаяB)*` или `cramer.test (tab.cont)*`. Функция возвращает в следующем порядке: тест на соответствие нулевому значению, доверительный интервал, рассчитанный бутстрепом, и значение коэффициента ассоциации.

Тест G на независимость (непараметрический). Чтобы провести тест, используют функцию : `G.test(tab.cont)*`. При статистически значимой величине *p* выполняют множественные парные сравнения `fisher.multcomp(tab.cont)*`.

87. Анализ двух взаимозависимых количественных переменных с использованием моделей

Типы модели

Используемая модель – линейная регрессия на основе *метода наименьших прямоугольников*, которая отличается от классической линейной регрессии на основе метода наименьших квадратов МНК (см. п. **76**). Действительно, в линейной регрессии МНК одна из переменных интерпретируется как объясняющая, в то время как другая – как объясняемая. В линейной регрессии на основе наименьших прямоугольников обе переменные величины рассматриваются совершенно равноправно и ни одна из них не объясняет другую. О таких переменных величинах говорят, что они *взаимозависимые*.

Фактор может быть добавлен к регрессии, чтобы определять параметры, определяющие различные условия эксперимента.

Построение модели

Чтобы построить модель используют функцию:
`модель <- least.rect(переменная.y~переменная.x)*`, где `переменная.y` и `переменная.x` – векторы, содержащие значение каждого наблюдения для каждой переменной величины (в одном и том же порядке). Порядок представления переменных в формуле имеет только графическое значение: `переменная.x` располагается по шкале абсцисс, а `переменная.y` – по шкале ординат. Обе переменные можно переставить местами без последствий для модели.

Чтобы оценить параметры модели в зависимости от различных уровней фактора:
`модель <- least.rect(переменная.y~переменная.x|фактор)*`, где `фактор` – вектор, содержащий условия проведения каждого наблюдения (в том же порядке, что и две другие переменные величины).

Проверка адекватности модели

Прежде чем проводить дальнейший анализ, необходимо проверить, насколько хорошо модель подогнана к анализируемым данным. Этот этап фундаментален для любой модели, так как любой тест, проводимый на недостаточно адекватной модели, просто ненадежен (или не соответствует действительности). См. п. **41** для подробного объяснения методики этой проверки.

Оценка значимости параметров и тесты

Значения оценок параметров регрессии представлены функцией `summary (модель)` в таблице `Coefficients`. Функция возвращает значение и доверительный интервал свободного члена `(Intercept)` и коэффициента угла наклона (который носит имя переменной величины `x`). Если в формулу модели был включен фактор, таблица содержит для каждого из этих двух параметров отдельную строку для каждого уровня фактора.

В рамках линейной регрессии на основе наименьших прямоугольников, сравнивается чаще всего оценка коэффициента угла наклона с теоретическим значением 1 (например, в аллометрии это соответствует изометрической связи между двумя сравниваемыми органами или структурами). Результат этого теста также возвращается функцией `summary(модель)`. Чтобы изменить теоретическое значение, нужно добавить аргумент `theo=значение` к вызову функции `least.rect ()*`. Если в формулу модели был включен фактор, тест выполняется для каждого угла наклона.

Не существует теста, который сравнивал бы несколько углов наклона или свободных членов. Между тем, если в модели был определен фактор и 95%-е

доверительные интервалы для коэффициентов угла наклона (или свободных членов) не перекрывают друг друга для двух различных уровней, то по определению эти оценки параметров значимо различаются с критическим порогом $\alpha = 5\ \%$ (см. п. **15**).

Прогноз на основе модели

Смысл моделей – не только показать силу зависимости, связывающую две анализируемые переменные `x` и `y`, но и предсказать значение, которое бы приняла переменная `y` при известных значениях переменной `x`. Поэтому, чтобы оценить величину `y`, требуется зафиксировать значения `x` и фактора, если он присутствует в модели.

Для прогнозирования могут быть использованы два метода, основанных на функции `predict()`:

- значение переменной `x` и фактора дается непосредственно в теле функции, в виде списка: `predict(модель,newdata=list(переменные))`, где `переменные` – это последовательность `переменная.x = значение, фактор =значение ...`;
- создать таблицу, содержащую столбцы переменной `x` и при необходимости фактора (имена столбцов должны строго совпадать с именами переменных модели), и заполнить каждую строку значениями, для которых должно быть сделано предсказание. Таким образом, делается прогноз по каждой строке таблицы: `predict(модель, newdata = таблица)`.

ПРИМЕР

Рассмотрим модель, включающую переменные величины `переменная.x` и `переменная.y` , меняющиеся от 0 до 30, и `фактор`, состоящий из двух уровней (`A` и `B`):

```
> модель <- least.rect(переменная.y~переменная.x|фактор)*
```

Можно предсказать значения переменной `y` таким образом :

```
> predict(модель,newdata= list(фактор="A",
                                переменная.x =10)).
```

Или для нескольких прогнозов :

```
> predict(модель,newdata=list(фактор =c("A","B"),
                              переменная.x =c(10,10))
```

Или предварительно создать таблицу следующего типа:

```
> таблица
  Фактор переменная.x
1      A          10
2      B          10
> predict(модель, newdata = таблица)
```

Графики

Чтобы проиллюстрировать взаимосвязь между переменными `x` и `y`, необходимо выполнить три этапа:

1. Нарисовать точки наблюдения: `plot(переменная.y~переменная.x)`.
2. Создать вектор, имеющий те же минимум и максимум, что `переменная.x`, но с очень маленьким интервалом между его значениями: `x <- seq2(переменная.x)*`.
3. Добавить на график прямую линию модельных значений функцией `lines(x,predict(модель,newdata= x))`.

Здесь определение всех точек модельной прямой основано на прогнозе значений, которые принимает `переменная.y` для каждого значения вектора `x`. Большое количество значений этого вектора и очень маленький интервал, который их отделяет, определяет гладкость графика (который на самом деле состоит из сотен точек,

соединенных между собой). Если модель содержит фактор, то необходимо зафиксировать все данные `x` для графика для одного из уровней фактора.

Примечание: функция `lines()` может быть использована несколько раз, чтобы начертить несколько кривых на одном графике, например, для нескольких уровней одного и того же фактора, включенного в модель.

ПРИМЕРЫ

Рассмотрим следующую модель :

```
> модель <- least.rect (переменная.y~переменная.x|фактор)*
```

Шаг 1 : нарисуем точки, соответствующие наблюдаемым данным :

```
> plot(переменная.y~переменная.x)
```

Шаг 2 : создадим вектор `x` :

```
> x <- seq2(переменная.x)
```

Шаг 3 : добавляем модельную прямую взаимозависимости переменных `x` и `y`.

Выбираем один из уровней фактора, включенного в модель:

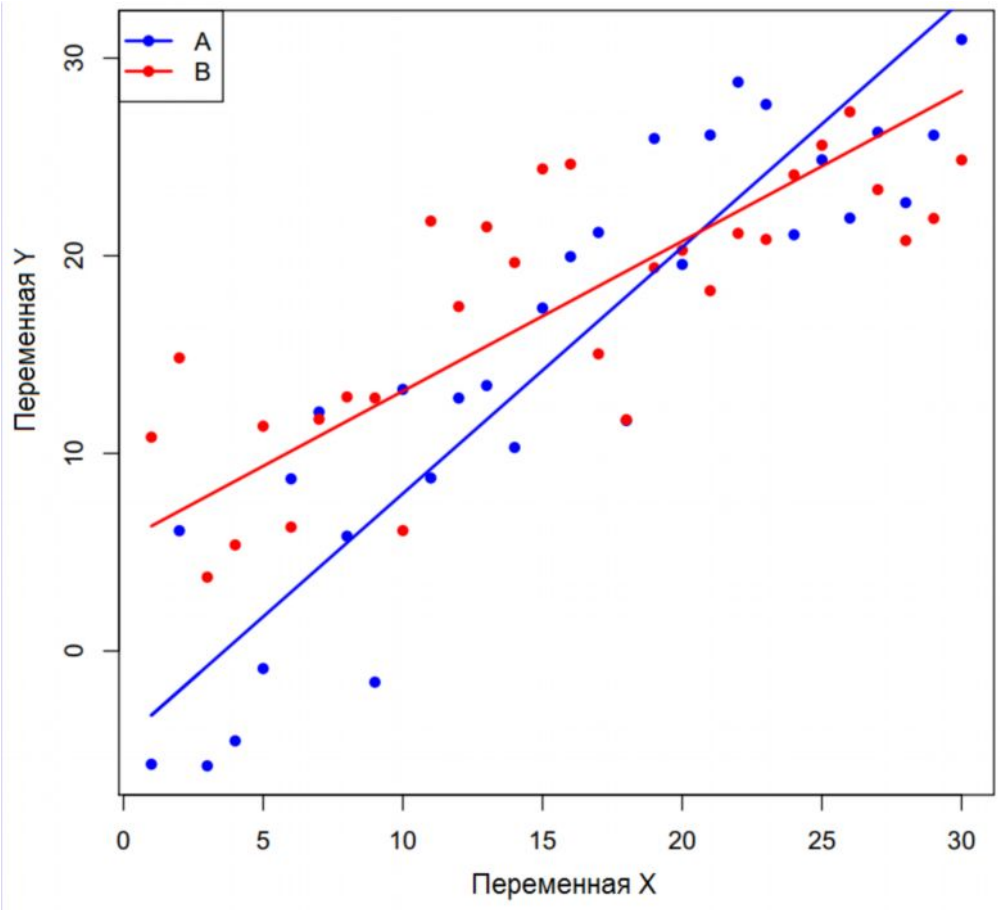
```
> lines(x,predict(модель,newdata=list(переменная.x = x,
      фактор =rep("A",length(x)))), col = 2)
```

Функция `rep()` является очень полезной в этой ситуации, поскольку она позволяет очень просто создать вектор той же длины, что и `x`, содержащий одно и тоже повторяющееся значение `A`.

Чтобы добавить модельную прямую, соответствующую уровню `B` фактора :

```
> lines(x,predict(модель,newdata=list(переменная.x = x,
      фактор =rep("B",length(x)))), col = 2)
```

Ниже представлен график для двух уровней заданного фактора.



МНОГОМЕРНЫЙ СТАТИСТИЧЕСКИЙ АНАЛИЗ

Как ориентироваться в "джунглях" многомерного анализа?

Вначале нужно задаться несколькими очень простыми вопросами. Основной из них: *сколько наборов данных (или "таблиц") есть для анализа* ?

1. **Один**. Таким образом, вся информация, используемая в анализе, содержится в единственном наборе данных.

Примечание: если есть какая-то другая переменная вне этого набора данных, которую следует учитывать в анализе (например, заранее составленные группы), то это – уже два набора данных (даже если второй набор содержит только одну переменную).

Следующий вопрос: *какова цель анализа* ?

(а) необходимо объединить отдельные объекты в группы (классы, кластеры). Для этого используется метод *кластеризации*, который будет объективно составлять группы.

(б) обобщить информацию, содержащуюся в наборе данных, для визуализации и интерпретации. Для этого используется метод *ординации*.

Примечание: всегда можно проверить, являются ли "внешние" переменные (т.е. известные заранее, но не учитываемые в анализе) существенно "коррелированными" с результатами ординации (см. п. **91**).

2. **Два.** В этом случае речь идет о *каноническом анализе*.

Следующий вопрос: *какова взаимосвязь между двумя наборами данных* ?

(а) один считается объяснением другого, т. е. один набор данных является "объясняющим", а другой – "объясняемым". В этом случае мы используем *асимметричный анализ*. Как и в случае любой модели (униарной или многомерной), объясняющих переменных должно быть намного меньше, чем число анализируемых объектов, и они не должны слишком коррелировать друг с другом.

(б) оба набора данных рассматриваются одинаково, на эквивалентной основе. В этом случае используется *симметричный анализ*.

Примечание: некоторые авторы используют термин "*канонический*" только для симметричного анализа.

3. **Больше двух наборов**. Здесь мы находимся все еще в рамках канонических методов. Все представленные в этом пособии алгоритмы относятся к симметричному анализу.

Три замечания:

- Для некоторых алгоритмов анализа может потребоваться предварительная обработка данных (см. п. **88**). Ситуации, в которых это может быть интересным (или необходимым), обсуждаются в соответствующих разделах.
- Независимо от анализа *"набор данных"* может состоять либо из исходных переменных (предварительно обработанных или не обработанных), либо из матрицы расстояний (см. п. **100**), либо из сводных переменных, полученных предыдущим методом многомерного анализа (например, координаты отдельных объектов по осям, созданным ординацией, см. п. **90**). Случаи, когда этот последний вариант может быть интересным, указаны в соответствующих разделах.
- Если несколько массивов анализируются одновременно, то их строки должны быть одинаковыми по числу и порядку, т. е. различные таблицы представляют собой наборы переменных, измеренных на одних и тех же объектах.

88. Предварительная обработка количественных данных

Прежде чем приступить к многомерному анализу, обычно следует рассмотреть целесообразность "пре-процессинга" данных. Ситуации, в которых это может быть интересным (или необходимым), обсуждаются в соответствующих разделах.

В целом, распространены три возможных этапа предварительной обработки (первый должен обязательно предшествовать остальным, в то время как два других не чувствительны к последовательности операций):

Преобразования: большое количество методов многомерного анализа учитывают взаимодействия на основе линейной модели, т.е. принимается предположение о линейности связей между переменными величинами набора данных (даже если объясняющие переменные величины рассматриваются как объект асимметричного анализа). Это часто оказывается справедливым, но далеко не всегда. Аналогичным образом, многие методы предполагают, что переменные имеют, по крайней мере, симметричное распределение, что также не всегда проверяется. Наконец, многие анализы предполагают также аддитивные связи между переменными, тогда как в фактическом наборе данных они довольно часто оказываются мультипликативными. Чтобы решить эти три проблемы, очень эффективным может оказаться преобразование данных. Наиболее распространенным и полезным преобразованием является логарифмирование (независимо от его базы), выполняемое достаточно просто: `таблица <- log(таблица)` , где `таблица` – массив исходных данных. Если набор данных содержит 0, то альтернативой логарифмированию является трансформация извлечением корня 4-ой степени (т. е. $\sqrt[4]{x}$), которая предпочтительнее классического log(x + 1), поскольку добавление константы может повлиять на небольшие значения. Для его реализации подается команда:
`таблица < - таблица^(1/4)`.

Примечание: если набор данных имеет композиционный тип (т. е. сумма значений каждого объекта всегда равна 1 или 100 %), то перед многомерным анализом преобразовать данные абсолютно необходимо. Наиболее распространенным преобразованием в этом случае является центрированное логарифмическое отношение (*Centered LogRatio* – CLR), выполняемое функцией из пакета `Hotelling`:
`таблица <- clr(таблица)`. Если набор данных содержит нули, нет альтернативы добавлению константы ко всем значениям. Выберите ее так, чтобы она была на несколько порядков меньше, чем наименьшее значение в массиве.

Центрирование: состоит в том, чтобы вычесть из каждого значения в таблице среднее значение столбца, в котором это значение находится (следствием этого является то, что все столбцы затем будут иметь нулевое среднее значение). Это позволяет привести к единому стандарту порядок величин различных переменных, что особенно важно, если эти порядки сильно различаются. Для анализа с использованием матрицы расстояния, на котором основаны многие алгоритмы классификации или ординации, этот эффект потенциально очень важен (за исключением случаев, когда необходимо учесть в анализе различия в размере или численности). Для методов ординации, основанных на массиве исходных переменных, центрирование не имеет никакого эффекта кроме упрощения внутренних вычислений.

Примечание: в более общем плане центрирование данных состоит в том, чтобы вычесть любое произвольное значение, постоянное для каждого столбца. Случаи, когда используется не среднее значение, очень редки и основаны на достаточно конкретных предположениях.

Нормировка: состоит в том, чтобы разделить каждое значение массива на константу, собственную для каждого столбца. Существует множество методов нормировки, цели которых не всегда одинаковы. В целом, наиболее распространенные методы направлены на уменьшение разницы в изменчивости переменных. Задача состоит в том, чтобы сбалансировать вес, заданный каждой переменной в анализе, потому что во многих методах этот вес зависит от дисперсии. Наиболее распространенным методом является деление вектора (столбца таблицы) на его стандартное отклонение, что приводит все переменные к единичной дисперсии. То есть, мы исключаем все априорные предположения из анализа и придаем одинаковый вес всем переменным. За исключением конкретных случаев, этот тип преобразования чаще всего очень полезен.

Для центрирования на основе среднего значения и нормировки на основе стандартного отклонения используют функцию: `таблица <- scale(таблица)`. См. `? scale` для информации о других способах центрирования и нормировки.

Примечание: Выражение "*центрированная-нормированная таблица*" подразумевает центрирование на основе среднего и нормировку на стандартное отклонение. Это также соответствует выражению "*стандартизированная* (*стандартная*) *таблица*".

89. Интерпретация корреляционного круга

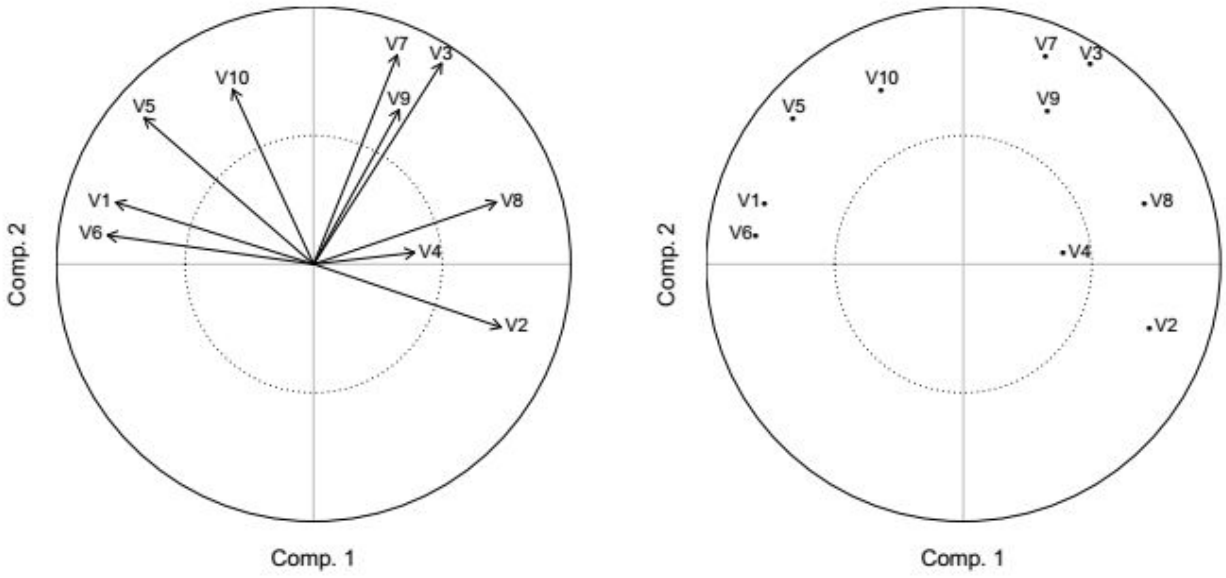
Круг корреляций – классический график, позволяющий интерпретировать результаты большинства методов многомерного анализа. Он позволяет одновременно показывать как направления взаимосвязи между анализируемыми величинами и основными осями координат (т.е. положение стрелок), так и непосредственно уровень корреляции между самими переменными.

Его интерпретация основывается на трех простых правилах:

- Чем больше длина стрелки, которая является синтезом обеих главных осей координат, тем бóльший информационный вклад вносится этой переменной. Поэтому при интерпретации концентрируется внимание на наиболее длинных стрелках (т.е. на наиболее информативных переменных).
- Угол между двумя стрелками (или между стрелкой и осью координат) указывает на величину корреляции между двумя переменными (или между переменной и осью координат):
  - острый угол = положительная корреляция (0° = корреляция равна 1);
  - прямой угол = никакой корреляции (90° = корреляция 0);
  - тупой угол = отрицательная корреляция (180° = корреляция равна -1).
- Чем ближе объект расположен к концу стрелки (отсчет ведется по перпендикуляру, опущенному из анализируемой точки на эту стрелку), тем бóльшее воздействие оказывается на объект этой переменной величиной (и наоборот).

Когда одновременно размещают на графике облако точек объектов и круг корреляций, то таким образом можно выделить переменные, которые структурируют распределение объектов, а также биологически интерпретировать оси координат.

Замечание: иногда переменные величины представляются не стрелками, а точками на круге корреляций. Это ровным счетом ничего не меняет в интерпретации, так как эти точки обычно соответствуют концам стрелок. Использование точек может обеспечить более читаемый график, если переменных достаточно много, но тогда углы между переменными (или между переменными и осями) уже не отображаются так четко.



90. Использование осей ординации в качестве значений переменных для дальнейшего анализа

Всегда можно использовать оси ординации (PCA, PCoA, смешанный анализ и др.) в качестве объясняющих переменных в другом анализе. Это может позволить:

- резко уменьшить число переменных (поскольку, в случае эффективной редукции, первые оси многомерной ординации сосредоточат в себе большую часть информации, содержащейся в данных);
- устранить корреляционные связи между оригинальными переменными, так как один из принципов многомерного анализа является ортогональность (пространственная перпендикулярность) осей координат, которые очень мало взаимосвязаны и не включают информацию, связанную с другими осями;
- перевести категориальные переменные в количественные переменные;
- перевести матрицы расстояние в таблицы переменных.

На практике это позволяет осуществить (1) разбиение данных на нескольких групп относительно одной или двух осей (см. п. **91**); (2) сформировать набор из одной обобщающей переменной, с которой все остальные переменные являются коррелированными; (3) сравнить результаты нескольких ординаций и т.д. Возможностей много.

Примечание 1: если используется информация, основанная на координатах объектов по осям nMDS, то выполняемый одномерный анализ должен быть непараметрическим, так как результаты неметрического многомерного шкалирования носят полуколичественный характер (т.е. относительны, см. п. **102**). По той же причине, большинство анализов, проводимых с несколькими таблицами, не являются значимыми с nMDS (за исключением прокрустового анализа, см. п. **112**).

Примечание 2: координаты объектов по осям PLS-DA не должны использоваться для проверки разделяемости на несколько групп. Для этого должны быть использованы тесты, описанные в п. **105**.

Получение координат объектов

Для всех методов многомерного анализа, представленных в этом путеводителе (а также и многих других), можно получить координаты объектов для одной или нескольких осей с помощью функции `MVA.scores(анализ)*`, где `анализ` – это имя объекта многомерного анализа (координаты находятся в узле `$coord` списка, возвращаемого функцией, выполнившей этот анализ). Аргументы `xax` и `yax` позволяют уточнить, на какие оси эти координаты должны быть представлены; по умолчанию они равны 1 и 2 соответственно. Если желательно постепенно получить координаты для трех и более осей, то их можно сохранить и объединить с помощью функции `cbind()`.

Пример.

Мы хотим получить координаты объектов относительно первых трех осей ординации, полученной методом PCA (с именем `PCA`). Сначала получаем координаты относительно первых двух главных осей :

```
> coord12 <- MVA.scores(PCA)$coord*
```

Добавляем третью ось :

```
> coord3 <- MVA.scores(PCA,xax=3,yax=NULL)$coord*
```

Параметр функции `yax=NULL` указывает, что необходимо получить координаты только одной оси (обозначенной аргументом `xax`). Далее мы объединяем в одну таблицу координаты по всем трем осям :

```
>   coord <- cbind(coord12,coord3)
```

Внимание: некоторые методы анализа, такие как CA (см. п. **97**) и CCA (см. п. **106**), рассчитывают координаты для строк и столбцов в виде одной таблицы, в то время как другие возвращают координаты объектов в нескольких списках одновременно (большинство анализов – в двух или более таблицах). Аргументы `set` и `space` функции `MVA.scores()*` позволяют уточнить, какие данные следует предоставить. См. `? MVA.scores` для получения подробной информации.

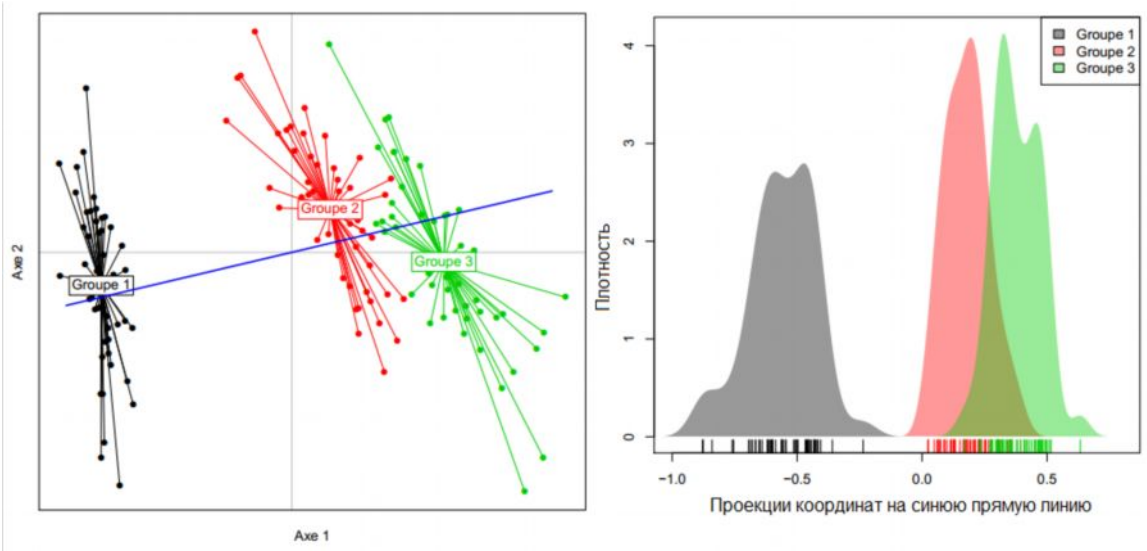
Может случиться, что мы хотим получить координаты объектов в каком-то определенном направлении относительно осей (например, по диагонали между двумя осями), поскольку это направление представляется наиболее значимым для последующего анализа. Эта процедура выполняется в несколько этапов:

1. Построить ординационный график на плоскости относительно рассматриваемых осей с использованием функции `MVA.plot()*` (см. более подробно разделы ниже, относящиеся к различным методам многомерного анализа).

2. Создать объект `прямая <- loc.slp()*`. Функция `loc.slp()*` (без входных аргументов) два раза просит выполнить двойной клик мышкой на графике, чтобы определить интересующее направление, и затем отображает заданную прямую линию функцией `abline(0, прямая)`. Можно повторить операцию `loc.slp()*` для уточнения положения линии.

3. Получить координаты объектов относительно двух основных осей с использованием функции `MVA.scores()*`.

4. Вычислить проекции координат объектов на прямую, определенную на шаге 2 : `coord.proj(координаты, прямая)*`, где `координаты` – это таблица координат, полученная на шаге 3.



Частный случай коинерционного анализа

Анализ коинерции (см. п. **113**) осуществляется с помощью функции `coinertia()` из пакета `ade4`. Эта функция имеет особенность использовать в качестве аргументов только результаты ординаций (а не таблицы переменных), которые чаще всего выполняются с помощью функций этого же пакета. На практике аргументы представляют собой ординации на основе одной таблицы: PCA, CA, MCA или смешанный анализ. Если MCA (см. п. **98**) и смешанный анализ (см. п. **99**) обычно выполняются с помощью пакета ade4, то PCA (см. п. **96**) и CA (см. п. **97**) осуществляется с помощью пакета `vegan`. Чтобы преобразовать результат этих ординаций в объекты, используемые функцией `coinertia()`, примените функцию: `анализ2 <- to.dudi(анализ1)*`, где `анализ1` – это имя PCA, CA или PCoA.

91. Зависимость между ординацией и внешними переменными

Всегда можно проверить, "коррелирован ли" результат ординации с внешними переменными величинами, не учтенными в анализе. В частности, этот тест может проводится относительно заданной факториальной плоскости (т. е. плоскости, состоящей из двух главных осей), либо одного измерения (оси или любой диагонали). Внутренние переменные, рассматриваемые в тесте, являются координатами отдельных объектов в факториальной плоскости или относительно заданной прямой. Внешние переменные могут быть количественными и/или качественными.

Как правило, такой подход применяется к анализам на основе одной таблицы (PCA, CA, PcoA и т.д.), где есть мало априорных ограничений. PLS-DA является исключением: разделяемость между группами не должна тестироваться на основе объектов анализа, а с использованием специального теста (см. п. **105**).

Замечание: результаты неметрического многомерного шкалирования nMDS имеют полуколичественный характер (т. е. относительный, см. п. **102**). И здесь использование непараметрических коэффициентов корреляции – единственно корректный подход, чтобы оценить взаимосвязь с внешними переменными. Это заставляет проводить вычисления только для одного измерения (а не факториальной плоскости, так как применяемое двумерное тестирование является параметрическим).

Тесты для одного измерения

Начинают с того, что формируют вектор координат точек относительно выбранного измерения (основной оси или любой диагонали, см. п. **90**). Новая созданная таким образом переменная является неограниченной количественной величиной, которая может затем использоваться в любом анализе, одномерном или двумерном.

Тесты для двух измерений

Чтобы выполнить тест (непараметрический), используют функцию из пакета `vegan`: `envfit(формула)`. См. п. **40** для подробного объяснения записи конструкции формулы. В этой формуле, отклик – таблица, содержащая координаты объектов, которые надо тестировать, относительно обеих осей (см. п. **90** для инструкций по ее формированию). Если ординация была создана пакетом `vegan`, то просто указывается имя соответствующего объекта.

В случае количественной переменной мы можем увидеть, каким образом она коррелирует с многомерной ординацией с использованием процедуры из трех этапов:

1. Рисуем ординационную диаграмму в пространстве двух осей с использованием функции `MVA.plot()*` (см. более подробно разделы ниже, относящиеся к различным методам многомерного анализа).
2. Сохраняем результат выполнения функции `envfit()` в объекте: `тест <- envfit(формула)`.
3. Добавляем на ординационную диаграмму информацию о внешних переменных: `plot (тест)`. Длина стрелок произвольна, но интерпретация та же, что и для круга корреляций (см. п. **89**).

Замечание: эта процедура идентична добавлению количественных внешних переменных на корреляционный круг и отличается только первым этапом. Длина стрелок внешних переменных там также произвольна, но интерпретация в терминах корреляции корректна. Отметим, что этот тест не учитывает взаимодействия между самими внешними переменными.

В случае оценки влияния качественной переменной, разделы, посвященные различным методам многомерного анализа, объясняют, как можно добавить группирующий фактор при визуализации ординационной диаграммы.

МНОГОМЕРНЫЙ АНАЛИЗ ДАННЫХ ОДНОЙ ТАБЛИЦЫ

Кластерный анализ

Чтобы кластеризация была выполнена корректно, она выполняется в четыре этапа в порядке приводимых ниже разделов п. **92-95.**

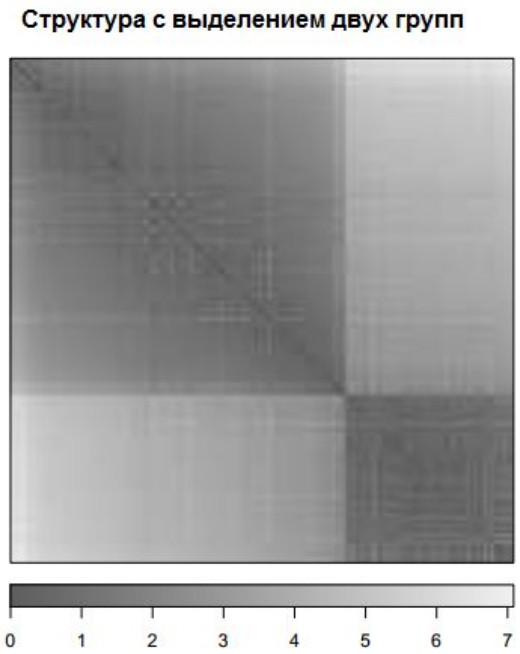
92. Кластеризация – этап 1: оценка тенденции данных к группировке

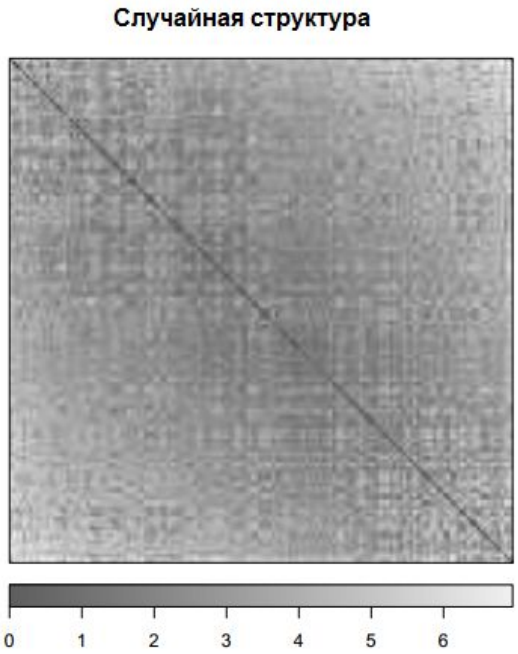
Любой метод кластерного анализа всегда даст какой-то результат (т. е. группы будут всегда сформированы). Если анализируемая таблица переменных представляет собой полностью случайную структуру, то разбиение даже на две группы фактически не имеет смысла. Прежде чем выполнять кластеризацию, необходимо убедиться, что в наборе данных имеется определенная группообразующая структура (англ. *clustering tendency*).

Можно использовать два метода, дополняющих друг друга, для проверки наличия тенденции к группировке: графический и статистический.

Графический метод

Метод называется "визуальная оценка тенденции к кластеризации" (VAT – *Visual Assessment of cluster Tendency*). Для его реализации используют функцию из пакета `seriation`: `dissplot(мат.расст)`, где `мат.расст` – матрица расстояний, вычисленная по таблице исходных переменных (см. п. **100**), или непосредственно сам набор данных. Функция возвращает график, напоминающий матрицу расстояний, в которой последовательность объектов автоматически перестроена для их группировки по наибольшей схожести. Интерпретация проста: если по главной диагонали появляются, по крайней мере, два темных квадрата, то в наборе данных есть группы (но нельзя сказать точно, сколько их на данный момент). Если график представляет однородное серое поле, то объекты структурированы случайным образом. В первом случае для выявления групп можно использовать ту или иную процедуру кластеризации, во втором случае анализ на этом этапе останавливается.





Статистический метод

Метод может использоваться только с исходным массивом переменных и основывается на статистике Хопкинса. Значение 0.5 для этой статистики указывает на случайное структурирование отдельных объектов, в то время как значение, которое отличается от него, приближаясь к 0, указывает на наличие групп.

Чтобы вычислить статистику Хопкинса, используют функцию из пакета `clustertend`: `hopkins(таблица, n=10)`, где `таблица` – исходный набор данных, аргумент `n` – число, используемое алгоритмом вычислений (его максимальное значение равно количеству строк `таблица` - 1).

93. Кластеризация – этап 2: оценка оптимального числа групп

Предполагается, что предыдущий шаг для достижения корректной кластеризации был успешно пройден (см. п. **92**).

Определение оптимального количества групп является одним из наиболее важных этапов в процедуре кластеризации. Стратегия основана на использовании индексов, вычисляемых для последовательности возможного количества групп (2, 3, 4...). Наилучшее значение индекса, указывает на оптимальное число кластеров.

К сожалению, существует, по крайней мере, около 30 индексов, которые не всегда возвращают одинаковое количество оптимальных групп. Поэтому идея состоит в том, чтобы рассчитать все эти индексы и использовать наиболее часто возвращаемое значение в качестве оптимального числа групп. Процедура немного отличается в зависимости от того, используется ли исходный массив переменных или непосредственно матрица расстояний (см. п. **100**).

Использование исходного массива переменных

Для реализации поиска числа групп используется функция из пакета `bclust`:

```
nc <- NbClust(таблица, diss=мат.раст, distance=NULL,
                                        method="ward.D2"),
```

где `таблица` – исходный набор данных и `мат.раст` – матрица расстояний, рассчитанная на его основе (см. п. **100**). Результат функции выводится в объект (здесь он носит имя `nc`), который используется для различных вариантов отображения результатов. Минимальное количество тестируемых групп – 2, а максимальное – 15. Чтобы изменить эти значения, используйте аргументы `min.nc` и `max.nc`.

Функция вычисляет значение 26 индексов для последовательности от 2 до 15 групп (по умолчанию) и возвращает два дополнительных блока результатов:

- резюме по оптимальному числу групп, оцененных на основе различных индексов (их количество иногда может быть меньше 26). Наиболее часто встречаемое число групп считается оптимальным.
- частотную диаграмму встречаемости числа групп. Наибольшее значение указывает на оптимальное количество кластеров.

Использование матрицы расстояний

Так можно оценить только пять индексов. Необходимо выполнить функцию:

```
nc <- NbClust(diss=мат.раст, distance=NULL, method="ward.D2",
                                          index = индекс)
```

последовательно для каждого из них, включая `индекс = "frey", "mcclain", "cindex", "silhouette"` или `"dunn"`. Каждый раз обращайте внимание на оптимальное количество групп, указанное в узле `$Best.nc` возвращаемого списка под названием `Number_clusters`. Наиболее часто возвращаемое значение из пяти индексов считается оптимальным числом групп.

94. Кластеризация – этап 3: выбор метода кластеризации

Предполагается, что предыдущие шаги для достижения корректной кластеризации были успешно пройдены (см. п. **92-93**).

Существует два больших семейства методов кластеризации:

- Методы неиерахического разделения, для которых необходимо априори указать необходимое число групп, (оцененное с помощью специальных алгоритмов, см. п. **93**). Эти методы стараются сформировать такие группы наилучшим способом.
- Иерархические методы, для которых нет необходимости задавать априори количество групп. Они приводят к полному дереву филогенетического типа, называемому дендрограммой. Затем это дерево "обрезается" до оптимального количества групп (см. п. **93**).

Метод неиерахического разделения

Мы рекомендуем использовать метод K-медоидов или PAM (*Partitioning Around Medoids*), как наиболее надежную версию известного метода K-средних.

Для его реализации используют функцию из пакета `cluster` :
`кластер <- pam(мат.раст, k=nb)`, где `мат.раст` – матрица расстояний, вычисленная по таблице переменных (см. п. **100**), или непосредственно сам набор данных, `nb` – желаемое количество групп. Группа, к которой принадлежит каждый объект, возвращается в `кластер$clustering`.

Для больших наборов данных см. информацию по функции `?clara`.

Иерархическая кластеризация

Чтобы построить дендрограмму, существует две идеи :

- последовательно объединять индивидуальные объекты и их группы во все более крупные подмножества, пока не образуется только одна группа, объединяющая их всех. Это – семейство иерархических аггломеративных методов.
- стартуя от одного корня, объединяющего все объекты, на каждом шаге делить образующие группы по степени их гетерогенности, пока не образуется столько групп, сколько есть объектов. Это – семейство иерархических дивизимных методов.

Иерархическая агломеративная кластеризация. На первом шаге создается дендрограмма с использованием функции из пакета `cluster`:
`dendro <- agnes(мат.раст, method="ward")`. Аргумент `method` уточняет алгоритм, по которому происходит группировка объектов; одним из наиболее распространенных является метод Уорда (см. `? agnes` для информации о других возможностях). Для визуализации дендрограммы можно использовать функцию из пакета `factoextra`: `fviz_dend(dendro)`.

Второй шаг – обрезать дендрограмму на соответствующем уровне, чтобы получить оптимальный состав групп. Для этого воспользуйтесь командой:
`кластер <- cutree(dendro, k=nb)`. Функция возвращает ссылки на группы, в которые включен каждый объект. Чтобы представить эти группы на дендрограмме, добавьте аргумент `k=nb` в функцию `fviz_dend()`.

Иерархическая дивизимная кластеризация. Оба шага аналогичны описанным выше процедурам, за исключением того, что дендрограмма строится в обратном направлении. Сначала создается дендрограмма: `dendro <- diana(мат.раст)`, которая визуализируется функцией: `fviz_dend(dendro)`. Затем дендрограмма обрезается, чтобы прикрепить каждый объект к группе: `cutree(dendro, k=nb)`. И в этом случае можно представлять группы на дендрограмме, используя аргумент `k=nb` в функции `fviz_dend()`.

Какой метод выбрать?

Можно напрямую выполнить сравнение результатов кластеризации тремя описанными выше методами с использованием следующей процедуры. Сначала выполняется проверочный тест с использованием функции из пакета `clValid`:

```
тест <- clValid(таблица, nClust=nb, clMethods=
        c("pam", "agnes", "diana"),
        validation=c("internal","stability"),method="ward"),
```

где `таблица` – исходный набор данных и `nb` – оптимальное количество групп. Метрика, используемая для вычисления матрицы расстояния, должна являться евклидовым, манхэттенским или на основе корреляции Пирсона (см. п. **100**). Аргумент `validation` уточняет, что три метода кластеризации должны быть сопоставлены между собой как по так называемым внутренним критериям, так и по другим критериям стабильности. Результаты сравнения затем просматриваются через `summary(тест)` в таблице `Optimal Scores`. Для каждого из семи критериев (первые четыре – стабильности, следующие три – внутренние) столбец `Method` содержит ссылку на наиболее эффективный метод. Рекомендуется выбрать тот, который появляется чаще всего.

Примечание: функция `clValid()` использует по умолчанию евклидово расстояние. Используйте аргумент `metric` для выбора другой метрики расстояния.

Если вы не можете выполнить прямое сравнение трех методов, все равно можно это сделать в режиме ручной проверки. Это делается на основе трех внутренних индексов качества кластеризации: ширины силуэта (*Silhouette width*) (см. п. **95**), индекса Данна и индекса связности. Для каждого из трех методов классификации рассчитать эти индексы можно, используя функции из пакета `fpc`: и `clValid` соответственно, где `кластер` - вектор, содержащий индекс группы каждого объекта (в числовой форме):

- Ширина силуэта:

```
cluster.stats(мат.раст,clustering=кластер)$avg.silwidth
```

- Индекс Данна:

```
cluster.stats(мат.раст,clustering=кластер)$dunn
```

- Индекс связности : `connectivity(мат.раст,clusters= кластер)`.

С точки зрения ширины силуэта и индекса Данна метод, дающий максимальное значение, является лучшим. С точки зрения индекса связности лучшим является метод, дающий минимальное значение.

95. . Кластеризация – этап 4: проверка результатов группировки

Предполагается, что предыдущие шаги для достижения корректной кластеризации были успешно пройдены (см. пп. **92-94**).

Методы кластеризации обычно очень эффективны, но может случиться так, что некоторое количество объектов включается в сформированные группы с известной долей неопределенностью (или, по крайней мере, мы это предполагаем, поскольку, как правило, нам неизвестно об "истинной" группировке). Чтобы идентифицировать эти объекты, мы можем использовать *Индекс ширины силуэта*. Он рассчитывается для каждого объекта и варьирует от -1 до 1 :

- значение, близкое к 1, указывает на объект, который, скорее всего, надежно попал в "свою" группу;
- значение, близкое к 0, указывает на объект, "застрявший" между двумя группами;
- отрицательное значение указывает на объект, который, вероятно, вообще плохо кластеризуется в анализируемой совокупности.

Для расчета значения индекса ширины силуэта используют функцию из пакета cluster: `sil <- silhouette(кластер, мат.раст)` , где `кластер` – вектор, содержащий индекс группы каждого объекта (в числовой форме) и `мат.раст` – матрица расстояния (вычисляется из набора переменных или представляет собой непосредственно набор данных, см. п. **100**). Функция возвращает таблицу, в каждой строке которой для каждого объекта указан номер группы, назначенный методом кластеризации (столбец `cluster`), номер ближайшей соседней группы (столбец `neighbor`) и индекс ширины силуэта (столбец `sil_width`). Если объекту соответствует отрицательное значение индекса, то можно вручную изменить назначенную ему группу в векторе `кластер`, заменив ее ближайшей группой (к которой он, скорее всего, и принадлежит).

Чтобы помочь себе в анализе значений индекса силуэта, можно :

- представить результаты графически : `plot(sil)`. На графике также будут показаны численность по группе, средний Индекс силуэта по группе и средний Индекс силуэта всех групп
- отсортировать их в том же порядке, что и на графике с использованием функции из пакета cluster: `sortSilhouette(sil)`.

Пример.

Мы получили следующие значения индекса силуэта (только для первых пяти объектов)

```
     cluster  neighbor       silwidth
[1,]       3         2     0.46166680
[2,]       2         3     0.17076888
[3,]       2         3    -0.04842929
[4,]       3         2     0.66999160
[5,]       3         2     0.46112097
```

Объект 3, который был отнесен к группе 2, имеет отрицательное значение индекса. Поэтому он, вероятно, относится не к группе 2, а к ближайшей группе, т. е. к группе 3. Таким образом, группу этого объекта можно изменить в векторе `кластер` (который содержит номера групп):

```
> кластер [3] < - 3
```

Ординация на основе одной таблицы исходных переменных

Ординация может быть выполнена непосредственно с таблицей **исходных переменных** или на основе **матрицы дистанций**. Такая матрица содержит расстояния в многомерном пространстве между всеми возможными парами объектов, которые могут быть рассчитаны на основе исходных переменных или получены в ходе непосредственных измерений (например, генетические или географические расстояния). Таким образом, в матрице расстояния нет больше никакой информации о возможных исходных переменных.

Выбор между двумя типами анализа относительно прост. Таблица исходных переменных используется, если ставится задача количественной оценки сравнительной значимости отдельных переменных и на этом основана биологическая интерпретация. Методы, использующие матрицу расстояния и основанные на гипотезе «разделения на основе расстояния» (*isolation-by-distance*), становятся в центре внимания, если задача ограничивается оценкой общего сходства отдельных объектов, пренебрегая при этом ролью описывающих их переменных.

Методы ординация на основе одной таблицы

Выбор метода анализа зависит от характера массива исходных данных и содержащихся в нем переменных:

1. Исходные данные представляют собой **классическую таблицу переменных** (т.е. анализируемые объекты представлены в строках, а переменные – в столбцах. Эти переменные могут быть:

   (а) все **количественные** → используйте *Анализ Главных Компонент* PCA;

   (б) все **номинальные** (т.е. неупорядоченные качественные). Здесь выбор зависит от количества переменных :

   - **две** → реорганизуйте данные в таблицу сопряженности (см. п.) и затем используйте *Анализ Соответствий* CA;
   - **более двух** → используйте *Множественный Анализ Соответствий* MCA;

   (в) нескольких **разных типов** (количественные, качественные номинальные, порядковые) или все порядковые → используйте *Смешанный Анализ*.

2. Таблица представляет собой **пересечение двух факторов**, где каждая клетка содержит количество объектов (т.е. мы также имеем фактически формат таблицы сопряженности) или двоичный отклик (таблица присутствия-отсутствия) → используйте *Факториальный Анализ Соответствий*.

96. Анализ главных компонент (PCA)

Англ. – *Principal component analysis* (*PCA*)

Подготовка данных

Анализ главных компонент работает корректнее, если переменные имеют приблизительно нормальное (по крайней мере, симметричное) распределение и связаны между собой линейными отношениями. Предварительное преобразование данных может в значительной степени улучшить ситуацию (см. п. **88**).

Также рекомендуется, как правило, стандартизировать переменные перед анализом (см. п. **88**). Это придает одинаковый вес всем переменным и формирует результаты с применением коэффициента корреляции, что часто проще. В этом разделе будет предполагаться, что переменные стандартизированы.

Проведение анализа

Для осуществления анализа можно, например, использовать функцию из пакета `vegan`: `PCA< - rda(таблица)` , где `таблица` является массивом исходных данных. Если вы хотите стандартизировать переменные, но не сделали этого раньше, добавьте аргумент `scale=TRUE`. По умолчанию переменные не стандартизируются.

Оценка качества анализа

Цель PCA состоит в том, чтобы сформировать некоторую информационную структуру, которая наилучшим образом "объясняет" общую дисперсию набора данных. Таким образом, для оценки качества анализа необходимо оценить долю общей дисперсии, объясняемую каждой осью. Чтобы получить эти доли, используйте, например, функцию: `MVA.synt(PCA)*`.

Примечание 1: процент дисперсии всегда располагается в порядке убывания (т. е. ось 1 объясняет больше дисперсии, чем ось 2, которая объясняет больше, чем ось 3...).

Примечание 2: нет абсолютного правила, сколько осей следует выделить для интерпретации. Это всегда компромисс между хорошим обобщением информации, содержащейся в исходных данных (которое увеличивается с количеством осей) и легкостью интерпретации (которая уменьшается с количеством осей).

В некоторых случаях процент дисперсии, объясняемый PCA, может быть относительно низким. Это не обязательно означает, что анализ бесполезен. Более значимо то обстоятельство, что расстояния между объектами в многомерном пространстве, образуемом главными компонентами, хорошо соответствуют фактическим расстояниям между объектами в исходной таблице данных. Чтобы проверить это соответствие, можно проанализировать диаграмму Шепарда, которую легко сформировать функцией из пакета `vegan`: `stressplot(PCA)`. Если точки на этой диаграмме располагаются примерно на одной прямой, то расстояния в пространстве PCA пропорциональны фактическим расстояниям, и для интерпретации можно опираться на результаты этого анализа. Если же точки явно далеки от прямой линии, то расстояния искажены, а интерпретация анализа сомнительна, поскольку его результаты не воспроизводят эмпирические данные.

Примечание 1: красная линия на диаграмме Шепарда показывает идеальную пропорциональность между расстояниями. Если точки сконцентрированы в относительной близости к этой красной линии, то это свидетельствует о высокой доле объясняемой дисперсии. Если они образуют более удаленную прямую, эта доля меньше, но важно, что пропорциональность расстояний сохраняется.

Примечание 2: по умолчанию схема Шепарда рассматривает первые две оси PCA. Если интерпретация требует учета большего количества осей, добавьте аргумент `K=NB`, где `NB` – общее количество задаваемых осей.

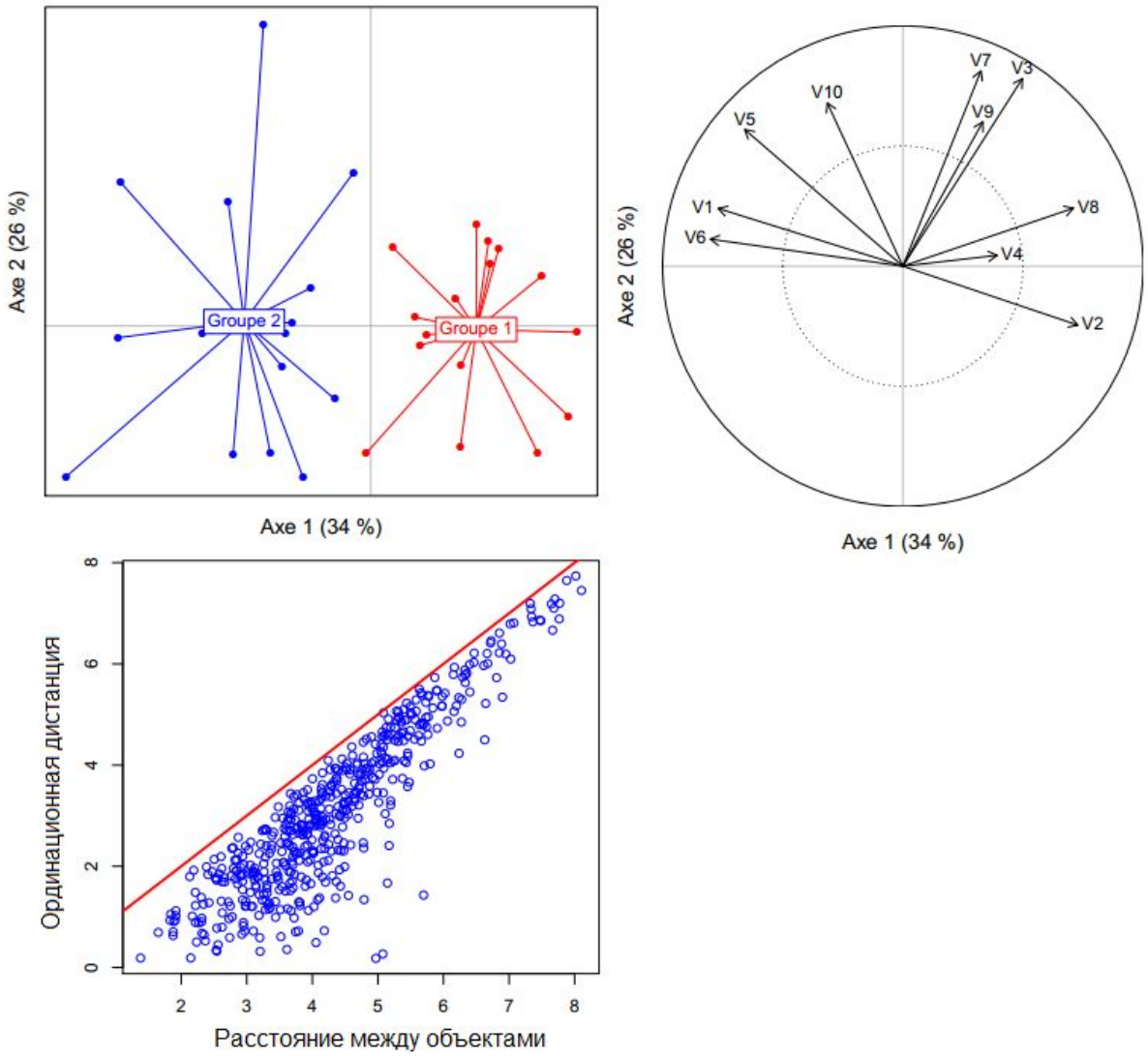
Графические представления

В PCA распространены два возможных графика: диаграмма ординации объектов, которая показывает их распределение по факториальной плоскости, состоящей из двух главных осей, и корреляционный круг (см. п. **89**), который позволяет интерпретировать это распределение.

График распределения объектов. Чтобы построить ординационную диаграмму, можно использовать функцию : `MVA.plot(PCA)`*. По умолчанию представлены оси 1-й (по горизонтали) и 2-й (по вертикали) главных компонент. Они могут быть изменены с помощью аргументов `xax` и `yax`.

Чтобы добавить на график разделение объектов на группы, используйте аргумент `fac=фактор`, где `фактор` определяет принадлежность каждого объекта к априори выделенным группам. Аргументы `col`, `pch`, `fac.lab`, `contours`, `stars` и `barycenters` позволяют настроить эстетику изображения. См `?MVA.scoreplot` для многих других графических опций.

Корреляционный круг. Строится функцией: `MVA.plot(PCA,"corr")`*. Чтобы удалить стрелки, добавьте аргумент `arrows=FALSE`. См. `?MVA.corplot` для многих других графических опций.



Интерпретация

График отдельных объектов позволяет определить, существует ли их структурирование в наборе данных (по группам, вдоль экологического градиента и т.д.). Если ставится задача группировки, их можно объективно выделить с помощью методов кластеризации (см. п. **92-95**). Кроме того, можно проверить, насколько значимо "коррелирует" структура, созданная PCA, с другими внешними факторами и/или ковариатами (см. п. **91**).

Во-вторых, круг корреляций позволяет определить переменные исходной таблицы, которые объясняют наблюдаемое структурирование. Для этого мы на графике распределения объектов определяем, какие направления могут иметь отношение к биологической интерпретации (это могут быть оси главных компонент или любые диагонали), и выделяем по корреляционному кругу переменные, которые наиболее коррелируют с этими направлениями (см. п. **89**).

97. Факторный анализ соответствий (CA)

Англ. – *Correspondence analysis (CA)*

Подготовка данных

Исходный набор данных представляет собой таблицу сопряженности (т. е. нулевые или положительные целые значения), либо массив присутствия-отсутствия (т. е. 0/1). В анализе соответствий строки и столбцы таблицы рассматриваются симметрично, т. е. нет принципиальных отличий между "объектами" и "переменными". Таким образом, строки и столбцы могут быть переставлены местами без изменения существа анализа.

Проведение анализа

Для выполнения анализа можно использовать функцию из пакета `vegan`:
`CA <- cca(таблица)` , где `таблица` является массивом исходных данных.

Оценка качества анализа

Цель анализа соответствий состоит в том, чтобы наилучшим способом обобщить исходную информацию, которая представляет собой соответствие между строками и столбцами набора данных. Меру этого соответствия будем называть *инерцией* (инерция на самом деле является более общим термином, и тот факт, что она олицетворяет собой уровень совпадения, является частным случаем). Поэтому для оценки качества анализа необходимо учитывать долю инерции, объясняемую каждой осью. Чтобы получить эти доли, используйте, например, функцию: `MVA.synt(CA)*`.

Примечание 1: доля объясненной инерции всегда располагается в порядке убывания (т.е. ось 1 объясняет больше инерции, чем ось 2, которая объясняет больше, чем ось 3 и т.д.).

Примечание 2: нет абсолютного правила, сколько осей следует использовать для интерпретации. Это всегда компромисс между хорошим обобщением информации, содержащейся в исходных данных (которое увеличивается с количеством осей) и легкостью для интерпретации (которая уменьшается с количеством осей).

В некоторых случаях доля инерции, объясняемая CA, может быть относительно низким. Это не обязательно означает, что анализ бесполезен. Более важно то обстоятельство, что расстояния между объектами в многомерном пространстве, образуемом осями соответствия, хорошо соответствуют фактическим расстояниям между объектами в исходной таблице данных. Чтобы проверить это соответствие, можно проанализировать диаграмму Шепарда, которую легко сформировать функцией из пакета `vegan`: `stressplot(CA)`. Если точки на этой диаграмме располагаются

примерно на одной прямой, то расстояния в пространстве CA пропорциональны фактическим расстояниям, и для интерпретации можно опираться на результаты этого анализа. Если же точки явно далеки от прямой линии, то расстояния искажены, а интерпретация анализа сомнительна, поскольку его результаты не воспроизводят эмпирические данные.

Примечание 1: красная линия на диаграмме Шепарда показывает идеальную пропорциональность между расстояниями. Если точки сконцентрированы относительно прямой, близкой к этой красной линии, то это свидетельствует о высокой доле объясняемой инерции. Если они образуют более удаленное вытянутое облако, эта доля меньше, но важно, что пропорциональность расстояний сохраняется.

Примечание 2: по умолчанию схема Шепарда рассматривает первые две оси CA. Если интерпретация требует учета большего количества осей, добавьте аргумент `K=NB`, где `NB` – общее количество задаваемых осей.

Графическое представление

В факторном CA каждая строка или столбец представлены точками на одной диаграмме, которую можно назвать "графиком ассоциации". Однако невозможно представить одновременно на одном и том же графике расстояния как между объектами по строкам и по столбцам без некоторого их смещения. Таким образом, следует выбирать между представлением без смещения по строкам (масштаб типа 1) или столбцам (масштаб типа 2).

Для представления графика ассоциаций без смещения расстояний между строками используется функция: `MVA.plot(CCA, points=FALSE, scaling=1)*`. Аналогичный график без смещения расстояний между столбцами формируется командой: `MVA.plot(CCA, points=FALSE, scaling=2)*`. По умолчанию представлены оси 1-й (по горизонтали) и 2-й (по вертикали) главных осей. Это обстоятельство может быть изменено с помощью аргументов `xax` и `yax`.

Чтобы добавить на график разделение объектов на группы, необходимо выполнить следующую процедуру:

```
> график <- MVA.plot(CCA, points=FALSE, col=colors)*,
```

где `colors` – вектор цветов с двумя значениями, первый для столбцов, а второй для строк. Если вы хотите отобразить группировку по столбцам, первый цвет должен быть `"white"`;. для группировки по строкам второй цвет должен быть `"white"`;.

```
> par(new=TRUE)
> MVA.plot(CCA, points =FALSE, xlim= график$xlim,
                    ylim= график$ylim, set=nb, fac=фактор)*,
```

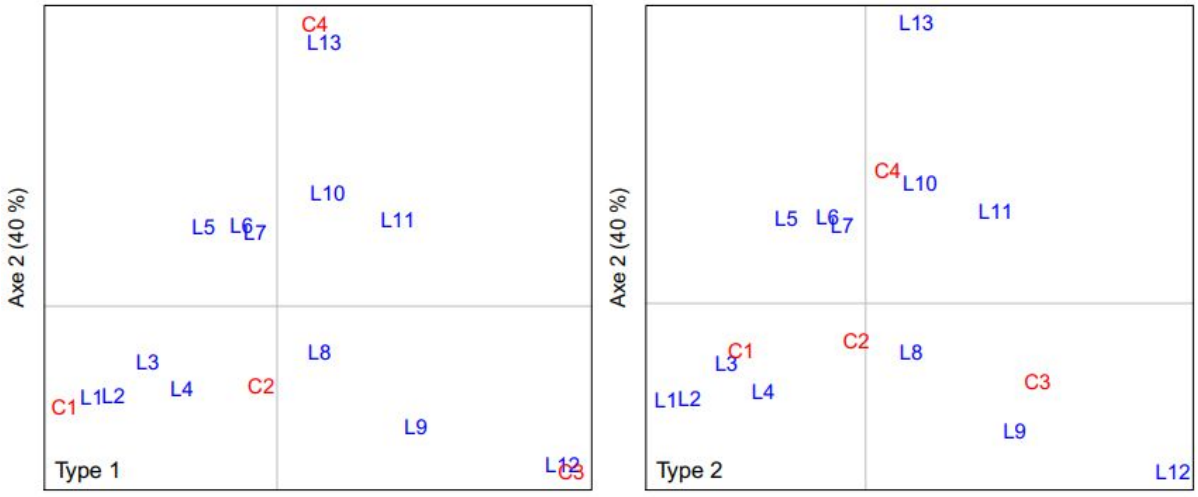
где `nb` равно 1 для группировки строк или 2 для группировки столбцов, а `фактор` – фактор, определяющий группу каждой строки / столбца. Аргументы `col`, `pch`, `fac.lab`, `contours`, `stars` и `barycenters` позволяют настроить эстетику изображения. См `?MVA.scoreplot` для многих других графических опций.

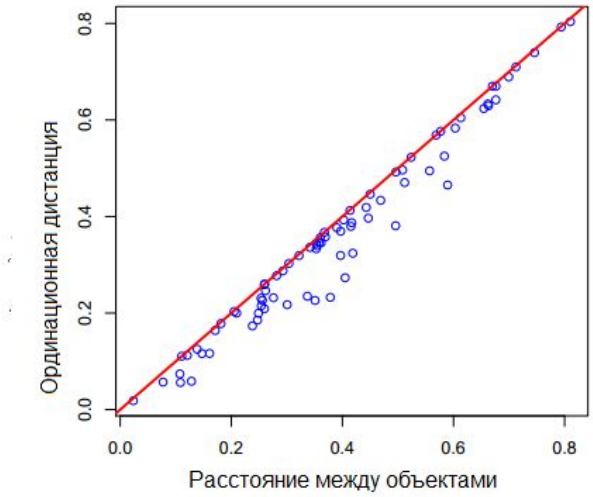
Интерпретация

Идея анализа соответствий относительно проста и основана на трех принципах :

- чем ближе точки, представляющие две строки, то тем больше эти строки соответствуют друг другу;
- то же самое для столбцов;
- близость между точками, представляющими строки и столбцы, указывает на связь между этими строками и столбцами.

Вообще говоря, чем больше точек на графике удалено от начала координат, тем сильнее ассоциация между строками и столбцами. Степень этой ассоциации непосредственно проверяется с помощью теста χ^2 на независимость (см. п. **86**).





98. Множественный анализ соответствий (MCA)

Англ. – *Multiple Correspondence analysis (MCA)*

Подготовка данных

Множественный анализ соответствий чувствителен к слабым эффектам (т.е. к градациям, представленным небольшими численностям объектов). В тех случаях, когда уровней факторов слишком много, лучше сгруппировать столбцы или строки с относительно малыми суммами для увеличения численности объектов в клетках.

Проведение анализа

Для выполнения анализа можно использовать функцию из пакета `ade4`:
`MCA <- dudi.acm(таблица, scannf=FALSE, nf=10)`, где `таблица` является массивом исходных данных.

Оценка качества анализа

Цель множественного анализа соответствий состоит в том, чтобы наилучшим способом обобщить исходную информацию, содержащуюся в наборе данных, которая называется *инерцией*. Поэтому для оценки качества анализа необходимо учитывать долю инерции, объясняемую каждой осью. Чтобы получить эти доли, используйте,

например, функцию: `MVA.synt(MCA)*`.

Примечание 1: доля инерции всегда располагается в порядке убывания (т. е. ось 1 объясняет больше инерции, чем ось 2, которая объясняет больше, чем ось 3 и т.д.).

Примечание 2: нет абсолютного правила, сколько осей следует использовать для интерпретации. Это всегда компромисс между хорошим обобщением информации, содержащейся в исходных данных (которое увеличивается с количеством осей) и легкостью для интерпретации (которая уменьшается с количеством осей).

Графические представления

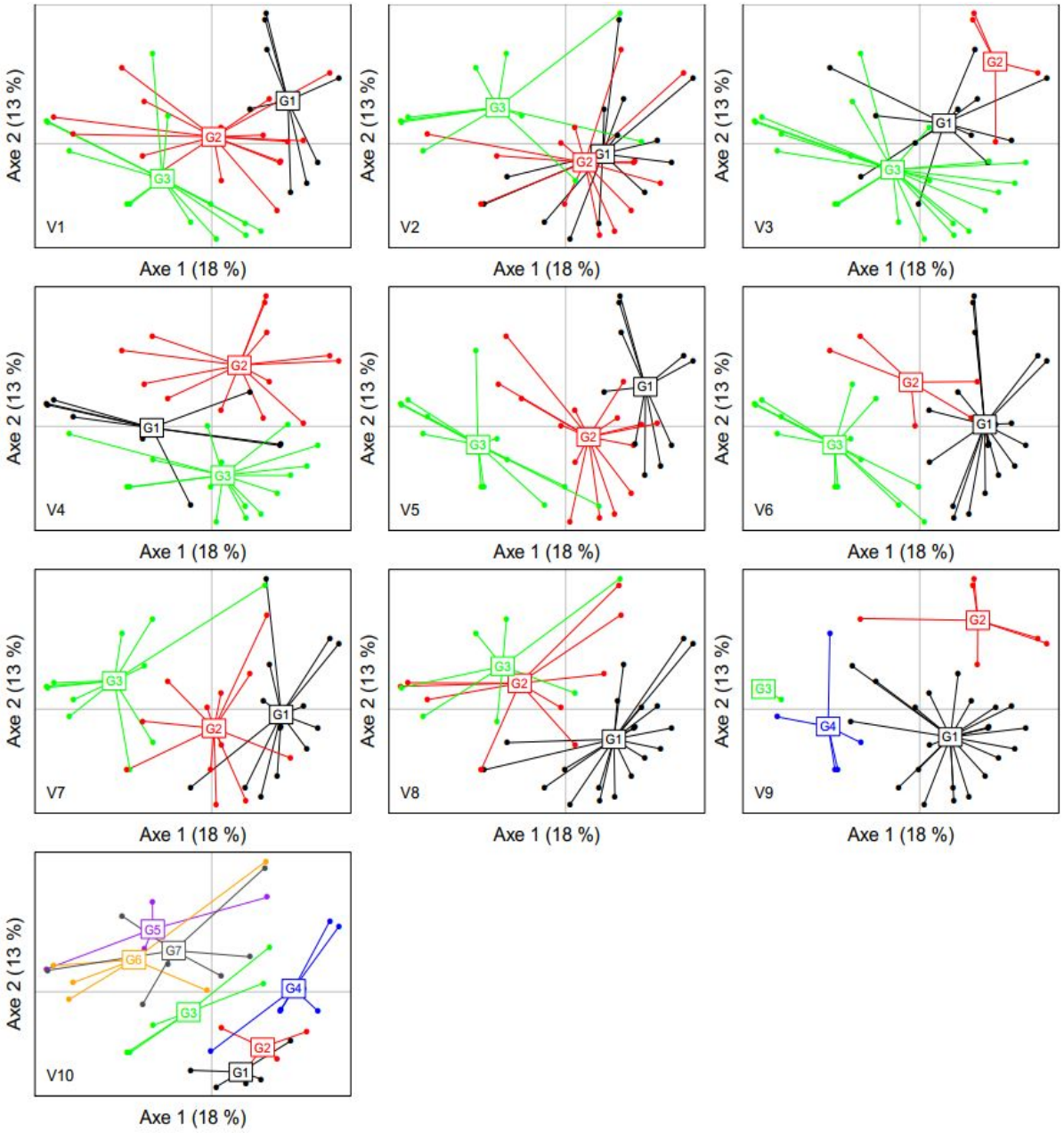
Единственным графическим представлением является диаграмма ассоциации объектов, которая показывает их распределение по факториальной плоскости, состоящей из двух осей. Чтобы построить график, можно использовать функцию: `MVA.plot(MCA)*`. По умолчанию представляются оси 1 (по горизонтали) и 2 (по вертикали), но они могут быть изменены с помощью аргументов `xax` и `yax`.

По умолчанию диаграмма ординации объектов рисуется столько раз, сколько есть переменных в наборе данных, и на каждом из графиков представлены уровни одной переменной. Чтобы представить только одну интересующую переменную (обязательно фактор), добавьте аргументы `byfac=FALSE` и `fac=переменная`. Аргументы `col`, `pch`, `fac.lab`, `contours`, `stars` и `barycenters` позволяют настроить эстетику изображения.

Интерпретация

Идея анализа соответствий относительно проста: чем ближе свойства двух разных переменных, тем больше они соответствуют друг другу в наборе данных. Интерпретация требует мысленно накладывать друг на друга различные графики, что серьезно усложняет ситуацию, если увеличивается количество переменных. Поэтому лучше ограничить их число.

Если вы хотите биологически интерпретировать оси, полученные при анализе (например, является ли эта ось структурным направлением облака точек), вы должны сосредоточиться на переменных, имеющих наибольший вес при построении этой оси. В MCA это измеряется соотношением коэффициентов корреляции, которое варьируется от 0 (нулевой вес) до 1 (максимальный вес). Чтобы получить отчеты, содержащие статистические оценки осей, используйте функцию: `scat.cr(ACM, axis=NB)*`, где `NB` – номер выбранной оси.


Axe 2 (13 %)
Axe 1 (18 %)
V1
G1
G2
G3
V2
G1
G2
G3
V3
G1
G2
G3
V4
G1
G2
G3
V5
G1
G2
G3
V6
G1
G2
G3
V7
G1
G2
G3
V8
G1
G2
G3
V9
G1
G2
G3
G4
V10
G1
G2
G3
G4
G5
G6
G7

99. Смешанный анализ

Англ. – *Mix analysis*

Подготовка данных

Как и PCA (см. п. **96**), смешанный анализ работает лучше, если количественные переменные набора данных имеют, по крайней мере, приблизительно симметричное распределение. Предварительное преобразование этих переменных может в значительной степени улучшить ситуацию (см. п. **88**).

Как и MCA (см. п. **98**), смешанный анализ чувствителен к низкой численности ячеек (т. е. к уровням качественных переменных, для которых число представленных объектов сравнительно мало). В тех случаях, когда уровней слишком много, лучше сгруппировать строки или столбцы для увеличения выраженности эффекта.

Проведение анализа

Для выполнения смешанного анализа используют функцию из пакета `ade4`: `Amix <-dudi.mix (таблица, scannf=FALSE, nf=10)` с указанием таблицы данных. Количественные переменные автоматически стандартизируются (см. п. **88**).

Оценка качества анализа

Цель смешанного анализа состоит в том, чтобы наилучшим способом обобщить исходную информацию, содержащуюся в наборе данных, которая называется *инерцией*. Поэтому для оценки качества анализа необходимо учитывать долю инерции, объясняемую каждой осью. Чтобы получить эти доли, используйте, например, функцию: `MVA.synt(AMix)*`.

Примечание 1: доля инерции всегда располагается в порядке убывания (т. е. ось 1 объясняет больше инерции, чем ось 2, которая объясняет больше, чем ось 3...).

Примечание 2: нет абсолютного правила, сколько осей следует использовать для интерпретации. Это всегда компромисс между хорошим обобщением информации, содержащейся в исходных данных (которое увеличивается с количеством осей) и легкостью для интерпретации (которая уменьшается с количеством осей).

Графические представления

Если в наборе данных имеется хотя бы одна количественная переменная, возможны два представления: диаграмма ординации объектов, которая показывает их распределение по факториальной плоскости, состоящей из двух главных осей, и корреляционный круг (см. п. **89**), который позволяет интерпретировать это распределение. Если нет количественной переменной, единственным возможным графиком является тот, который связан с объектами.

График распределения объектов. Чтобы построить ординационную диаграмму, можно использовать функцию : `MVA.plot(AMix)*`. По умолчанию представлены оси 1-й (по горизонтали) и 2-й (по вертикали). Они могут быть изменены с помощью аргументов `xax` и `yax`.

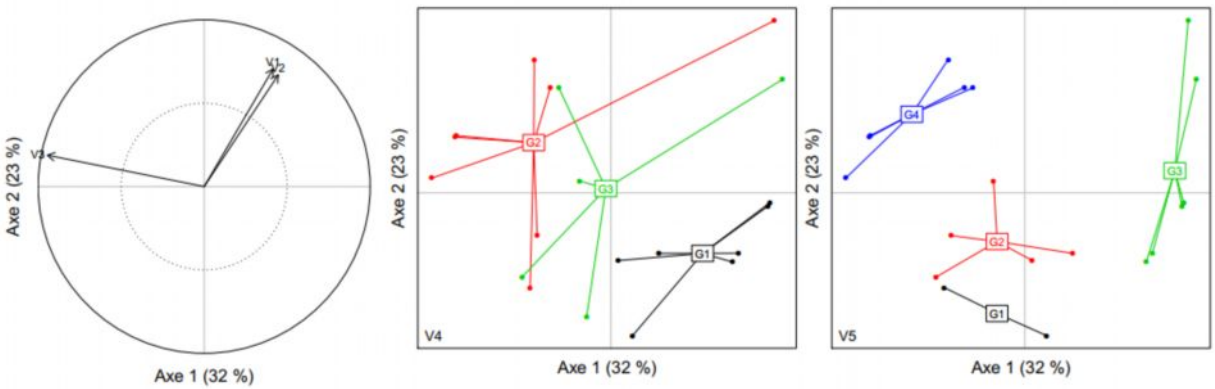
Чтобы добавить на график разделение объектов на группы, используйте аргумент `fac=фактор`, где `фактор` определяет принадлежность каждого объекта к группе. Аргументы `col`, `pch`, `fac.lab`, `contours`, `stars` и `barycenters` позволяют настроить эстетику изображения. См `?MVA.scoreplot` для многих других графических опций.

Корреляционный круг. Строится функцией: `MVA.plot(AMix,"corr")*`. Чтобы удалить стрелки, добавьте аргумент `arrows=FALSE`. См. `?MVA.corplot` для многих других графических опций.

Интерпретация

Смешанный анализ является своего рода промежуточным звеном между анализом главных компонент PCA (которому он эквивалентен, если есть только количественные переменные) и множественным анализом соответствий MCA (которому эквивалентен, если есть только номинальные неупорядоченные качественные переменные). Таким образом, интерпретация выполняется как на основе PCA для количественных переменных (см. п. **96**), так и MCA для качественных переменных (см. п. **98**). Взаимосвязь между количественными и качественными переменными истолковывается просто: чем ближе центр тяжести группы расположен к концу стрелки (при проецировании перпендикуляра к этой стрелке), тем выше среднее значение количественной переменной, представленной стрелкой, для объектов этой группы, (и наоборот, см. п. **89**).

Если вы хотите биологически интерпретировать оси, полученные при анализе (например, является ли эта ось структурным направлением облака точек), вы должны сосредоточиться на переменных, имеющих наибольший вес при построении этой оси. В смешанном анализе это измеряется соотношением коэффициентов корреляции, которое варьируется от 0 (нулевой вес) до 1 (максимальный вес). Чтобы получить отчеты, содержащие статистические оценки осей, используйте функцию:
`scat.cr(ACM, axis=NB)*`, где `NB` – номер выбранной оси.



Ординация на основе матрицы расстояний

Анализ главных координат (PCoA) и неметрическое многомерное шкалирование (nMDS) имеют одну и ту же цель : создать ординацию на основе матрицы расстояния. Однако в деталях они различаются двумя обстоятельствами :

1. PCoA сохраняет фактические расстояния между объектами, в то время как nMDS использует полуколичественную информацию (ранги). Другими словами, расстояния, наблюдаемые на диаграмме, сформированной PCoA, могут быть непосредственно интерпретированы как расстояния, представленные в матрице дистанций, тогда как для ординации, выполненной nMDS, единственная доступная информация имеет тип "это расстояние больше, чем иное", поскольку фактические расстояния не сохраняются.

2. Как и другие методы ординации, PCoA формирует набор осей, равный числу переменных, сохраняет несколько главных из них для интерпретации, в то время как остальная часть информации, связанная с неучтенными осями, теряется. Метод nMDS производит ординацию на основе априори заданного числа осей (на практике 2 или 3), поэтому интерпретируется целиком вся имеющаяся информация.

Если вы хотите сохранить количественную информацию о фактических расстояниях, принимая во внимание, однако, что вы имеете дело только с ее частью, то выберите PCoA. Если вы хотите обобщить всю информацию для интерпретации, теряя, однако, количественную информацию о фактических расстояниях, то выберите nMDS.

Примечание: если планируется "сопоставить" результаты ординации с внешними переменными (см. п. **91**), то здесь корректно использовать только PCoA.

100. Матрицы дистанций

Матрица дистанций – это матрица, в которой каждая пара объектов характеризуется расстоянием между ними. Таким образом, это (а) квадратная матрица (т. е. столько строк и столбцов, сколько отдельных объектов), (б) симметричная (поскольку расстояние между A и B такое же, как между B и A) и (в) диагональ матрицы, состоящая из 0, отделяет два симметричных треугольника (один нижний и другой верхний) . Для отображения и хранения часто используется только нижний треугольная часть матрицы.

ПРИМЕР.

Матрица расстояний (здесь между тремя объектами) может быть представлена полностью или одной из ее треугольных частей (обычно, нижней), которая содержит в себе всю информацию :

```
      A    B    C                A     B
A  0.00 2.17 2.10           B  2.17
B  2.17 0.00 0.29           C  2.10  0.29
C  2.10 0.29 0.00
```

В некоторых предметных областях непосредственно работают с матрицами расстояний (географические расстояния, расстояния, основанные на генетических последовательностях нуклеотидов или аминокислот и т.д.). В других областях использование матриц расстояния позволяет обобщить информацию, содержащуюся в нескольких переменных. Однако в последнем случае вклад каждой переменной в расстояния между объектами теряется.

Если набор данных представляет собой массив переменных, то можно выбрать два типа анализа (по исходным переменным или по матрице расстояния) и этот вопрос относительно просто решается. Таблица исходных переменных используется, если

ставится задача количественной оценки сравнительной значимости отдельных переменных и на этом основана биологическая интерпретация. Матрица расстояния используется, если задача ограничивается оценкой общего сходства отдельных объектов, пренебрегая при этом ролью описывающих их переменных.

Расстояние и сходство – это полностью связанные (но противоположные) меры, т.е. чем выше расстояние (или диссимиляция) между двумя объектами, тем ниже их сходство. Таким образом, многие расстояния вычисляются по индексу сходства простым отношением: `Расстояние = 1 - Сходство`.

Матрица расстояний называется евклидовой, если она представлена в евклидовом (т. е. традиционном геометрическом многомерном пространстве). Иногда важно знать, является ли матрица расстояний евклидовой или нет, поскольку ряд методов анализа основывается на этом предположении. Независимо от типа матрицы расстояния, можно просто проверить, является ли она евклидовой, используя функцию из пакета `ade4`: `is.euclid(мат.раст)`. Чтобы сделать матрицу расстояния евклидовой, простым и практически всегда эффективным решением является извлечение из нее квадратного корня: `мат.раст2<-sqrt(мат.раст)`.

Выбор формулы для расчета расстояния зависит, прежде всего, от типа имеющихся данных, а также от их предметной области. Поэтому мы не будем здесь останавливаться, чтобы сравнивать их все. Функции для расчета наиболее распространенных расстояний представлены в зависимости от типа данных.

Двоичные данные (0/1)

Функция `dist.binary()` из пакета `ade4` предлагает десять расстояний, основанных на индексах сходства, в том числе наболее традиционных (индексы Жаккара, Сокаля-Мишнера или Съеренсена-Дайса). Все они пронумерованы от 1 до 10 (см. `?dist.binary` для получения полного списка). Вычисление матрицы расстояния осуществляется командой: `dist.binary(таблица, номер)`, где задаются `таблица` данных и `номер` выбранного расстояния. Все матрицы, возвращаемые этой функцией, являются евклидовыми.

Важным моментом при работе с двоичными данными является вопрос о двойном нуле, т.е. одновременном отсутствии признака, зарегистрированного у двух объектов. Это обстоятельство иногда может иметь смысл, но чаще всего его нет. Индексы, учитывающие двойное отсутствие в качестве доказательства сходства (например, индекс Сокаля-Мишнера) называются симметричными, а те, кто делает обратное предположение (например, индексы Жаккара и Съеренсена-Дайса), называются асимметричными.

Таблицы пропорций

Сюда входят численности при пересечении уровней двух факторов или процентные данные. В обоих случаях значения могут быть только положительными или нулевыми. Значения одного и того же объекта (т. е. в одной строке таблицы данных) интерпретируются как проценты от общего количества строк.

Функция `dist.prop()` из пакета `ade4` предлагает пять расстояний, которые пронумерованы от 1 до 5 (см. `? dist.prop` для полного списка). Вычисление матрицы расстояния: `dist.prop(таблица, номер)`. Только метод № 3 формирует евклидовы матрицы расстояния.

Расстояние, часто используемое, но не предлагаемое `dist.prop()` – это расстояние Брея-Кёртиса. Чтобы вычислить его, используют функцию из пакета `vegan`: `vegdist(таблица,"bray")`. Возвращаемая матрица не является евклидовой.

Количественные данные

Функция `dist()` позволяет вычислить наиболее распространенные расстояния (евклидово, манхэттенское и т.д.). Чтобы ее использовать, подают команду: `dist (таблица, "метод")`, где `метод` – название формулы для подсчета расстояния в кавычках (см. `?dist` для полного списка). Другие формулы измерения расстояния предлагаются функцией из пакета `ade4`: `dist.quant()` , которая работает с порядковыми номерами методов (см. `? dist.quant` для полного списка).

Альтернативой вычислению расстояний в зависимости от значений переменных (что делают предыдущие функции) является использование корреляции между отдельными объектами. Для вычисления таких расстояний можно использовать функцию из пакета `factoextra`: `get_dist(таблица, method="pearson")`.

Примечание: для количественных данных может быть полезным (и даже рекомендуется, если вы хотите выполнить группировку) стандартизировать таблицу данных перед вычислением расстояний, чтобы придать одинаковый вес всем переменным (см. п. **88**).

Смешанные данные (переменные нескольких типов)

Используемое расстояние – это расстояние Гувера, которое позволяет обрабатывать как количественные, так и двоичные (закодированные 0/1), порядковые и номинальные переменные. Матрица вычисляется с использованием функции `daisy(таблица)` из пакета `cluster`. Номинальные и порядковые качественные переменные (т.е. неупорядоченные и упорядоченные факторы, см. п. **52** для их создания) свободно распознаются как таковые. Для двоичных переменных лучше явно указать, следует ли их рассматривать как симметричные или нет. В общем случае можно добавить аргумент `type= list(symm=var.sym,asymm=var.asym)`, где `var.sym` – вектор с именами (или номерами) столбцов таблицы, которые следует рассматривать как симметричные, и `var.asym` – то же самое для переменных, которые следует учитывать асимметрично. Возвращаемая матрица не является евклидовой.

Генетические данные

Генетический анализ данных – это мир сам по себе. Многие методы оценки генетического расстояния предлагаются пакетом `adegenet`, которые работают с данными, отформатированными определенным образом. См. `?adegenet` (после загрузки пакета) для подробных объяснений.

В конце концов, можно импортировать в R любую матрицу расстояния, которая будет считаться массивом. Необходимо преобразовать этот массив в объект матрицы расстояний, чтобы он мог использоваться как таковой. Для этого нужно подать команду: `мат.раст <- as.dist(таблица)`.

101. Анализ главных координат (PCoA)

Англ. – *Principal coordinate analysis (PCoA) = Metric multidimensional scaling (MDS)*

Подготовка данных

Необходимо предварительно проверить, является ли матрица расстояний евклидовой или нет (см. п. **100**).

Проведение анализа

Для осуществления анализа главных координат можно, например, использовать функцию из пакета `vegan` : `PCoA<-dbrda(мат.расст~1)`, где `мат.расст` является матрицей дистанций. Если эта матрица не евклидова, необходимо применить исправление. Для этого добавьте аргумент `add=TRUE`.

Оценка качества анализа

Цель анализа главных координат PCoA состоит в том, чтобы сформировать некоторую информационную структуру, которая наилучшим образом "объясняет" общую дисперсию матрицы расстояний. Таким образом, для оценки качества анализа необходимо оценить долю общей дисперсии, связанную с каждой осью. Чтобы получить эти доли, используйте, например, функцию : `MVA.synt(PCoA)*`.

Примечание 1: доли дисперсии всегда рассматриваются в порядке убывания (т. е. ось 1 объясняет больше дисперсии, чем ось 2, которая объясняет больше, чем ось 3...).

Примечание 2: нет абсолютного правила, сколько осей следует использовать для интерпретации. Это всегда компромисс между хорошим обобщением информации, содержащейся в матрице дистанций (которое увеличивается с количеством осей) и легкостью для интерпретации (которая уменьшается с количеством осей).

В некоторых случаях доля дисперсии, объясняемый PCoA, может быть относительно низкой. Это не обязательно означает, что анализ бесполезен. Более значимо то обстоятельство, что расстояния между объектами в многомерном пространстве, образуемом главными координатами, хорошо соответствуют фактическим расстояниям между объектами в матрице расстояний. Чтобы проверить это соответствие, можно проанализировать диаграмму Шепарда, которую легко сформировать функцией из пакета `vegan`: `stressplot(PCoA)`. Если точки на этой диаграмме располагаются примерно на одной прямой, то расстояния в пространстве PCoA пропорциональны исходным расстояниям, и для интерпретации можно опираться на результаты этого анализа. Если же точки явно далеки от прямой линии, то расстояния искажены, а интерпретация анализа сомнительна, поскольку его результаты не воспроизводят эмпирические данные.

Примечание 1: красная линия на диаграмме Шепарда показывает идеальную пропорциональность между расстояниями. Если точки сконцентрированы относительно прямой, близкой к этой красной линии, то это свидетельствует о высокой доле объясняемой дисперсии. Если они образуют более удаленную прямую, то эта доля меньше, но важно, что пропорциональность расстояний сохраняется.

Примечание 2: по умолчанию схема Шепарда рассматривает первые две оси PCoA. Если интерпретация требует учета большего количества осей, добавьте аргумент `K=NB`, где `NB` – общее количество задаваемых осей.

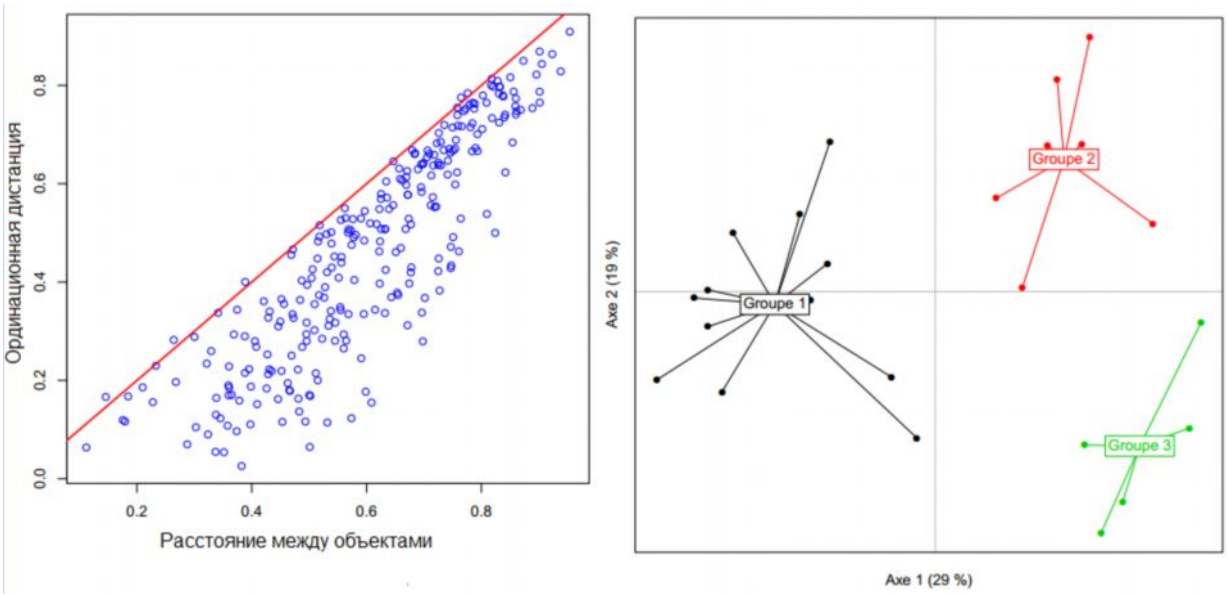
Графические представления

Единственным графическим представлением является диаграмма ординации объектов, которая показывает их распределение по факториальной плоскости, состоящей из двух осей. Чтобы построить график, можно использовать функцию : `MVA.plot(PCoA)*`. По умолчанию представляются оси 1 (по горизонтали) и 2 (по вертикали), но они могут быть изменены с помощью аргументов `xax` и `yax`.

Чтобы добавить на график разделение объектов на группы, используйте аргумент `fac=фактор`, где `фактор` определяет принадлежность каждого объекта к группе. Аргументы `col`, `pch`, `fac.lab`, `contours`, `stars` и `barycenters` позволяют настроить эстетику изображения. См `?MVA.scoreplot` для многих других графических опций.

Интерпретация

График отдельных объектов позволяет определить, существует ли их структурирование в наборе данных (по группам, вдоль экологического градиента или иное). Если ставится задача группировки, их можно объективно выделить с помощью методов кластеризации (см. п. **92-95**). Кроме того, можно проверить, насколько значимо "коррелирует" структура, созданная PCoA, с другими внешними факторами и/или ковариатами (см. п. **91**).



102. Неметрическое многомерное шкалирование (nMDS)

Англ. – *Non metric multidimensional scaling (nMDS)*

Проведение анализа

Для осуществления неметрического многомерного шкалирования nMDS рекомендуется использовать функцию из пакета `vegan` :
`nMDS <- metaMDS(мат.раст, k=nb.axes, trace=FALSE)`, где `мат.раст` является матрицей дистанций, `nb.axes` – количество осей, которые необходимо построить. Аргумент `trace=FALSE` подавляет вывод информации на промежуточных шагах вычисления.

Замечание: nMDS – это анализ, в котором результат отличается в зависимости от числа априори заданных осей.

Оценка качества анализа

Качество nMDS оценивается с помощью индикатора под названием *стресс*, который варьируется от 0 до 1 (иногда от 0 до 100 %). Его значение можно получить с помощью функции `MVA.synt(nMDS)*`. Эмпирически считается, что при стрессе менее 0.05 выполнено отличное приближение расстояний в сформированном пространстве к исходным расстояниям, от 0.05 до 0.1 – хорошее, от 0.1 до 0.2 –

корректное, а при стрессе более 0.2 искажение расстояний велико.

Еще один способ оценить качество nMDS – это проверить, насколько хорошо расстояния между объектами в многомерном пространстве, образуемом неметрическими шкалами, соответствуют фактическим расстояниям между объектами в матрице расстояний. Здесь, правда, речь идет скорее о порядке этих расстояний, поскольку nMDS использует не фактические расстояния, а только полуколичественные оценки типа «это расстояние больше, чем другое». Для этого формируется диаграмма Шепарда функцией из пакета `vegan`: `stressplot(nMDS)`. На этой диаграмме, если точки остаются вблизи красной ступенчатой кривой, то порядок расстояний хорошо воспроизводится и можно опираться на результаты анализа для интерпретации. Если точки сгруппированы вдали от красной кривой, то порядок расстояний не сохраняется и интерпретация анализа бесполезна, поскольку он не воспроизводит реальность.

Примечание: стресс на самом деле является обобщенным индикатором сгущения точек вокруг кривой диаграммы Шепарда.

Графические представления

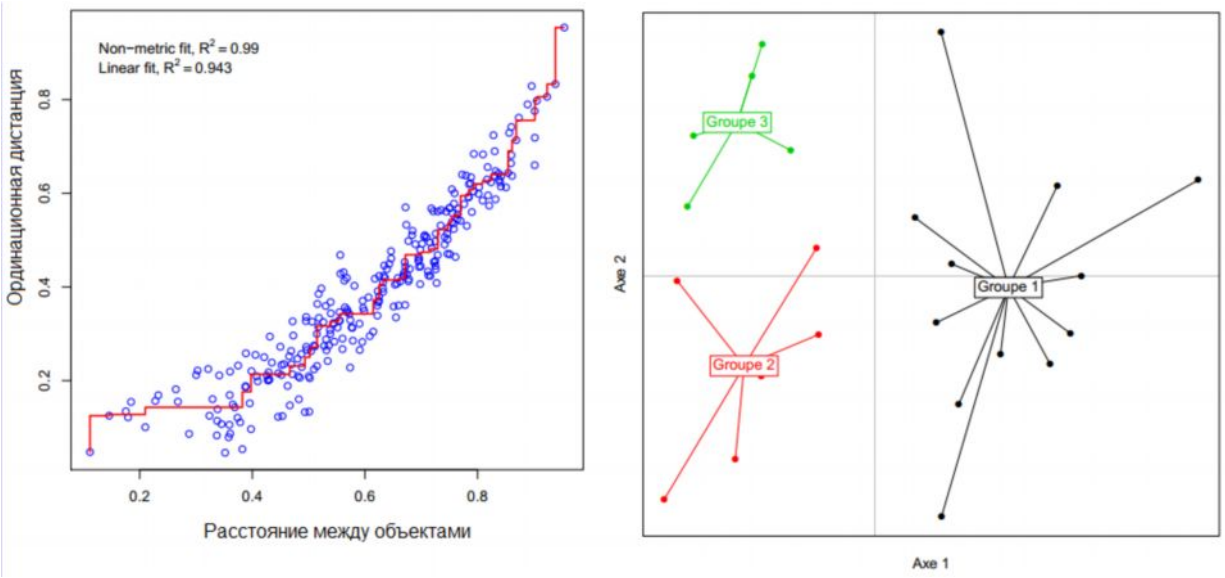
Единственным графическим представлением является диаграмма ординации объектов, которая показывает их распределение по факториальной плоскости, состоящей из двух шкал. Чтобы построить график, можно использовать функцию : `MVA.plot(nMDS)*`. По умолчанию представляются шкалы 1 (по горизонтали) и 2 (по вертикали), но они могут быть изменены с помощью аргументов `xax` и `yax`.

Чтобы добавить на график разделение объектов на группы, используйте аргумент `fac=фактор`, где `фактор` определяет принадлежность каждого объекта к группе. Аргументы `col`, `pch`, `fac.lab`, `contours`, `stars` и `barycenters` позволяют настроить эстетику изображения. См `?MVA.scoreplot` для многих других графических опций.

Интерпретация

График отдельных объектов позволяет определить, существует ли их структурирование в наборе данных (по группам, вдоль экологического градиента или иное). Если ставится задача группировки, их можно объективно выделить с помощью методов кластеризации (см. п. **92-95**).

Внимание: nMDS не сохраняет фактические расстояния между отдельными объектами, а только относительную информацию о них. Поэтому нет смысла "коррелировать" результаты nMDS с внешними переменными.



МНОГОМЕРНЫЙ АНАЛИЗ ДАННЫХ ДВУХ ТАБЛИЦ

Асимметричный анализ таблиц исходных переменных

Выбор конкретного метода анализа зависит от цели и характера переменных в обоих массивах исходных данных: таблице внешних (объясняющих, независимых) переменных **X** и таблице объясняемых переменных (многомерного отклика) **Y**:

Основная цель анализа состоит в том, чтобы выполнить ординацию, т.е. оценить многомерную зависимость $\mathbf{Y} = f(\mathbf{X})$ и ее различные варианты зависят от состава этих таблиц.

1. Таблица внешних переменных **X** состоит из **одной качественной переменной** (т. е. фактора, определяющего группу). В этом случае мы говорим о *дискриминантном анализе*.

   Таблица многомерного отклика **Y** состоит из:

   (а) массива **количественных переменных**. Число этих переменных :

   - гораздо **меньше, чем объектов** (минимум в 5 раз меньше), и они мало коррелированы между собой → используйте *Линейный Дискриминантный Анализ* (LDA);
   - так же много или **больше, чем число объектов**, и/или они коррелируют между собой → используйте *Дискриминантный Анализ* с использованием *регрессии на основе Частных Наименьших Квадратов* (PLS-DA).

   (б) таблицы сопряженности или таблицы присутствия-отсутствия → используйте *Дискриминантый Анализ Соответствия* (DCA)

2. Таблица внешних переменных **X** состоит из нескольких переменных (**количественных** и/или качественных) или одной количественной переменной

   Таблица многомерного отклика **Y** состоит из:

   (а) массива **количественных переменных** → используйте *Анализ Избыточности* (RDA);

   (б) **таблицы сопряженности** или таблицы присутствия-отсутствия → используйте *Канонический Анализ Соответствия* (CCA);

   (в) если массив переменных не вписывается в два предыдущих условия → преобразуйте его в **матрицу расстояния** (см. п. **100**) и используйте *Анализ Избыточности на основе Матрицы Расстояния* (db-RDA).

Другая возможная цель анализа состоит в том, чтобы просто выполнить статистический тест и проанализировать набор количественных переменных. Все переменные в таблице **Y** должны быть количественными. Если это условие не соблюдается, необходимо преобразовать эту таблицу в матрицу дистанций (см. п. **100**) и выполнить тест с использованием метрики расстояния.

103. Анализ избыточности (RDA)

Англ. – *Redundancy analysis* (RDA) = *Principal components of instrumental variables*

Подготовка данных

Анализ избыточности работает корректнее, если переменные имеют приблизительно нормальное (по крайней мере, симметричное) распределение и связаны между собой линейными отношениями. Предварительное преобразование данных может в значительной степени улучшить ситуацию (см. п. **88**).

Также необходимо, чтобы дисперсионно-ковариационные матрицы (многомерный эквивалент дисперсии) были однородными по отношению к различными уровням качественных объясняющих переменных (если таковые имеются). Чтобы проверить это, можно использовать функцию из пакета `vegan` : `anova(betadisper(dist(таб.Y), фактор))`, где `таб.Y` – это исходный массив объясняемых переменных, для которых `фактор` определяет группировку.

Наконец, рекомендуется, как правило, стандартизировать переменные перед анализом (см. п. **88**). Это придает одинаковый вес всем переменным и формирует результаты с применением коэффициента корреляции, что часто проще. В этом разделе будет предполагаться, что переменные стандартизированы.

Проведение анализа

Для осуществления анализа можно, например, использовать функцию из пакета `vegan` : `RDA <- rda (формула, data=таб.X)` , где `таб.X` является таблицей, содержащей объясняющие переменные. Аргумент `формула` (см. п. **40**) в левой части в качестве отклика должен содержать таблицу `таб.Y` , включающую объясняемые переменные. Если вы хотите стандартизировать эти переменные, но не сделали этого раньше, добавьте аргумент `scale=TRUE`. По умолчанию переменные не стандартизируются.

Общая объясняющая способность

Анализ избыточности RDA состоит из двух этапов :

1. Разброс данных в объясняемой таблице **Y** разлагается на две составляющие: (а) вариацию, вызываемую внешними переменными (так называемую "ограниченную[1]" или объясняемую вариацию) и (б) остаточную вариацию (т.е. "неограничиваемую" или необъясняемую). RDA проводит вычисления с показателями вариации, которые можно трактовать как многомерную дисперсию.

2. Реализуются два отдельных анализа главных компонент: первый PCA на основе дисперсии от влияния внешних переменных ("ограниченные главные компоненты") и второй PCA – без учета внешних факторов ("неограничиваемые главные компоненты").

Общий объясняющий потенциал анализа RDA можно оценить с помощью доли "ограниченной дисперсии" (т.е. дисперсии в данных, которая объясняется внешними переменными). Чем выше эта доля, тем более полезна сформированная совокупная информационная структура RDA, объединенная с объясняющими переменными. Чтобы получить эти доли, используйте, например, функцию: `MVA.synt(RDA)*`, где результат – в первой возвращаемой таблице.

---

[1] Много раздумий вызвала проблема точного перевода термина **constrained** / **unconstrained**. Предлагается, вероятно, не самый оптимальный вариант. *Прим. пер.*

Оценка качества анализа

Тесты внешних переменных. Значимость эффекта воздействия внешних переменных проверяется с использованием функции `MVA.anova(RDA)*`. При этом осуществляется тест по *F*-критерию с использованием рандомизации. Если хотя бы одна объясняющая переменная оказывает значимое влияние, то можно опираться на результаты RDA для интерпретации. Если ни одна из объясняющих переменных не имеет значимого *p*-значения, то интерпретация результатов такого анализа вряд ли вызовет большой интерес, поскольку эффект влияния внешних переменных не обозначен.

Если влияние внешнего фактора было оценено как значимое, то можно провести множественные сравнения между уровнями фактора (или комбинациями условий взаимодействия факторов). Для выполнения множественных сравнений используйте функцию: `pairwise.factorfit(RDA, Фактор)*`, где `Фактор` является интересуемым внешним фактором.

Обобщенная оценка. Если хотя бы одна объясняющая переменная оказывает значимый эффект воздействия, то интерес представляет блок результатов по "ограниченному" PCA. Как и в случае с классическим PCA (см. п. **96**), качество этого анализа оценивается по доле дисперсии, объясненной каждой осью. Чтобы получить эти доли, используйте, например, функцию : `MVA.synt(RDA)*`, где результат – во второй возвращаемой таблице.

Примечание 1: здесь указывается доля от "ограниченной дисперсии", а не от общей, как это имеет место в классическом PCA.

Примечание 2: доли дисперсии всегда располагаются в порядке убывания (т. е. ось 1 объясняет больше дисперсии, чем ось 2, которая объясняет больше, чем ось 3...).

Примечание 3: нет абсолютного правила, сколько осей следует выделить для интерпретации. Это всегда компромисс между хорошим обобщением информации, содержащейся в исходных данных (которое увеличивается с количеством осей) и легкостью для интерпретации (которая уменьшается с количеством осей).

Графические представления

В RDA строятся два возможных графика: диаграмма ординации объектов, которая показывает их распределение по факториальной плоскости, состоящей из двух главных осей, и корреляционный круг (см. п. **89**), который позволяет интерпретировать это распределение.

График распределения объектов. Чтобы построить ординационную диаграмму, можно использовать функцию: `MVA.plot(RDA)*`. По умолчанию представлены оси 1-й (по горизонтали) и 2-й (по вертикали) главных компонент. Они могут быть изменены с помощью аргументов `xax` и `yax`. Чтобы представить только PCA без учета влияния внешних факторов, добавьте аргумент `space=2`.

Чтобы добавить на график разделение объектов на группы, используйте аргумент `fac=фактор`, где `фактор` определяет принадлежность каждого объекта к группе. Аргументы `col`, `pch`, `fac.lab`, `contours`, `stars` и `barycenters` позволяют настроить эстетику изображения. См `?MVA.scoreplot` для многих других графических опций.

Корреляционный круг. Строится функцией: `MVA.plot(RDA,"corr")*`. Чтобы удалить стрелки, добавьте аргумент `arrows=FALSE`. См. `?MVA.corplot` для многих других графических опций.

Как и в случае с ординационной диаграммой объектов, добавьте аргумент `space=2`, чтобы представить только PCA без учета влияния внешних факторов. По

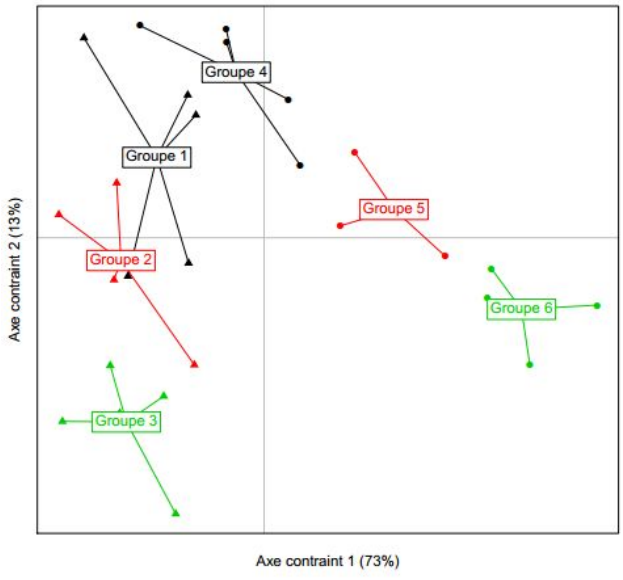
умолчанию представлены стрелки как переменных, подлежащие объяснению, так и количественных объясняющих переменных. Чтобы представить на графике только количественные объясняющие переменные, добавьте аргумент `set=1`, а только для отображения переменных, которые нужно объяснять – `set=2`.

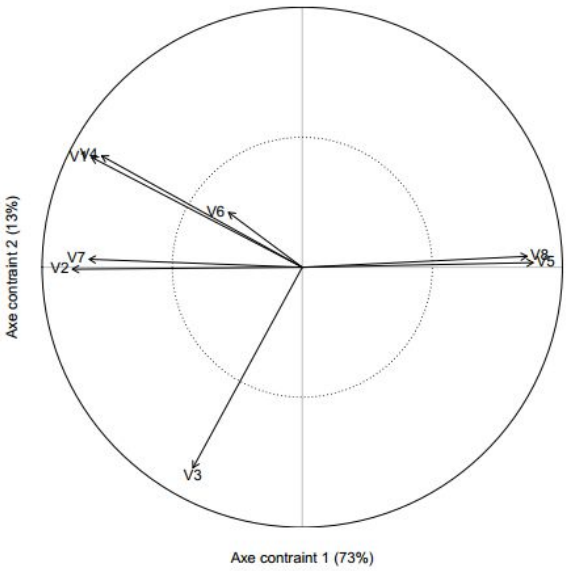
Интерпретация

По нашему мнению, RDA – это единственный способ корректно интерпретировать результаты о связи многомерного отклика с внешними переменными.

График отдельных объектов позволяет определить, существует ли структурирование таблицы **Y**, связанное с влиянием объясняющих переменных из таблицы **X** . На нем можно усмотреть, как точки объектов образуют сгущения для разных уровней внешнего фактора, либо формируют линейные градиенты относительно внешних ковариат.

Круг корреляций позволяет определить переменные, которые влияют на возможную группировку объектов и/или являются причиной линейной упорядоченности точек. Для этого мы на графике распределения объектов определяем, какие направления могут иметь отношение к биологической интерпретации (это могут быть оси главных компонент или любые диагонали), и выделяем по корреляционному кругу переменные, которые наиболее коррелируют с этими направлениями (см. п. **89**). Способы сделать это – те же самые, что и при интерпретации диаграмм PCA (см. п. **96**), за исключением, конечно, того, что переменные таблицы **Y** зависят от переменных таблицы **X**, но не наоборот.





104. Линейный дискриминантный анализ (LDA)

Англ. – *Linear discriminant analysis* (LDA)

Подготовка данных

Количественные переменные должны иметь многомерное нормальное распределение. См. п. **65** для описания проверки этого условия. Однако результаты линейного дискриминантного анализа LDA достаточно устойчивы, если это предположение соблюдается не полностью.

Также необходимо, чтобы дисперсионно-ковариационные матрицы (многомерный эквивалент дисперсии) были однородными по отношению к различными уровням группирующего фактора. Чтобы это проверить, можно использовать функцию из пакета `vegan` : `anova(betadisper(dist(таб.Y), фактор))`, где `таб.Y` – это исходный массив объясняемых данных, для которых `фактор` определяет группировку. Опять же, LDA довольно устойчив к умеренному несоблюдению этого условия.

Если эти два условия вообще не соблюдаются, то значительно улучшить ситуацию может предварительное преобразование данных (см. п. **88**).

Примечание: количественные переменные должны быть стандартизированы, но на практике функции, реализующие LDA, будут делать это автоматически

Проведение анализа

Дискриминантный анализ обычно реализуется с использованием функции из пакета `MASS` `LDA <- lda(таблица, фактор)`.

Оценка качества анализа

Доля ошибок классификации. Одна из функций любого дискриминантного анализа состоит в том, чтобы предсказать группу, к которой принадлежит объект и для которого известны значения количественных переменных. Таким образом, один из способов оценки качества дискриминационного анализа заключается в том, чтобы проверить, насколько безошибочно он позволяет оценивать номер группы неизвестного объекта. Для этого используется метод кросс-проверки, который выполняет следующую процедуру: (а) набор данных случайным образом разбивается на части; (б) последовательно одна из частей выделяется для контроля, а по остальным осуществляется построение дискриминантной модели; (в) выполняется классификация контрольных объектов, не участвовавших в построении модели. Для большей надежности эти действия могут повторяться многократно (каждый раз при случайном разбиении набора данных).

Для выполнения перекрестной проверки может быть использована функция: `MVA.cv(массив, фактор, model="LDA")*`. По умолчанию функция разбивает набор данных на 7 частей (7-*fold cross-validation*), но этот параметр может быть изменен с помощью аргумента `K=NB`, где `NB` – это количество поднаборов данных для разбиения. Функция может также автоматически настроить это число, если по крайней мере одна группа содержит менее 7 объектов. По умолчанию вся процедура повторяется 10 раз (аргумент `repet`), т.е. в конечном итоге формируется 7×10 = 70 подмоделей.

Если вы хотите использовать LDA для целей прогноза, сохраните результат работы функции `MVA.cv()` в соответствующем объекте.

Тестирование. Проверка значимости компонентов дискриминантной модели выполняется с использованием MANOVA (*Multivariate ANalysis of VAriance*), который является расширением ANOVA на многомерные случаи. Условия использования этого

теста такие же, как и условия LDA. См. п. **108** для описания теста.

Если фактор имеет значимый эффект и существует более двух групп, необходимо провести множественные сравнения, чтобы определить, какие уровни группирующего фактора отличаются. См. п. **43** для описания техники этих сравнений.

Примечание 1: доля ошибок классификации и значимость фактора всегда согласованы: если процент ошибок невелик, фактор имеет значимый эффект (и наоборот).

Примечание 2: Если фактор не имеет значимого эффекта, интерпретация результатов LDA на практике не имеет смысла, и любая попытка предсказаний бессмысленна.

Графические представления

В LDA строятся два возможных графика: диаграмма ординации объектов, которая показывает их распределение по факториальной плоскости, состоящей из двух осей дискриминации, и круг корреляций (см. п. **89**), который позволяет интерпретировать особенности распределения объектов.

Примечание: если существует только две группы, то LDA формирует только одну ось. Тогда график отдельных объектов и круг корреляций сводятся к одному измерению.

График распределения объектов. Чтобы построить ординационную диаграмму, можно использовать функцию : `MVA.plot(LDA, fac=фактор)*`. По умолчанию представлены оси 1-й (по горизонтали) и 2-й (по вертикали) главных компонент. Они могут быть изменены с помощью аргументов `xax` и `yax`.

В случае двумерного графика аргументы `col`, `pch`, `fac.lab`, `contours`, `stars` и `barycenters` позволяют настроить эстетику изображения. См `?MVA.scoreplot` для многих других графических опций.

Корреляционный круг. Строится функцией: `MVA.plot(LDA,"corr")*`. Чтобы удалить стрелки, добавьте аргумент `arrows=FALSE`. См. `?MVA.corplot` для многих других графических опций.

Интерпретация

График отдельных объектов позволяет оценить, как структурируются группы, т. е. насколько хорошо они разделяются или тесно перекрываются друг с другом.

Круг корреляции позволяет выделить переменные, под влиянием которых эти группы различаются. Для этого на графике отдельных объектов выявляются направления, которые имеют отношение к биологической интерпретации (это могут быть оси или любые диагонали), и определяются переменные, которые наиболее тесно коррелируют с этими направлениями на корреляционном круге (см. п. **89**).

Прогноз на основе модели

Роль дискриминантных моделей – не только понимание того, за счет чего группы отличаются друг от друга, но и предсказание класса произвольного объекта при известных значениях независимых переменных. Поэтому, чтобы выполнить прогноз номера группы, требуется зафиксировать значение всех переменных.

Прогнозирование выполняется в три этапа:

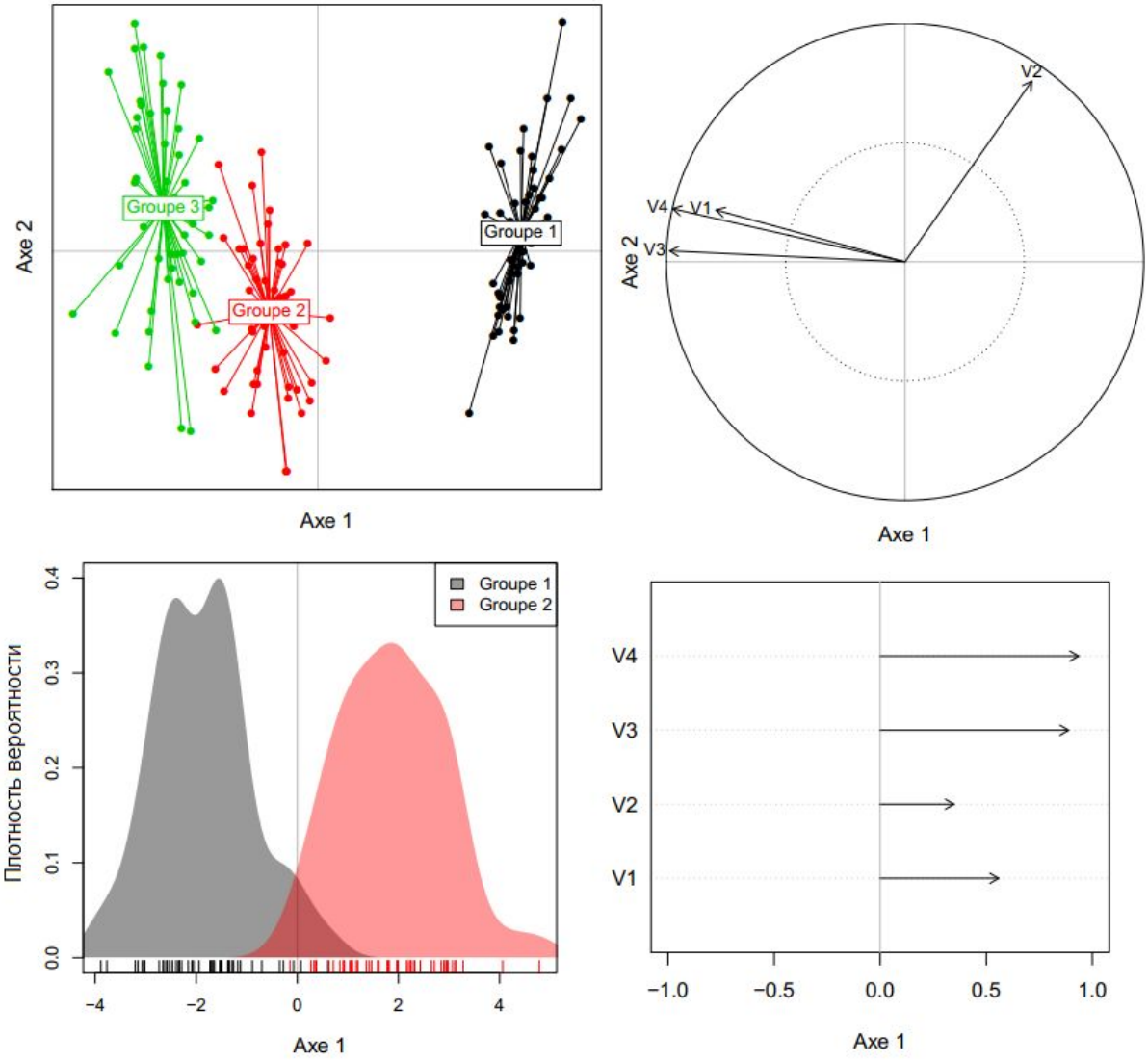
1. Создание из исходного набора серии подмоделей данных путем перекрестной проверки.

2. Формирование таблицы, содержащей столбцы с количественными переменными (имена столбцов должны быть строго идентичны именам переменных в исходном наборе данных). В каждую строку этой таблицы заносятся значения, на основе которых должно быть сделано предсказание (прогноз делается для каждой строки таблицы).

Если исходный набор данных был стандартизирован, необходимо также стандартизировать набор данных для прогнозирования. Для этого используют функцию из пакета `vegan`: `нов.таб <- stand(нов.таб, таблица)`, где `нов.таб` – массив для классификации объектов.

3. Реализация прогноза: `predict(LDA.cv, нов.таб)` , где `LDA.cv` является результатом перекрестной проверки (см. шаг 1). Функция возвращает для каждой строки `нов.таб` предсказанную группу (столбец `Group`) и вероятность этого предсказания (столбец `Prob`).

Примечание: преимущества использования большого количества подмоделей, полученных во время перекрестной проверки, заключаются в возможности выполнить прогноз с помощью каждой подмодели и обобщить полученные вероятности, что позволяет увеличить надежность "среднего" прогноза.



105. Дискриминантный анализ с использованием регрессии на основе частных наименьших квадратов (PLS-DA)

Англ. – *Partial least squares discriminant analysis* (PLS-DA) = *Projection to latent structures discriminant analysis*

Подготовка данных

Настоятельно рекомендуется стандартизировать количественные переменные (см. п. **88**). Предварительное преобразование также часто полезно, чтобы сделать связи между этими переменными более линейно выраженными. В любом случае, таблица количественных переменных должна быть преобразована в матрицу, например, `таблица <- as.matrix(таблица)`.

Примечание: если переменные были стандартизированы (см. п. **88**), результат стандартизации – это уже всегда матрица.

Фактор также должен быть преобразован в переменные-индикаторы. Для этого используют функцию: `var.ind <- dummy(фактор)*`, где `фактор` – это фактор, определяющий группы.

Проведение анализа

Для выполнения анализа PLS-DA используют функцию из пакета `pls`:
`PLSDA <- cppls(var.ind~таблица)`.

Примечание: по умолчанию создается столько осей, сколько есть количественных переменных в `таблица`, что иногда может привести к ошибке. Если это происходит, то уменьшите число осей, добавив аргумент `ncomp=10` (10 осей обычно оказывается в значительной степени больше, чем это необходимо).

Оценка качества анализа

Доля ошибок классификации. Одна из функций любого дискриминантного анализа состоит в том, чтобы предсказать группу, к которой принадлежит объект и для которого известны значения количественных переменных. Таким образом, один из способов оценки качества дискриминационного анализа заключается в том, чтобы проверить, насколько безошибочно он позволяет оценивать номер группы неизвестного объекта. Для этого используется метод кросс-проверки, который выполняет следующую процедуру: (а) набор данных случайным образом разбивается на части; (б) последовательно одна из частей выделяется для контроля, а по остальным осуществляется построение дискриминантной модели; (в) выполняется классификация контрольных объектов, не участвовавших в построении модели. Для большей надежности эти действия могут повторяться многократно (каждый раз при случайном разбиении набора данных).

Однако в случае PLS-DA тестирование существенно усложняется, и должна быть использована процедура двойной перекрестной проверки (*double cross model validation* или 2CV). В этом алгоритме на каждом шаге "внешней" кросс-проверки (*outer loop*) проводится вторая "внутренняя" перекрестная проверка (*inner loop*).

Для выполнения этой операции используется функция : `MVA.cmv(таблица, фактор,model="PPLS-DA",crit.inn="NMC")*`. По умолчанию разбиение исходной таблицы осуществляется на 7 частей на уровне внешнего цикла (7-*fold cross-validation*) и на 6 частей на уровне внутреннего цикла (6-*fold cross-validation*). Эти значения могут быть изменены с помощью аргументов `kout` и `kinn`. Функция также может регулировать эти величины автоматически, если, по крайней мере, одна группа содержит менее 7 объектов. По умолчанию вся процедура повторяется 10 раз (аргумент `repet`), что в конечном итоге приводит к

формированию 7×10 = 70 суб-моделей. Если вы хотите использовать PLS-DA в целях прогнозирования, сохраните результат функции `MVA.cmv()` в объекте.

Примечание 1: если *g* – количество групп, то, чтобы проверить и интерпретировать PLS-DA, часто используются *g* - 1 оси, потому что это теоретически достаточно. Чтобы следовать этому принципу, используйте аргумент `ncomp=nb`, где nb = *g* - 1.

Примечание 2 : существует несколько версий PLS-DA. Наиболее эффективной является PPLS-DA (*Powered* PLS-DA), которая и рекомендуется использовать в этом Путеводителе.

Тестирование. Проверка значимости компонентов дискриминантной модели выполняется с использованием двойной перекрестной проверки, реализуемой функцией: `MVA.test(таблица, фактор, model="PPLS-DA", cmv=TRUE)*`. В ней доступны аргументы `kout` , `kinn`, и, что особенно важно, `ncomp`. Оценка продолжительности времени, необходимого для этого теста, сравнительно велика.

Если фактор имеет значимый эффект и существует более двух групп, целесообразно провести множественные сравнения, чтобы определить, какие уровни группирующего фактора отличаются. Для этого можно использовать функцию: `pairwise.MVA.test(таблица, фактор, model="PPLS-DA", cmv=TRUE)*`. Всегда анализируйте аргументы `kout` , `kinn`, и особенно `ncomp`, поскольку оценка времени, необходимого для достижения всех парных сравнений часто очень велика.

Примечание: До любой интерпретации PLS-DA абсолютно необходимо проверить, оказывает ли фактор значимое влияние. Действительно, графические изображения всегда имеют тенденцию выделять отдельные группы, даже если это разбиение оказывается абсолютно случайным (это явление тем более вероятно, когда число количественных переменных значительно больше, чем объектов). Таким образом, нельзя полагаться на эти графики, если группы не различались статистически значимо.

Графические представления

В PLS-DA строятся два возможных графика: диаграмма ординации объектов, которая показывает их распределение по факториальной плоскости, состоящей из двух осей, и круг корреляций (см. п. **89**), который позволяет интерпретировать особенности распределения объектов.

График распределения объектов. Чтобы построить ординационную диаграмму, можно использовать функцию : `MVA.plot(LDA, fac=фактор)*`. По умолчанию представлены оси 1-й (по горизонтали) и 2-й (по вертикали) главных компонент. Они могут быть изменены с помощью аргументов `xax` и `yax`.

Аргументы `col`, `pch`, `fac.lab`, `contours`, `stars` и `barycenters` позволяют настроить эстетику изображения. См `?MVA.scoreplot` для многих других графических опций.

Корреляционный круг. Строится функцией: `MVA.plot(LDA,"corr")*`. Чтобы удалить стрелки, добавьте аргумент `arrows=FALSE`. См. `?MVA.corplot` для многих других графических опций.

Интерпретация

Предполагается, что группы, определяемые внешним фактором, существенно различаются. В противном случае интерпретация должна быть прекращена.

График отдельных объектов позволяет оценить, как структурируются группы, т. е. насколько они разделяются, или перекрываются друг с другом.

Круг корреляции позволяет выделить переменные, по которым эти группы различаются. Для этого на графике отдельных объектов выявляются направления, которые имеют отношение к биологической интерпретации (это могут быть оси или

любые диагонали), и определяются переменные, которые наиболее коррелируют с этими направлениями на корреляционном круге (см. п. **89**).

Прогноз на основе модели

Роль дискриминантных моделей – не только понимание того, за счет чего группы отличаются друг от друга, но и предсказание класса произвольного объекта при известных значениях независимых переменных. Поэтому, чтобы выполнить прогноз номера группы, требуется зафиксировать значение всех переменных.

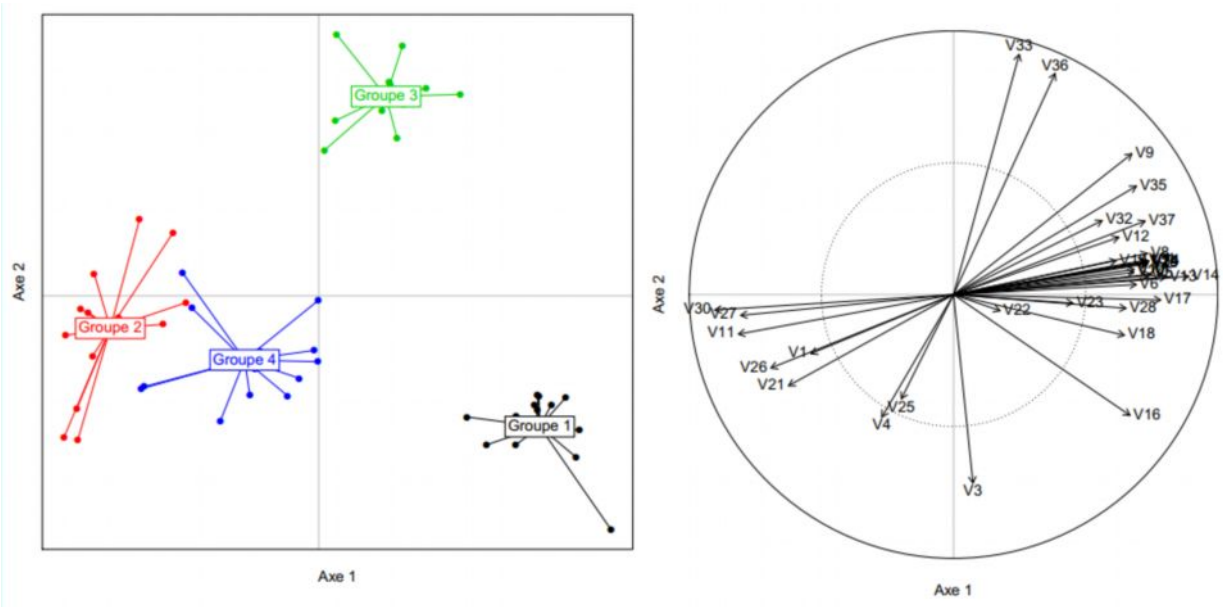
Прогнозирование выполняется в три этапа:

1. Создание из исходного набора серии подмоделей данных путем двойной перекрестной проверки.

2. Формирование таблицы, содержащей столбцы с количественными переменными (имена столбцов должны быть строго идентичны именам переменных в исходном наборе данных). В каждую строку этой таблицы заносятся значения, на основе которых должно быть сделано предсказание (прогноз делается для каждой строки таблицы). Если исходный набор данных был стандартизирован, необходимо также стандартизировать набор данных для прогнозирования. Для этого используют функцию из пакета `vegan`: `нов.таб <-stand (нов.таб, таблица)`, где `нов.таб` – массив для классификации объектов.

3. Реализация прогноза: `predict(PLSDA.vc, нов.таб)`, где `PLSDA.vc` является результатом двойной перекрестной проверки (см. шаг 1). Функция возвращает для каждой строки `нов.таб` предсказанную группу (столбец `Group`) и вероятность этого предсказания (столбец `Prob`).

Примечание: преимущества использования большого количества подмоделей, полученных во время двойной перекрестной проверки, заключаются в возможности выполнить прогноз с помощью каждой подмодели и обобщить полученные вероятности, что позволяет увеличить надежность "среднего" прогноза.



106. Канонический анализ соответствий (CCA)

Англ. – *Canonical correspondence analysis* (CCA) =

Correspondence analysis with respect to instrumental variables

Подготовка данных

Обычно для таблицы сопряженности или в массиве присутствия-отсутствия строки и столбцы в определенном смысле симметричны, т. е. нет "множества объектов" и "множества переменных". Однако канонический анализ соответствий CCA требует, чтобы объекты, для которых определены объясняющие переменные, были размещены по строкам как в обычной таблице. Таким образом, мы будем считать, что в строках присутствуют "объекты".

Проведение анализа

Для выполнения анализа можно использовать функцию из пакета `vegan`: `CCA <- cca(формула, data=таб.X)` , где `таб.X` является таблицей, содержащей объясняющие переменные. Аргумент `формула` (см. п. **40**) в левой части в качестве отклика должен содержать таблицу `таб.Y` , включающую объясняемые переменные.

Общая объясняющая способность

Канонический анализ соответствий CCA состоит из двух этапов :

1. Вариация данных в объясняемой таблице **Y** разлагается на две составляющие: (а) вызываемая внешними переменными (так называемая "ограниченная" или объясняемая вариация) и (б) остаточная вариация (т.е. "неограничиваемая" или необъясняемая). CCA работает с определенной информационной мерой, называемой *инерцией*, которая оценивает соответствие между строками и столбцами набора данных (инерция на самом деле является более общим термином и использование ее для оценки соответствия является частным случаем). На этом этапе таблица многомерного отклика **Y** считается асимметричной и ее строки играют роль отдельных объектов.
2. Реализуются два отдельных анализа соответствий: первый CA на основе инерции от влияния внешних переменных ("ограниченные оси соответствия") и второй CA – без учета внешних факторов ("неограничиваемые оси соответствия ").

Общий объясняющий потенциал анализа CCA можно оценить с помощью доли "ограниченной инерции" (т.е. инерции в данных, которая объясняется внешними переменными). Чем выше эта доля, тем более полезна сформированная информационная структура. Чтобы получить эти доли, используйте, например, функцию: `MVA.synt(CCA)`*, где нужный результат в первой возвращаемой таблице.

Оценка качества анализа

Тесты внешних переменных. Значимость эффекта воздействия внешних переменных проверяется с использованием функции `MVA.anova(RDA)`*. При этом осуществляется тест по *F*-критерию с использованием рандомизации. Если хотя бы одна объясняющая переменная оказывает значимое влияние, то можно опираться на результаты RDA для интерпретации. Если ни одна из объясняющих переменных не имеет значимого *p*-значения, то интерпретация результатов такого анализа вряд ли вызовет большой интерес, поскольку эффект влияния внешних переменных не обозначен.

Если влияние внешнего фактора было оценено как значимое, то можно провести множественные сравнения между уровнями фактора (или комбинациями условий взаимодействия факторов). Для выполнения множественных сравнений используйте функцию: `pairwise.factorfit(CCA, Фактор)`*, где `Фактор` является интересуемым внешним фактором.

Обобщенная оценка. Если хотя бы одна объясняющая переменная имеет значимый эффект, то интерес представляет блок результатов по "ограниченному" CA. Как и в случае с обычным анализом соответствий (см. п. **97**), качество этого анализа оценивается по доле инерции, объясненной каждой осью. Чтобы получить эти доли, используйте, например, функцию : `MVA.synt(CCA)*`, где результат представлен во второй возвращаемой таблице.

Примечание 1: здесь указывается доля не от общей инерции, как это имеет место в обычном CA, а от "ограниченной инерции".

Примечание 2: доля инерции всегда располагается в порядке убывания (т. е. ось 1 объясняет больше инерции, чем ось 2, которая объясняет больше, чем ось 3...).

Примечание 3: нет абсолютного правила, сколько осей следует выделить для интерпретации. Это всегда компромисс между хорошим обобщением информации, содержащейся в исходных данных (которое увеличивается с количеством осей) и легкостью для интерпретации (которая уменьшается с количеством осей).

Графические представления

Если в составе объясняющих переменных (т.е. таблице `таб.X`) есть хотя бы одна количественная переменная, то в CCA возможно построение двух графиков: диаграммы ассоциации объектов, которая показывает их распределение по факториальной плоскости, состоящей из двух главных осей, и корреляционного круга (см. п. **89**), который позволяет интерпретировать это распределение. Если нет ни одной количественной объясняющей переменной, то единственным возможным графиком является ассоциация.

График ассоциации объектов. На этом графике строки и столбцы представлены каждой отдельной точкой. Однако невозможно представить одновременно на одном и том же графике расстояния как между объектами по строкам, так и по столбцам, без некоторого их смещения. Таким образом, следует выбирать между представлением без смещения по строкам (масштаб типа 1) или столбцам (масштаб типа 2).

Для представления графика ассоциаций без смещения расстояний между строками используется функция: `MVA.plot(CCA, points=FALSE, scaling=1)*`. Аналогичный график без смещения расстояний между столбцами формируется командой: `MVA.plot(CCA, points=FALSE, scaling=2)*`. По умолчанию представлены оси 1-й (по горизонтали) и 2-й (по вертикали) главных компонент. Они могут быть изменены с помощью аргументов `xax` и `yax`.

Чтобы представить график только CA без учета влияния внешних факторов, добавьте аргумент `space=2`. Аргументы `col`, `pch`, `fac.lab`, `contours`, `stars` и `barycenters` позволяют настроить эстетику изображения. См `?MVA.scoreplot` для многих других графических опций.

Чтобы добавить на график разделение объектов на группы, необходимо выполнить следующую процедуру:

```
> график <- MVA.plot(CCA, points=FALSE, col=colors)*,
```

где `colors` – вектор цветов с двумя значениями: первый для столбцов, а второй для строк. Если вы хотите отобразить группировку по столбцам, то первый цвет должен быть `"white"`, а для группировки строк второй цвет должен быть `"white"`;.

```
> MVA.plot(CCA, points =FALSE, xlim= график $xlim,
          ylim= график $ylim, set=nb, fac=фактор) *,
```

где `nb` равно 1 для группировки строк или 2 для группировки столбцов, а фактор-это фактор, определяющий группу каждой строки / столбца. Аргументы `col`, `pch`, `fac.lab`, `contours`, `stars` и `barycenters` позволяют настроить эстетику изображения. См `?MVA.scoreplot` для многих других графических опций.

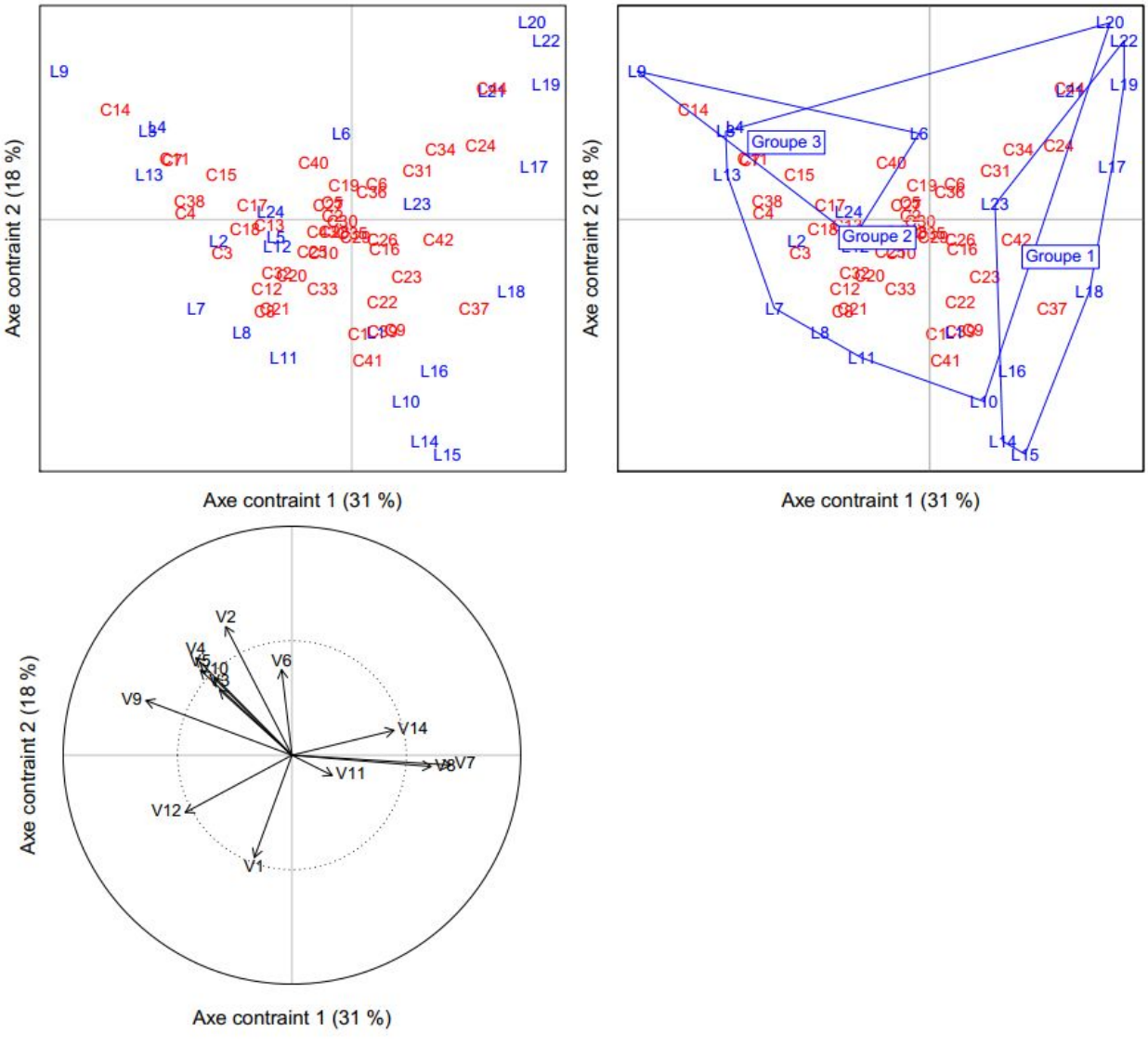
Корреляционный круг. Строится функцией: `MVA.plot(CCA,"corr")*`. На графике представлены только внешние количественные переменные из таблицы **X**. Если они отсутствуют, то график не имеет смысла. Чтобы удалить стрелки, добавьте аргумент `arrows=FALSE`. См. `?MVA.corplot` для многих других графических опций.

Интерпретация

По нашему мнению, CCA – это единственный корректный способ интерпретировать результаты о связи многомерного отклика с внешними переменными.

График ассоциации отдельных объектов позволяет (а) идентифицировать связи между строками и столбцами таблицы, которые могут быть объяснены так же, как и в обычном FCA (см. п. **97**), (б) оценить, какие точки объектов образуют сгущения для разных уровней внешнего фактора и (в) выделить линейные градиенты точек относительно внешних ковариат из таблицы **X**.

Круг корреляций позволяет определить переменные, которые влияют на возможную группировку объектов и/или являются причиной линейной упорядоченности точек. Для этого мы на графике ассоциации объектов определяем, какие направления могут иметь отношение к биологической интерпретации (это могут быть главных оси соответствия или любые диагонали), и выделяем по корреляционному кругу переменные, которые наиболее коррелируют с этими направлениями (см. п. **89**).



107. Дискриминантный анализ соответствий (CDA)

Англ. – *Correspondence discriminant analysis* (CDA)

Подготовка данных

Обычно для таблицы сопряженности или в массиве присутствия-отсутствия строки и столбцы в определенном смысле симметричны, т. е. нет "множества объектов" и "множества переменных". Однако дискриминантный анализ соответствий CDA требует, чтобы объекты, с которыми связан группирующий фактор, размещались в строках, как в обычной таблице. Таким образом, мы будем считать, что в строках присутствуют "объекты".

Примечание: CDA состоит из двух этапов. Первый из них заключается в том, чтобы выполнить анализ соответствия (см. п. **97**) для исходной таблицы сопряженности (или присутствия-отсутствия). Таблица при этом рассматривается как эквивалентная относительно строк и столбцов. На втором этапе координаты строк таблицы сопряженности на осях CA используются для выполнения LDA (см. п. **104**), в котором столбцы в таблице сопряженности будут использоваться в качестве переменных для построения линейной дискриминантной модели.

Проведение анализа

Для выполнения анализа можно использовать функцию из пакета `ade4`:

```
CDA <- discrimin.coa(таб.Y,фактор, scannf=FALSE,
                                   nf=nlevels(фактор)-1),
```

где `таб.Y` является таблицей таблицы сопряженности (или присутствия-отсутствия), содержащей объясняемые переменные, а `фактор` определяет группировку ее строк.

Оценка качества анализа

Доля ошибок классификации. Одна из функций любого дискриминантного анализа состоит в том, чтобы предсказать группу, к которой принадлежит объект и для которого известны значения количественных переменных (т.е. столбцы в таблице сопряженности). Таким образом, один из способов оценки качества дискриминантного анализа заключается в том, чтобы проверить, насколько безошибочно он позволяет оценивать группу, к которой возможно принадлежит неизвестный объект. Для этого используется метод кросс-проверки, который выполняет следующую процедуру: (а) набор данных случайным образом разбивается на части; (б) последовательно одна из частей выделяется для контроля, а по остальным осуществляется построение дискриминантной модели; (в) выполняется классификация контрольных объектов, не участвовавших в построении модели. Для большей надежности эти действия могут повторяться многократно (каждый раз при случайном разбиении набора данных).

Для выполнения перекрестной проверки может быть использована функция : `CDA.cv(таб.Y, фактор)*`. По умолчанию функция разбивает набор данных на 7 частей (7-*fold cross-validation*), но этот параметр может быть изменен с помощью аргумента `K=NB`, где `NB` – это количество поднаборов данных для разбиения. Функция может также автоматически настроить это число, если, по крайней мере, одна группа содержит менее 7 объектов. По умолчанию вся процедура повторяется 10 раз (аргумент `repet`), т.е. в конечном итоге формируется 7×10 = 70 подмоделей.

Если вы хотите использовать CDA для целей прогноза, сохраните результат работы функции `CDA.cv()` в соответствующем объекте.

Тестирование. Тест, который необходимо выполнить, основан на использовании координат отдельных объектов на осях ординации CA. Проверка значимости компонентов модели CDA реализуется функцией: `CDA.test(массив, фактор)*`.

Если g – количество групп, то по умолчанию для тестирования выбирается g - 1 осей CDA. Это число осей может быть изменено с помощью аргумента `ncomp=NB`, где `NB` является желаемым для тестирования количеством осей.

Если сохраняется только одна ось (по выбору или потому, что группировка велась только на два класса), то проводится тест ANOVA. Если выбрано по крайней мере две оси, то используется MANOVA (*Multivariate ANalysis Of VAriance*), т.е. расширение дисперсионного анализа на многомерный случай.

Если фактор имеет значимый эффект и существует более двух групп, целесообразно провести множественные сравнения, чтобы определить, какие уровни группирующего фактора отличаются между собой. Для этого используйте функцию: `pairwise.CDA.test(таб.Y,фактор)*`. Всегда принимайте во внимание аргумент `ncomp`, поскольку оценка времени, необходимого для выполнения всех парных сравнений, может быть достаточно велика.

Примечание: если фактор не имеет значимого эффекта, то интерпретация результатов CDA некорректна, а любая попытка предсказания бессмысленна.

Графические представления

В CDA возможно построение двух графиков: диаграммы ассоциации объектов, которая показывает их распределение по факториальной плоскости, состоящей из двух главных осей, и корреляционного круга (см. п. **89**), который позволяет интерпретировать это распределение.

Примечание: если существует только две группы, CDA производит только одну ось. В этом случае диаграмма ассоциации объектов и круг корреляций сводятся к одному измерению.

График ассоциации объектов. Для представления ординационной диаграммы используется функция: `MVA.plot(CDA, fac=фактор)*`. По умолчанию представлены оси 1-й (по горизонтали) и 2-й (по вертикали) главных соответствий. Они могут быть изменены с помощью аргументов `xax` и `yax`. Аргументы `col`, `pch`, `fac.lab`, `contours`, `stars` и `barycenters` позволяют настроить эстетику изображения. См `?MVA.scoreplot` для многих других графических опций.

Корреляционный круг. Строится функцией: `MVA.plot(CDA, "corr")*`. Чтобы удалить стрелки, добавьте аргумент `arrows=FALSE`. См. `?MVA.corplot` для многих других графических опций.

Интерпретация

График ассоциации отдельных объектов позволяет оценить, как структурируются группы, т. е. насколько они разделяются или перекрываются друг с другом.

Круг корреляции позволяет выделить переменные, по которым эти группы различаются. Для этого на графике отдельных объектов выявляются направления, которые имеют отношение к биологической интерпретации (это могут быть оси или любые диагонали), и определяются переменные, которые наиболее коррелируют с этими направлениями на корреляционном круге (см. п. **89**).

Прогноз на основе модели

Роль дискриминантных моделей – не только понимание того, за счет чего группы отличаются друг от друга, но и предсказание класса произвольного объекта при известных значениях независимых переменных. Поэтому, чтобы выполнить прогноз номера группы, требуется зафиксировать значение всех переменных.

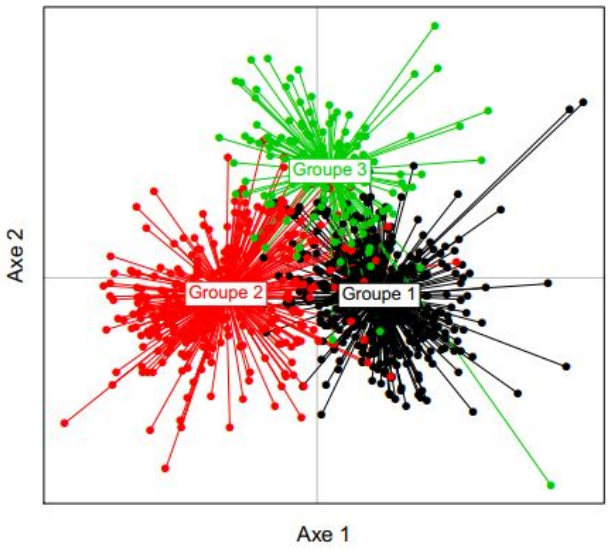
Прогнозирование выполняется в три этапа:

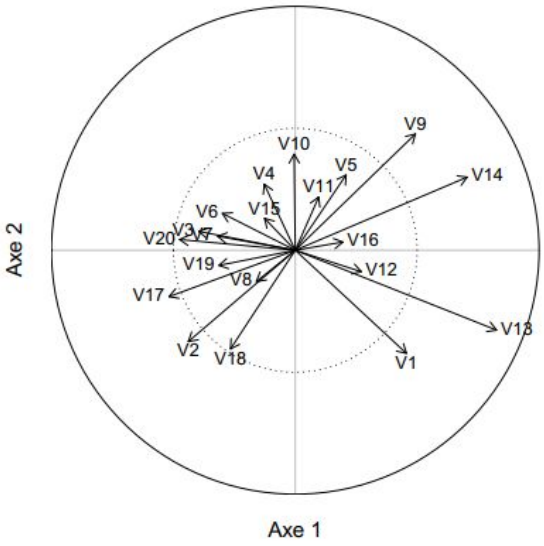
1. Создание из исходного набора серии подмоделей данных путем перекрестной проверки.

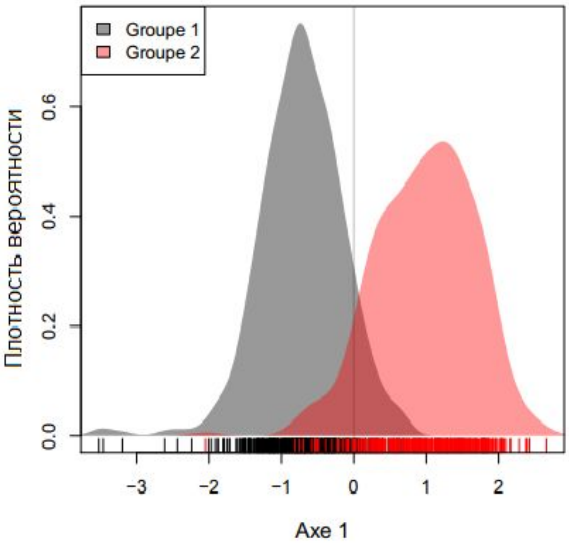
2. Формирование таблицы, содержащей по одному столбцу для каждого столбца исходной таблицы (имена столбцов должны строго совпадать с именами в исходном наборе данных), и заполнение каждой строки значениями, по которым должно быть сделано предсказание.

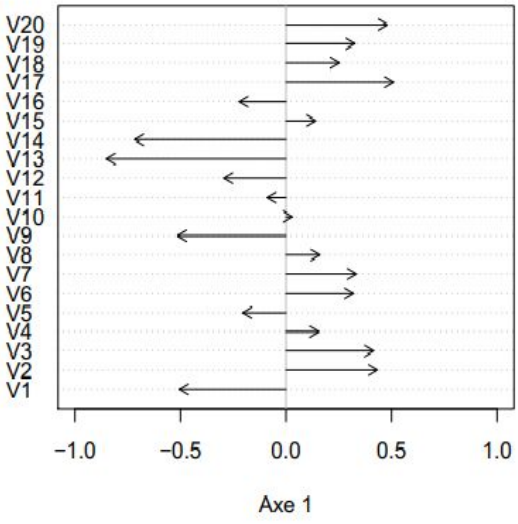
3. Реализация прогноза: `predict(CDA.vc, нов.таб)` , где `CDA.vc` является результатом перекрестной проверки (см. шаг 1). Функция возвращает для каждой строки `нов.таб` предсказанную группу (столбец `Group`) и вероятность этого предсказания (столбец `Prob`).

Примечание: преимущества использования большого количества подмоделей, полученных во время перекрестной проверки, заключаются в возможности выполнить прогноз с помощью каждой подмодели и обобщить полученные вероятности, что позволяет увеличить надежность "среднего" прогноза.









108. Многомерная линейная модель

Обсуждаемые здесь процедуры анализа используют основные понятия построения моделей, изложенные в п. **39- 42**.

Подготовка данных

Обычно для таблицы сопряженности или в массиве присутствия-отсутствия строки и столбцы в определенном смысле симметричны, т. е. нет "множества объектов" и "множества переменных". Однако канонический анализ соответствий CCA требует, чтобы объекты, для которых определены объясняющие переменные, были помещены в строках как в обычной таблице. Таким образом, мы будем считать, что в строках присутствуют "объекты".

Проведение анализа

Для выполнения анализа можно использовать функцию из пакета `vegan`:
`CCA <- cca(формула, data=таб.X)` , где `таб.X` является таблицей, содержащей объясняющие переменные. Аргумент `формула` (см. п. **40**) в левой части в качестве отклика должен содержать таблицу `таб.Y` , включающую объясняемые переменные.

Отклик

Модель, которая будет использоваться, предполагает, что дисперсионно-ковариационные матрицы (многомерный эквивалент дисперсии) являются однородными по отношению к различными уровням качественных объясняющих переменных. Чтобы проверить это, можно использовать функцию из из пакета `vegan`:
`anova(betadisper(dist (таб.Y), фактор))`, где `таб.Y` –исходный массив объясняемых данных, для которых `фактор` определяет группировку. Обычно модель достаточно устойчива при умеренном несоблюдении этого условия. Если это предположение вообще не соблюдается, то предварительное преобразование эмпирической таблицы может в значительной степени помочь улучшить ситуацию (см. п. **88**).

В любом случае, количество переменных, подлежащие объяснению, должно быть намного меньше, чем число наблюдаемых объектов, и они не должны быть слишком коррелировать друг с другом. Если это двойное условие не соблюдается, преобразуйте таблицу, которую нужно объяснить, в матрицу расстояния (см. п. **100**) и используйте эту матрицу в качестве отклика для аналогичного анализа (см. п. **110**).
После выполнения этих проверок и, возможно, преобразования объясняемой таблицы, она должна быть преобразована в матрицу: `таблица <- as.matrix(таблица)`.

Типы модели

Используемая модель представляет собой Многомерную Линейную модель (*Multivariate Linear Model* или MLM), которая является расширением классической линейной модели с одной объясняемой переменной (см. п. **76**). В отличие от обычной линейной модели, отклик MLM включает несколько переменных.

Построение модели

Для создания модели используют обычную базовую функцию:
`модель <- lm(формула).`
См. п. **40** для подробного объяснения особенностей построения формулы. В этой формуле отклик является матрицей (т. е. таблицей, которую нужно объяснить).

В целом, как всегда, можно сказать, что :

- включение независимого фактора позволяет проверить, различается ли положение отдельных объектов в многомерном пространстве переменных, подлежащих объяснению, относительно уровней этого фактора;
- включение ковариаты позволяет проверить, существует связь между этой ковариатой и объясняемыми переменными;
- включение взаимодействия двух независимых переменных позволяет проверить, зависит ли один эффект от другого (это же рассуждение – для наблюдений, приуроченных к двум и более отрезкам времени). Особый случай – анализ взаимодействия между ковариатой и фактором, который позволяет проверить, зависит ли отношение между независимой переменной и откликом от уровней фактора.

Примечание: несмотря на то, что анализ влияния ковариат не является теоретической проблемой в MLM, на практике трудно представить такой эффект в многомерной структуре (за исключением того, что он интерпретируется отдельно для каждой объясняемой переменной). Но анализ влияния факторов для этого типа модели вполне корректен.

Проверка адекватности модели

Прежде чем проводить дальнейший анализ, необходимо проверить, насколько хорошо модель подогнана к анализируемым данным. Этот этап фундаментален для любой модели, так как любой тест, проводимый на недостаточно адекватной модели, просто ненадежен (или не соответствует действительности). Однако подогнанную MLM-модель сложнее проверить, чем модель с одномерным откликом. Единственным реально оцениваемым критерием является распределение остатков модели (т.е. разностей между фактическими наблюдаемыми значениями и значениями, предсказанными моделью). Для правильно подогнанной модели остатки должны иметь многомерное нормальное распределение (расширение обычного распределения на многомерный случай).

Для выполнения этой проверки используйте функцию: `plotresid(модель)*`. Предположение о нормальности принимается, если точки примерно выровнены по одной прямой. Эти точки могут полностью не располагаться по прямой линии, но если они остаются примерно в доверительном интервале (показан пунктиром), то подгонка модели считается корректной.

Если предположение о нормальности остатков отвергается, альтернативой MLM является преобразование всех переменных, которые необходимо объяснить, в матрицу расстояний (см. п. **100**) и использование этой матрицы в качестве отклика для аналогичного анализа (см. п. **110**).

Тестирование

Проверка значимости компонентов многомерной модели выполняется с использованием MANOVA (*Multivariate ANalysis of VAriance*), который является расширением ANOVA на многомерный случай. Для проведения теста используйте функцию из пакета `car` : `Manova(модель)`. Проводимый тест относится к Типу II (подробное объяснение проверки гипотез см. в п. **42**).

Если фактор имеет значимый эффект и существует более двух групп, целесообразно провести множественные сравнения, чтобы определить, какие уровни группирующего фактора отличаются. См. п. **43** для описания техники этих сравнений.

Если ковариата имеет значимое влияние, то на самом деле нет другого решения для оценки направления действия эффекта, кроме как отдельно изучить его по каждой объясняемой переменной. Это направление задается знаком соответствующего параметра. Полную информацию обо всех параметрах модели можно получить с помощью команды `summary(модель)`. Значения самих коэффициентов называются

Estimate и находятся в таблице Coefficients. Если знак коэффициента для количественной переменной отрицательный, то значение соответствующей объясняемой переменной уменьшается при увеличении ковариаты, а если он положительный, то имеем обратную тенденцию.

Прогноз на основе модели

Смысл моделей – не только показать роль различных независимых переменных в изменчивости матрицы объясняемых переменных, но и предсказать значение, которое бы принял отклик при известных значениях независимых переменных. Поэтому, чтобы оценить значение одной из объясняемых переменных, требуется зафиксировать значение всех независимых переменных.

Для прогнозирования могут быть использованы два метода и оба они основаны на функции predict():

- значение каждой из независимых переменных дается непосредственно в теле функции в виде списка (с одной или несколькими записями): predict(модель,newdata=list(переменные)), где переменные – это последовательность var1= значение,var2=значение...
- создать таблицу, содержащую столбцы по каждой независимой переменной (имена столбцов должны строго совпадать с именами переменных модели), и заполнить каждую строку значениями, для которых должно быть сделано предсказание. Таким образом, делается прогноз по каждой строке таблицы:

predict(модель, newdata = таблица).

ПРИМЕРЫ

Имеем модель, содержащую фактор с двумя уровнями (A и B), ковариату ковариата в пределах от 0 до 30, и их взаимодействия:

```
>  модель <- lm(отклик~фактор*ковариата)
```

Можно предсказать значение каждой объясняемой переменной таким образом:

```
> predict(модель, newdata= list(фактор="A", ковариата=10)
```

Или для нескольких прогнозов:

```
> predict(модель,newdata=list(фактор =c("A","B"),
                    ковариата =c(10,10))
```

Или предварительно создать таблицу следующего типа:

```
> таблица
  Фактор ковариата
1      A        10
2      B        10
> predict(модель, newdata= таблица)
```

Асимметричный анализ с использованием матрицы дистанций

Выбор метода анализа зависит от цели исследования :

1. Цель может состоять в том, чтобы выполнить ординацию; тогда используйте анализ избыточности на основе матрицы расстояния.
2. Цель может состоять в том, чтобы просто выполнить статистический тест и проанализировать матрицу расстояния.

109. Анализ избыточности на основе матрицы расстояний (db-RDA)

Англ. – *Distance-based redundancy analysis* (db-RDA)

Подготовка данных

Необходимо предварительно проверить, является ли матрица расстояния евклидовой или нет (см. п. **100**).

Проведение анализа

Для осуществления анализа избыточности можно, например, использовать функцию из пакета `vegan`: `dbRDA <- dbrda(формула, data=таб.X)`, где `таб.X` является таблицей, содержащей объясняющие переменные. Аргумент `формула` (см. п. **40**) в левой части в качестве отклика должен содержать матрицу дистанций `мат.раст`. Если эта матрица неевклидова, необходимо применить исправление. Для этого добавьте аргумент `add=TRUE`.

Общая объясняющая способность

Анализ избыточности db-RDA состоит из двух этапов :

1. Выполняется анализ главных координат PCoA на матрице расстояния (см. п. **101**).
2. Реализуется анализ избыточности RDA (см. п. **103**) на основе результатов PCoA. Результат RDA состоит из двух блоков анализа главных компонент PCA (см. п. **96**): первый PCA на основе дисперсии от влияния внешних переменных ("ограниченные главные компоненты") и второй PCA – без учета внешних факторов ("неограничиваемые главные компоненты").

Общий объясняющий потенциал анализа db-RDA можно оценить с помощью доли "ограниченной дисперсии" (т.е. дисперсии матрицы расстояний, которая объясняется внешними переменными). Чем выше эта доля, тем более полезна сформированная информационная структура, тем больше вариация компонентов `мат.раст`, которую следует объяснить, связана с независимыми переменными **X**. Чтобы получить эти доли, используйте, например, функцию: `MVA.synt(dbRDA)*`, где результат располагается в первой возвращаемой таблице.

Оценка качества анализа

Тесты внешних переменных. Значимость эффекта воздействия внешних переменных проверяется с использованием функции `MVA.anova(dbRDA)*`. При этом осуществляется тест по *F*-критерию с использованием рандомизации. Если хотя бы одна объясняющая переменная оказывает значимое влияние, то можно опираться на результаты db-RDA для интерпретации. Если ни одна из объясняющих переменных не имеет значимого *p*-значения, то интерпретация результатов такого анализа вряд ли вызовет большой интерес, поскольку эффект влияния внешних переменных не установлен.

Если влияние внешнего фактора было оценено как значимое, то можно провести множественные сравнения между уровнями фактора (или комбинациями условий взаимодействия факторов). Для выполнения множественных сравнений используйте

функцию: `pairwise.factorfit(dbRDA, Фактор)*`, где `Фактор` является интересуемым внешним фактором.

Обобщенная оценка. Если хотя бы одна объясняющая переменная имеет значимый эффект, то интерес представляет блок результатов по "неограничиваемому" PCA. Как и в случае с классическим PCA (см. п. **96**), качество этого анализа оценивается по доле дисперсии, объясненной каждой осью. Чтобы получить эти доли, используйте, например, функцию: `MVA.synt(dbRDA)*`, где результат – во второй возвращаемой таблице.

Примечание 1: здесь указываются доли от "ограниченной дисперсии", а не от общей, как это имеет место в классическом PCA.

Примечание 2: доля дисперсии всегда располагается в порядке убывания (т. е. ось 1 объясняет больше дисперсии, чем ось 2, которая объясняет больше, чем ось 3...).

Примечание 3: нет абсолютного правила, сколько осей следует выделить для интерпретации. Это всегда компромисс между хорошим обобщением информации, содержащейся в исходных данных (которое увеличивается с количеством осей), и легкостью для интерпретации (которая уменьшается с количеством осей).

Графические представления

Если в составе объясняющих переменных (т.е. таблице `таб.X`) есть хотя бы одна количественная переменная, то в db-RDA возможно построение двух графиков: диаграммы ординации объектов, которая показывает их распределение по факториальной плоскости, состоящей из двух главных осей, и корреляционного круга (см. п. **89**), который позволяет интерпретировать это распределение. Если нет ни одной количественной объясняющей переменной, то единственным возможным графиком является ординация объектов.

График ординации объектов. Для представления ординационной диаграммы используется функция : `MVA.plot(dbRDA)*`. По умолчанию представлены оси 1-й (по горизонтали) и 2-й (по вертикали) главных компонент. Они могут быть изменены с помощью аргументов `xax` и `yax`.

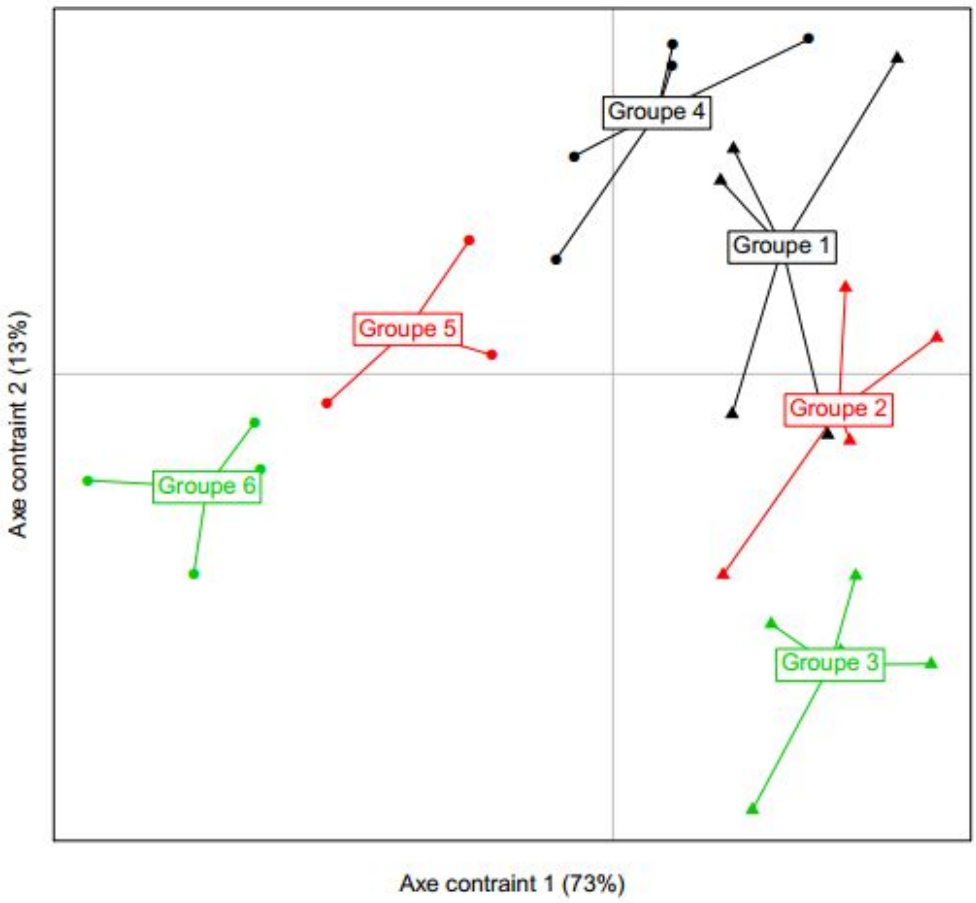
Чтобы добавить на график разделение объектов на группы, используйте аргумент `fac=фактор`, где `фактор` определяет принадлежность каждого объекта к группе. Аргументы `col`, `pch`, `fac.lab`, `contours`, `stars` и `barycenters` позволяют настроить эстетику изображения. См `?MVA.scoreplot` для многих других графических опций.

Корреляционный круг. Строится функцией: `MVA.plot(dbRDA,"corr")*`. На графике представлены только внешние количественные переменные из таблицы **X**. Если они отсутствуют, то график не имеет смысла. Чтобы удалить стрелки, добавьте аргумент `arrows=FALSE`.

Интерпретация

График ординации отдельных объектов позволяет: (а) идентифицировать связи между строками и столбцами таблицы, которые могут быть объяснены (так же, как и в обычном PCA, см. п. **96**), (б) оценить, как точки объектов образуют сгущения для разных уровней внешнего фактора и (в) выделить линейные градиенты относительно внешних ковариат из таблицы **X** .

Круг корреляции позволяет выделить объясняющие количественные переменные, которые формируют эти градиенты. Для этого на графике отдельных объектов выявляются направления, которые имеют отношение к биологической интерпретации (это могут быть оси или любые диагонали), и определяются переменные, которые наиболее коррелируют с этими направлениями на корреляционном круге (см. п. **89**).



110. Анализ связей с одной матрицей дистанций

Тест, который необходимо выполнить, является непараметрическим и его часто называют «MANOVA с использованием рандомизации». Он основан на использовании многомерной линейной модели (довольно похожей на MLM, см. п. **108**) и позволяет изучить влияние нескольких объясняющих переменных (количественных и/или качественных, а также, возможно, их взаимодействий). Однако *p*-значения получают с использованием алгоритма перестановки, поэтому тест считается непараметрическим.

Для проведения теста используйте функцию: `adonis.II(формула)*`. См. п. **40** для подробного объяснения методики построения формулы. В этой формуле отклик – матрица расстояния и эта матрица должна быть евклидовой.

Если какой-либо фактор (или взаимодействие нескольких факторов) оказывает значимое влияние, то можно провести множественные сравнения между уровнями фактора (или комбинациями условий взаимодействия факторов). Для выполнения множественных сравнений используйте функцию:
`pairwise.perm.manova(мат.раст, фактор)*`, где `Фактор` является интересуемым внешним фактором.

Если значимый эффект имеет ковариата, то интерпретация не так проста. Возможными действиями являются либо выполнение анализа главных координат PCoA на матрице расстояний (см. п. **101**), а затем сопоставление его результатов с ковариатой (см. п. **91**), либо выполнение db-RDA (см. п. **109**). Первый подход предпочтительнее, если ковариация не является контролируемой переменной.

Симметричный анализ двух таблиц

Обсуждаемые ниже методы анализа могут касаться исходных таблиц переменных и/или массивов значений координат в осях, сформированных предыдущей ординацией. Пользуясь этим подходом, можно связать две матрицы расстояния, таблицу переменных и матрицу расстояния, массив количественных переменных и таблицу качественных факторов и т.д. Возможности тут многочисленны.

Выбор анализа в основном зависит от привычек и традиций в различных научных дисциплинах. Можно перечислить основные отличительные черты различных методов:

- Анализ методом частных наименьших квадратов с двумя блоками данных (2B-PLS) используется в основном для анализа соответствия между двумя массивами переменных (обязательно количественных) без ограничений по их числу.
- - Прокрустовый анализ используется главным образом для анализа соответствия между двумя таблицами, в каждый з которых по две переменных (в обязательном порядке количественных), или для проверки соответствия между двумя ординациями с двумя осями в каждой.
- - Совместный инерционный (ко-инерционный) анализ используется главным образом для анализа соответствия между двумя массивами переменных (всех типов) или для проверки соответствия между двумя ординациями без ограничений по количеству переменных / осей.
- - Прокрустовый совместный инерционный анализ обобщает прокустовый анализ, если хотя бы одна таблица содержит более двух переменных. В случае двух количественных массивов переменных его легче интерпретировать, чем двублочный анализ PLS и анализ совместной инерции.

111. Двублочный метод частных наименьших квадратов (2B-PLS)

Англ. – *2-block partial least squares* (2B-PLS) =
2-block projection to latent structures (*2B-PLS*)

Подготовка данных

Двублочный метод частных наименьших квадратов 2B-PLS работает корректнее, если переменные имеют приблизительно нормальное (по крайней мере, симметричное) распределение и связаны между собой линейными отношениями. Предварительное преобразование данных может в значительной степени помочь улучшить ситуацию (см. п. **88**).

Рекомендуется, как правило, стандартизировать переменные перед анализом (см. п. **88**). Это придает одинаковый вес всем переменным и формирует результаты с применением коэффициента корреляции, что часто проще. В этом разделее будет предполагаться, что переменные стандартизированы.

Проведение анализа

Для осуществления двублочного анализа по методу частных наименьших квадратов можно использовать функцию из пакета `mixOmics`:
`PLS2B<-pls(таблица1, таблица2, mode="canonical")` с двумя таблицами `таблица1` и `таблица2` в качестве исходных данных. Если вы хотите стандартизировать переменные, но не выполнили эту операцию раньше, функция `pls()` сделает это по умолчанию. Также по умолчанию создается две пары осей и, чтобы изменить это значение, добавьте аргумент `ncomp=NB`, где `NB` –необходимое вам количество пар осей.

Примечание: мы говорим в 2B-PLS о парах осей, потому что анализ создает две отдельных ординационных модели, одну для первого массива, а другую для второго. Таким образом, у нас есть ось 1 для первой таблицы и ось 1 для второй таблицы (и так далее).

Оценка качества анализа

Тесты внешних переменных. Цель 2B-PLS состоит в том, чтобы сформировать некоторую информационную структуру, которая наилучшим образом оценивает ковариацию между двумя наборами данных. Но для того, чтобы интерпретация анализа была актуальной, нужно, чтобы оценка ковариации была значимой. Значимость ковариации проверяется функцией `cov.test(таблица1, таблица2)*`, реализующей перестановочный тест. Если *p*-величина оказывается значимой, результаты 2B-PLS могут быть интерпретированы. В противном случае объяснение не имеет смысла, поскольку между этими двумя таблицами нет существенной связи.

Примечание: ковариация тесно связана с понятием корреляции (см. п. **84**). Таким образом, она может быть отрицательной или положительной, указывая на отрицательную или положительную связь между двумя таблицами (или частями этих таблиц). Однако ковариация не ограничена диапазоном от -1 до +1, поэтому ее абсолютное значение напрямую не интерпретируется (отсюда и интерес к корреляции, которая является стандартизированной версией ковариации).

Обобщенная оценка. Если ковариация между двумя массивами является значимой, считается, что качество анализа оценивается по доле ковариации, объясненной каждой парой осей. Эти доли можно получить с помощью `MVA.synt(PLS2B)*` в первом блоке таблиц, возвращаемых функцией.

Примечание 1: доли, возвращаемые функцией `MVA.synt()`, на самом деле представляют собой возведенные в квадрат оценки доли ковариации. Независимо от сути интерпретации, чем выше доля, тем больше пара осей, о которой идет речь, объясняет значительную часть ковариации между наборами данных.

Примечание 2: доля ковариации всегда располагается в порядке убывания (т. е. пара осей 1 объясняет больше ковариации, чем пара осей 2, которая объясняет больше, чем пара осей 3...).

Примечание 3: нет абсолютного правила, сколько осей следует выделить для интерпретации. Это всегда компромисс между хорошим обобщением информации, содержащейся в исходных данных (которое увеличивается с количеством осей) и легкостью для интерпретации (которая уменьшается с количеством осей).

Графические представления

В 2B-PLS есть две возможности представления для каждой таблицы: диаграмма ординации объектов, которые показывают положение этих объектов на факториальной плоскости, состоящей из двух осей, и круг корреляций (см. лист **89**), который позволяет интерпретировать распределение объектов.

График ординации объектов. Для представления ординационной диаграммы для первой таблицы используется функция : `MVA.plot(PLS2B, space=1)*`, а для второй таблицы – `MVA.plot(PLS2B, space=2)*`. По умолчанию представлены оси 1-й (по горизонтали) и 2-й (по вертикали) главных компонент. Они могут быть изменены с помощью аргументов `xax` и `yax`. См `?MVA.scoreplot` для многих других графических опций.

Корреляционный круг. Строится функцией `MVA.plot(PLS2B, "corr", space=1)*` для первой таблицы и `MVA.plot(PLS2B, "corr", space=2)*` для второй соответственно. Чтобы

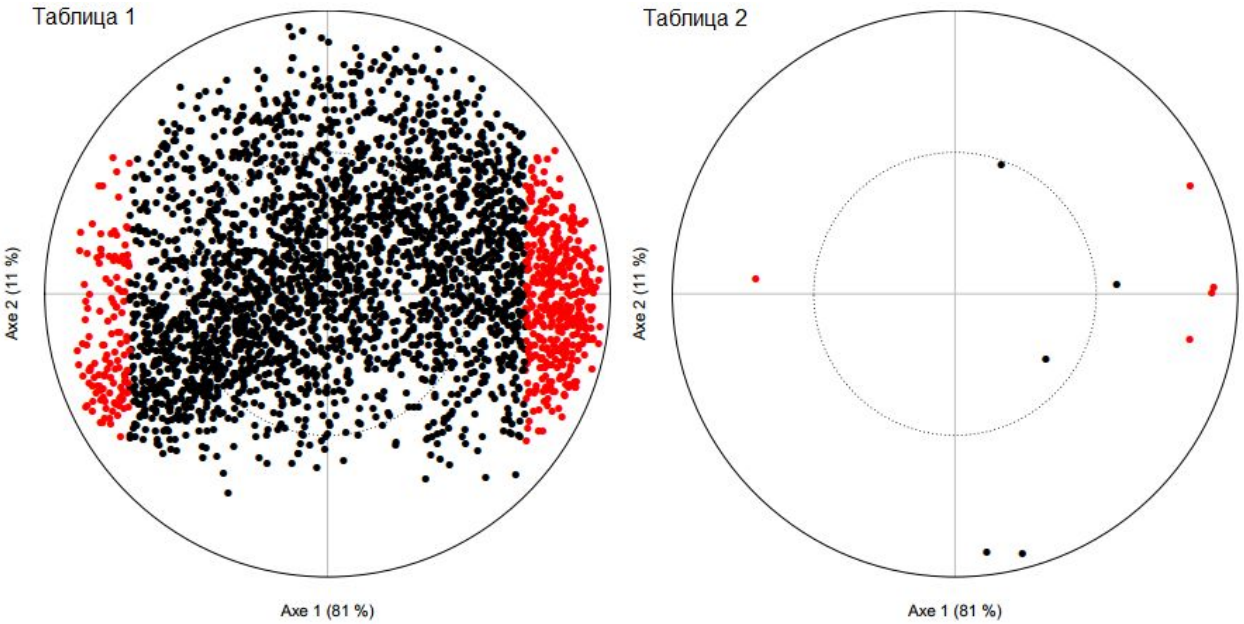
удалить стрелки, добавьте аргумент `arrows=FALSE`. См. `?MVA.corplot` для многих других графических опций.

Интерпретация

Цель двублочного PLS-анализа – это только оценить связи между переменными внутри каждого массива (в этом качестве интерпретация основывается главным образом на кругах корреляции), а, прежде всего, выделить ассоциации между переменными двух различных таблиц.

Объясненные доли ковариации указывают, на каких осях следует сосредоточиться при интерпретации. В наиболее распространенных случаях одной пары осей достаточно, чтобы объяснить большую часть ковариации между двумя массивами. В этом случае для каждой таблицы указываются переменные, наиболее коррелирующие с первой осью (см. п. **89**). Именно эти переменные наиболее связаны.

Для одной таблицы подобная интерпретация обычно проста: переменные на противоположных концах оси отрицательно коррелируют (см. п. **89**). Однако для взаимосвязи между переменными разных таблиц отсутствует существенная информация: в каком именно направлении коррелируют эти пары осей. Разложение корреляции по осям можно получить функцией `MVA.synt()`* во втором блоке таблиц, который она возвращает. Если корреляция для первой пары осей положительна, то переменные, направленные слева от оси 1 для первой таблицы, положительно связаны с переменными, направленными слева от оси 1 для второй таблицы. Если корреляция для первой пары осей отрицательна, то эта закономерность меняется на противоположную.



112. Прокрустовый анализ

Подготовка данных

Прокустовый анализ интересен, когда он выполняется для двух таблиц с двумя столбцами в каждой (см. п. **114** для случая, когда, по крайней мере, одна таблица содержит более двух столбцов). Это могут быть два исходных массива с двумя переменными каждый или две оси предыдущей ординации (см. п. **90** для получения координат отдельных объектов на этих осях). Таким образом, одним из распространенных применений прокрустова анализа является, например, сравнение результатов двух ординаций, выполненных с использованием матрицы расстояний, в частности двух nMDS (см. п. **102**).

Проведение анализа

Для выполнения прокустового анализа, можно использовать функцию из пакета `vegan`: `Proc <- procrustes(таблица1, таблица2)`, где `таблица1` и `таблица2` являются двумя таблицами.

Примечание: первая и/или вторая из этих двух таблиц может быть также непосредственно ординацией, если она была выполнена в пакете `vegan`.

Оценка качества анализа

Можно оценить уровень "корреляции" между данными двух таблиц с использованием теста из пакета `vegan`: `protest(таблица1, таблица2)`. Функция возвращает статистику, похожую на коэффициент корреляции (*Correlation in a symmetric Procrustes rotation*), и результат непараметрического теста, называемого PROTEST (на основе коэффициента корреляции). Превышение *p*-значения над критическим уровнем указывает на то, что между двумя таблицами существует значимое совпадение.

Графическая интерпретация

Единственным возможным представлением в прокустовом анализе является диаграмма ординации объектов, которая показывает положение отдельных объектов как для `таблица1`, так и для `таблица2` на факториальной плоскости, состоящей из двух осей. Эти две оси соответствуют переменным в `таблица1`.

Чтобы построить график, используйте функцию: `MVA.plot(Proc,"pairs")*`. Каждый объект представлен стрелкой, которая начинается из точки с координатами объекта из таблицы 1 и заканчивается в точке с координатами этого же объекта из таблицы 2. (График ординации таблицы 2 предварительно поворачивается, перемещается и/или масштабируется, чтобы наилучшим образом вписаться в график таблицы 1). См. `?MVA.pairplot` для описания многих графических опций.

Интерпретация

Чем короче стрелки на графике ординации объектов, тем больше это указывает на возможное совпадение между двумя таблицами (т. е. объекты примерно расположены в одном и том же месте, независимо от массива). С другой стороны, чем длиннее стрелки, тем больше это указывает на слабое совпадение между двумя таблицами. Всегда наблюдается согласованность между графиком и результатом статистического теста. Длина стрелок никогда не одинакова для всех объектов; таким образом, можно определить, какие из них "похожи" в обеих таблицах или, наоборот, очень разные.

113. Совместный инерционный анализ (CIA)

Англ. – *Co-inertia analysis* (CIA)

Подготовка данных

Совместный инерционный (ко-инерционный) анализ CIA – это наиболее общий метод оценки сопряженности двух таблиц. Последние могут иметь различный тип (например, классическую таблицу переменных и таблицу сопряженности) или содержать переменные, измеренные в различных шкалах (количественных и/или качественных).

Первым шагом в анализе является выполнение отдельной ординации для обеих таблиц. В зависимости от характера переменных чаще всего используются PCA (см. п. **96**), анализ соответствий CA и MCA (см. п. **97-98**) или смешанный анализ (см. п. **99**). Оба анализа не обязательно должны быть одинаковыми.

Примечание 1: CIA требует, чтобы веса отдельных строк обеих таблиц были равны между собой. С PCA, MCA и смешанным анализом этих проблем нет, так как, если не будет явного специального вмешательства, то все строки имеют одинаковый вес. Для CA ситуация отличается, потому что в этом анализе вес строки зависит от суммы значений этой строки. Чтобы связать CA с другой ординацией при выполнении CIA, необходимо откорректировать эту вторую ординацию, указав в качестве весов строк те, которые используются в CA. Эта операция может быть выполнена автоматически с помощью `анализС <- ord.rw(анализ, CA)`*, где `анализ` является ординацией, сопрягаемой с `CA`, а результирующая ординация `анализС` уже может использоваться в CIA.

Примечание 2: по причине, объясненной в предыдущем замечании, невозможно связать в CIA две ординации, выполненных анализом соответствий CA.

Проведение анализа

Для осуществления совместного инерционного анализа можно использовать функцию из пакета `ade4`:
`CIA <- coinertia(анализ1, анализ2, scannf=FALSE,nf=10)`, где `анализ1` и `анализ2` – ординации, проведенные отдельно по каждой таблице.

Функция `coinertia()` работает с ординациями, выполняемыми с помощью функций пакета `ade4`. Это верно для MCA (см. п. **98**) и смешанного анализа (см. п. **99**), но не для PCA (см. п. **96**) и CA (см. п. **97**), которые обычно выполняются с помощью пакета `vegan`. Чтобы преобразовать результат одной из этих ординаций в объект, используемый функцией `coinertia()`, можно воспользоваться функцией:
`анализС <- to.dudi(анализ)*`, где `анализ` – имя ординации PCA или CA. Если использовалась функция `ord.rw()*`, то возвращаемая ей ординация уже напрямую совместима с функцией `coinertia()`.

ПРИМЕРЫ

В CIA требуется выполнить сопряжение ординаций PCA (`PCA`) и CA (`CA`), и обе из них были созданы с помощью пакета `vegan`, как описано в пп. **96** и **97**. Сначала мы корректируем `PCA`, назначая ее строкам веса, используемые `CA` :

```
> PCA.c <- ord.rw(PCA, CA)
```

Объект `PCA.c` непосредственно совместим с функцией `coinertia()`. Однако это не относится к объекту `CA`. Мы преобразуем его в нужный формат :

```
> CA.c <- to.dudi(CA)
```

Теперь можно осуществить CIA:

```
> CIA <- coinertia(PCA.c, CA.c, scannf=FALSE, nf=10) .
```

Оценка качества анализа

Тесты внешних переменных. Цель совместного инерционного анализа состоит в том, чтобы наилучшим образом сформировать некоторую информационную структуру, которая оценивает ко-инерцию между двумя наборами данных (т.е. часть их общей вариации, которая изменяется синхронно). Но для того, чтобы интерпретация анализа была актуальной, нужно, чтобы оценка ко-инерции была значимой. Значимость ко-инерции проверяется с использованием коэффициента RV, вычисляемого функцией `randtest(CIA)*`, которая реализует перестановочный тест. Статистика RV изменяется в диапазоне от 0 (ко-инерция отсутствует) до 1 (полная ко-инерция). Если *p*-величина оказывается значимой, результаты CIA могут быть интерпретированы. В противном случае объяснение не имеет смысла, поскольку между этими двумя таблицами нет существенной связи.

Примечание: если одна из двух ординаций, сопрягаемых CIA, является CA, необходимо добавить аргумент `fixed=NB`, где `NB` равен 1, если CA – первый аргумент `coinertia()`, или 2, если CA – второй аргумент.

Обобщенная оценка. Если совместная инерция между двумя массивами является значимой, считается, что качество анализа оценивается по доле ко-инерции, объясненной каждой парой осей. Эти доли можно получить с помощью `MVA.synt(CIA)*` в первом блоке таблиц, возвращаемых функцией.

Примечание 1 : мы говорим в CIA о парах осей, потому что анализ использует две отдельных ординационных модели, одну для первого массива, а другую для второго. Таким образом, у нас есть ось 1 для первой таблицы и ось 1 для второй таблицы (и так далее).

Примечание 2: доля совместной инерции всегда располагается в порядке убывания (т. е. пара осей 1 объясняет больше ко-инерции, чем пара осей 2, которая объясняет больше, чем пара осей 3...).

Примечание 3: нет абсолютного правила, сколько осей следует выделить для интерпретации. Это всегда компромисс между хорошим обобщением информации, содержащейся в исходных данных (которое увеличивается с количеством осей) и легкостью для интерпретации (которая уменьшается с количеством осей).

Графические представления

В CIA есть два возможных представления для каждой таблицы: диаграмма ординации объектов, которая показывает положение ее объектов на факториальной плоскости, состоящей из двух осей, и круг корреляций (см. п. **89**), который позволяет интерпретировать распределение этих объектов.

Примечание: для MCA доступен только график ординации объектов (см. п. **98**).

График ординации объектов. Чтобы представить ординационную диаграмму для первой таблицы используется функция : `MVA.plot(CIA, space=1)*`, а для второй таблицы – `MVA.plot(CIA,space=2)*`. По умолчанию представлены оси 1-й (по горизонтали) и 2-й (по вертикали) главных компонент. Они могут быть изменены с помощью аргументов `xax` и `yax`.

Чтобы добавить на график разделение объектов на группы, используйте аргумент `fac=фактор`, где `фактор` определяет принадлежность каждого объекта к группе. Аргументы `col`, `pch`, `fac.lab`, `contours`, `stars` и `barycenters` позволяют настроить эстетику изображения.

Корреляционный круг. Строится функцией `MVA.plot(CIA,"corr",space=1)*` для первой и `MVA.plot(CIA,"corr",space=2)` для второй таблицы соответственно. Чтобы удалить стрелки, добавьте аргумент `arrows=FALSE`. См. `?MVA.corplot` для многих других графических опций.

Интерпретация

Цель CIA – выделить связи между переменными двух различных таблиц. Для количественных переменных интерпретация основана на сравнении двух кругов корреляций: две стрелки, указывающие в одном направлении или в диаметрально противоположных направлениях, указывают на две коррелированные переменные (см. п. **89**). Разложение корреляции по осям можно получить функцией `MVA.synt()*` во втором блоке таблиц, который она возвращает. Если корреляция для первой пары осей положительна, переменные, направленные слева от оси 1 для первой таблицы, положительно связаны с переменными, направленными слева от оси 1 для второй таблицы. Если корреляция для первой пары осей отрицательная, то эта закономерность меняется на противоположную.

Для качественных переменных интерпретация похожа на MCA: чем ближе два подмножества точек, тем больше связаны группирующие факторы (см. п. **98**). Опять же, величина связи зависит от корреляции между парами осей.

Отношения между количественными и качественными переменными интерпретируются как смешанный анализ: чем ближе центр тяжести группы расположен к концу стрелки (при проецировании перпендикуляра к этой стрелке), тем выше среднее значение количественной переменной, представленной стрелкой, для объектов этой группы, (и наоборот, см. п. **89**).

Примечание: связи между переменными в одном массиве интерпретируются как PCA (см. п. **96**), MCA (см. п. **98**) и смешанный анализ (см. п. **99**).

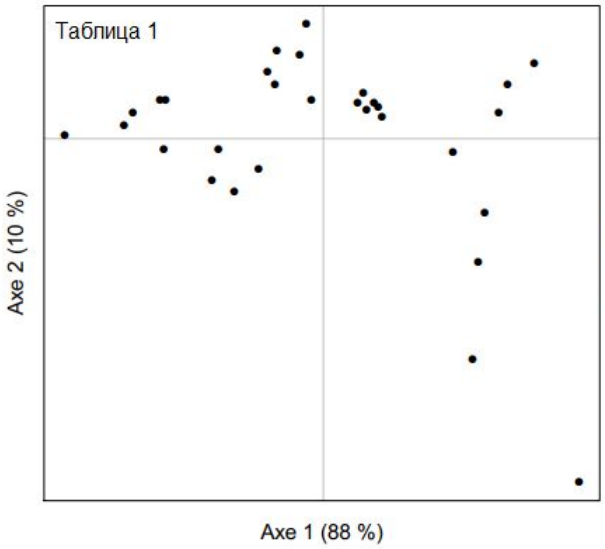


Таблица 1
Axe 2 (10 %)
Axe 1 (88 %)

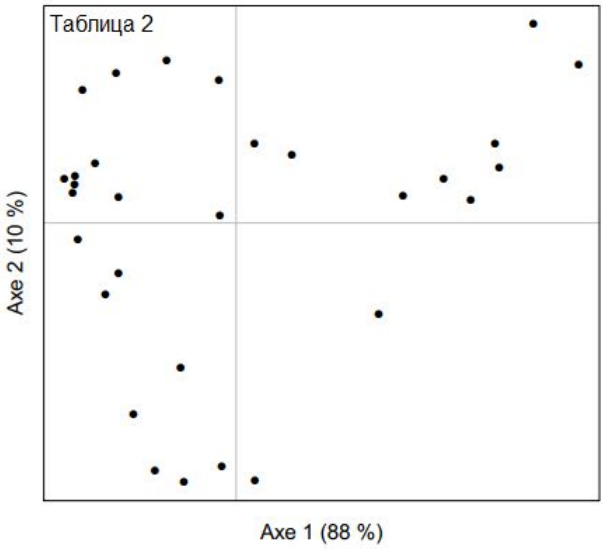
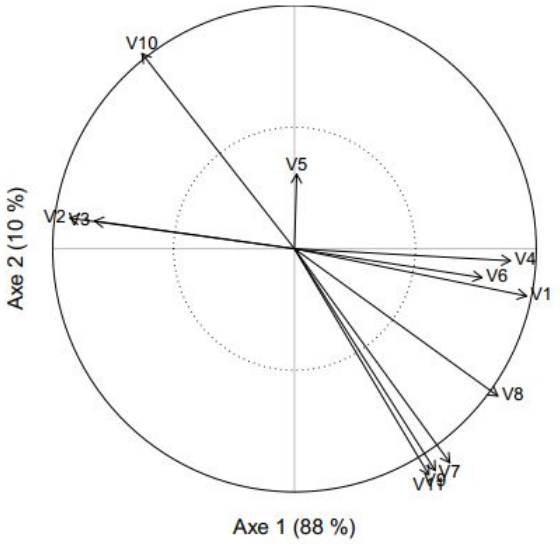
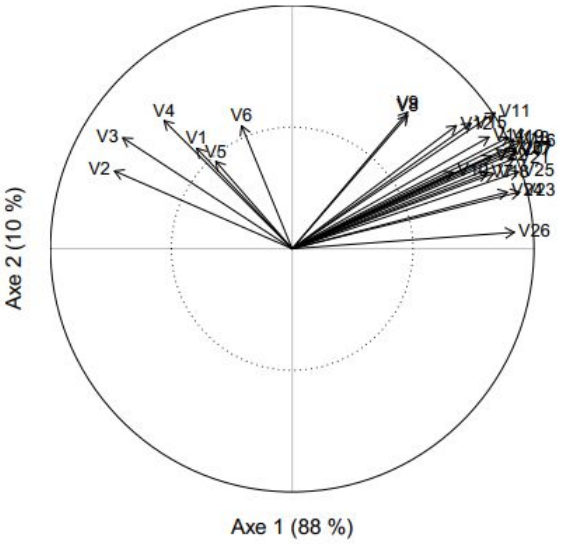


Таблица 2
Axe 2 (10 %)
Axe 1 (88 %)


V10
V5
V2
V3
V4
V6
V1
V8
V7
Axe 2 (10 %)
Axe 1 (88 %)


V4
V6
V3
V1
V5
V2
V11
V25
V26
Axe 2 (10 %)
Axe 1 (88 %)

114. Прокрустовый совместный инерционный анализ (PCIA)

Англ. – *Procustean co-inertia analysis* (PCIA)

Подготовка данных

PCIA обобщает прокрустовый анализ (см. см. п. **112**), если, по крайней мере, одна из сопоставляемых таблиц содержит более двух столбцов. Обе таблицы могут быть исходными массивами переменных или осями предыдущей ординации (см. п. **90** для получения координат отдельных объектов на этих осях). Все переменные должны быть количественными, и часто выгодна стандартизация этих переменных (см. п. **88**).

Проведение анализа

Для осуществления прокрустового совместного инерционного анализа используют функцию из пакета `ade4`: `PCIA <- procuste(таблица1, таблица2)`, где `таблица1` и `таблица2` – таблицы или ординации, проведенные отдельно по каждой таблице.

Оценка качества анализа

Можно оценить уровень "соответствия" между двумя таблицами с использованием теста из пакета `ade4`: `randtest(PCIA)`. Функция возвращает статистику, называемую m^2 (`Observation`) , которая изменяется от 0 (идеальное совпадение) до 1 (никакого совпадения), и результат непараметрического теста, называемого PROTEST (на основе m^2). Превышение *p*-значения над критическим уровнем указывает на то, что между двумя таблицами существует значимое совпадение.

Графические представления

В PCIA есть два возможных представления: диаграммы ординации объектов, которые показывают положение соответствующих объектов с координатами из каждой исходной таблицы на факториальной плоскости, состоящей из двух осей, и круг корреляций (см. п. **89**), который позволяет интерпретировать распределение этих объектов.

Примечание: в PCIA каждый объект имеет две координаты в одной и той же факториальной плоскости, одна для `таблица1`, а другая для `таблица2`. Таким образом, в одном и том же пространстве формируются два облака точек.

График ординации объектов. Для представления ординационной диаграммы используется функция : `MVA.plot(PCIA, "pairs")*`. Каждый объект представлен стрелкой. Она начинается из точки с координатами объекта из таблицы 1 и заканчивается в точке с координатами этого же объекта из таблицы 2. См. `?MVA.pairplot` для описания многих графических опций.

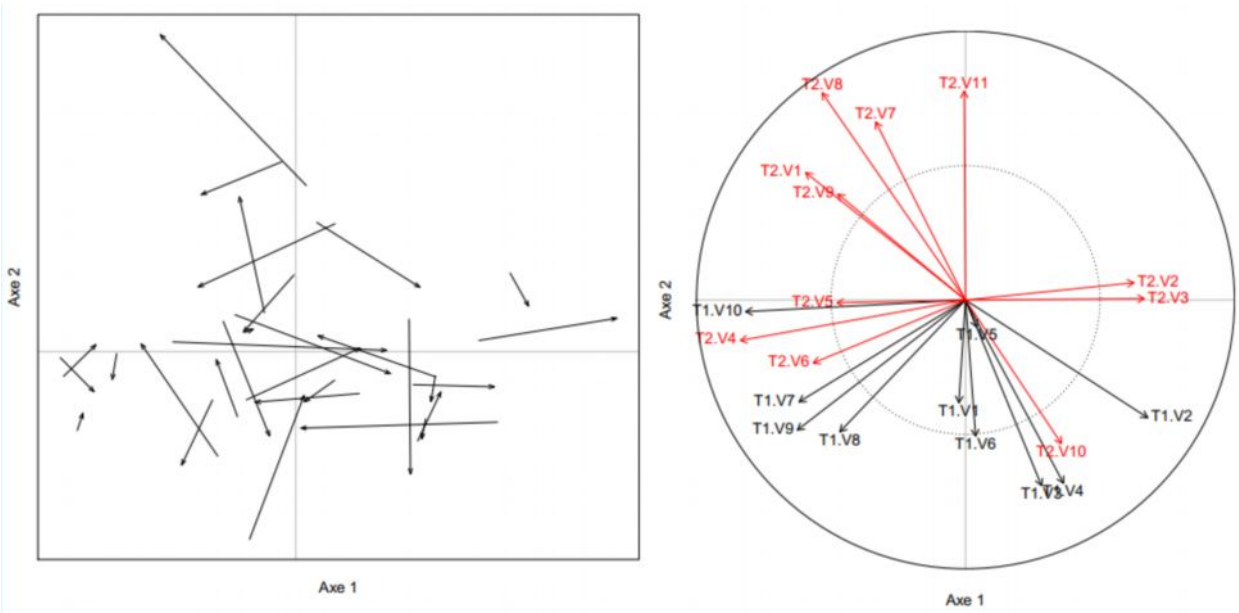
Корреляционный круг. Строится функцией `MVA.plot(PCIA,"corr")*`. По умолчанию показываются переменные из обеих таблиц. Чтобы показать переменные только из одной таблицы, используйте аргументы `space=1` и `space=2` для `таблица1` и `таблица2` соответственно.

Чтобы удалить стрелки, добавьте аргумент `arrows=FALSE`. Чтобы по-разному представлять переменные различных массивов, добавьте аргумент `fac=фактор`, где `фактор` является фактором, определяющим группу каждой переменной. Аргументы `col`, `pch`, `fac.lab`, `contours`, `stars` и `barycenters` позволяют настроить эстетику изображения. См. `?MVA.corplot` для многих других графических опций.

Интерпретация

Чем короче стрелки на ординационной диаграмме объектов, тем больше это указывает на возможное совпадение между двумя таблицами (т. е. объекты примерно расположены в одном и том же месте, независимо от массива). С другой стороны, чем длиннее стрелки, тем больше это указывает на слабое совпадение между двумя таблицами. Всегда наблюдается согласованность между графиком и результатом статистического теста. Длина стрелок никогда не одинакова для всех объектов; таким образом, можно определить, какие из них "похожи" в обеих таблицах или, наоборот, очень разные..

При условии, что совпадение между двумя таблицами является значимым, круг корреляции позволяет выделить переменные, которые определяют эти расхождения. Его интерпретация идентична кругу классических корреляций (см. п. **89**).



Симметричный анализ более чем двух таблиц

Выбор анализа зависит от цели исследования:

1. Если цель состоит в том, чтобы выделить общую информацию для нескольких таблиц → используйте *обобщенный канонический корреляционный анализ с регуляризацией* (RGCCA) .
2. Если цель состоит в том, чтобы проверить соответствие между несколькими таблицами и определить консенсусную конфигурацию, т. е. « среднюю» (уникальную) ординацию для всех этих массивов → используйте *обобщенный прокрустовый анализ*.

115. Обобщенный канонический корреляционный анализ с регуляризацией (RGCCA)

Англ. – *Regularized generalized canonical correlation analysis* (RGCCA)

Подготовка данных

Обобщенный канонический корреляционный анализ с регуляризацией RGCCA требует, чтобы все переменные были количественными. Это могут быть исходные переменные, реально наблюдаемые или измеренные, и/или оси предыдущей ординации (см. п. **90** для получения координат отдельных объектов в последнем случае). Различные массивы (называемые "блоками") не обязательно должны содержать одинаковое количество переменных.

Анализ работает корректнее, если переменные имеют приблизительно нормальное (по крайней мере, симметричное) распределение и в каждом массиве связаны между собой линейными отношениями. Предварительное преобразование данных может в значительной степени помочь улучшить ситуацию (см. п. **88**). Также рекомендуется, как правило, стандартизировать переменные перед анализом (см. п. **88**), что придает одинаковый вес всем переменным.

Примечание: фактор можно интегрировать в обработку данных, преобразовав его в индикаторные переменные функцией `var.ind <- dummy(фактор)*`, где `фактор` – вектор, определяющий группу каждого объекта. Таблица `var.ind` затем обрабатывается так же, как и другие переменные, за исключением преобразований, которые для индикаторных переменных никогда не требуются. Этот способ позволяет использовать RGCCA в качестве дискриминантного анализа.

Все таблицы должны быть включены в единый список. Для этого выполните команду: `блоки <- list(блок1=таблица1, блок2=таблица2...)`, где `таблица1`, `таблица2` и т.д. являются отдельными таблицами (которые в дальнейшем анализе будут называться `блок1`, `блок2`...). Каждый отсек списка обязательно должен иметь уникальное имя.

Наконец, перед анализом необходимо определить взаимосвязи между блоками (т.е. таблицы, которые необходимо связать между собой). Ничто не требует, чтобы анализ искал общую информацию во всех возможных парах таблиц "один-к-одному". Напротив, лучше определить отношения между этими парами, которые наиболее актуальны для проверки биологических гипотез. Эти связи определяются в симметричной матрице (*design matrix*), в которой блоки расположены как в строках, так и столбцах, а сама таблица заполнена 0 (нет отношения) и 1 (оценить отношение). Диагональ обязательно должна состоят только из 0.

ПРИМЕР

Блоки 1 и 2 должны объединяться попарно с блоком 3, но не между собой :

```
> matrice <- matrix(c(0,0,1,0,0,1,1,1,0), nrow=3)
> matrice
      [,1] [,2] [,3]
[1,]    0    0    1
[2,]    0    0    1
[3,]    1    1    0
```

Проведение анализа

Для выполнения RGCCA используется функция из пакета `mix0mics` : `RGCCA<-wrapper.rgcca(блоки, C=дизайн, tau="optimal", ncomp=nb)`, где `дизайн` – матрица связи таблиц и `nb` – число осей для построения.

Примечание 1: Если аргумент `C` не указывается, то рассматриваются все возможные пары двух массивов.

Примечание 2: по умолчанию функция стандартизирует все наборы данных, что часто полезно, поскольку это дает одинаковый вес всем переменным (см. п. **88**). Чтобы не стандартизировать, добавьте аргумент `scale=FALSE`.

Примечание 3: RGCCA создает столько отдельных ординаций, сколько массивов. Таким образом, у нас есть ось 1 для каждого блока, ось 2 для каждого блока и так далее. Если аргумент `ncomp` не указан, для каждого блока строится только одна ось.

Примечание 4: цель RGCCA состоит в том, чтобы максимизировать некоторую информацию, которая связана с двумя разными сущностями: (а) общей объясненной межблочной дисперсией и (б) корреляцией между осями одного и того же уровня (т. е. осями 1, осями 2...). Аргумент `tau` позволяет уточнить, как необходимо искать необходимый для исследователя оптимум. Таким образом, можно найти:

- максимум общей информации о каждом блоке, объясненной анализом ("оптимум дисперсии"), задав `tau=rep(1, nb)`, где `nb` – количество блоков;
- максимум корреляции между осями одного и того же уровня ("оптимум корреляции"), задав `tau=rep(0, nb);`
- или лучший компромисс между этими двумя сущностями `(tau="optimal"),` что задается по умолчанию.

Оценка качества анализа

Качество анализа оценивается в долях от общей внутриблочной дисперсии, объясняемой каждой осью каждого блока, и корреляцией между осями, которые связаны между собой в соответствии с матрицей дизайна. Все эти значения могут быть получены с помощью функции `MVA.synt(RGCCA)*` в разных таблицах, возвращаемых функцией (по одной на каждый критерий и на каждый блок).

Примечание: в отличие от большинства многомерных методов, объясненные доли дисперсии не обязательно располагаются в порядке убывания (хотя это чаще всего происходит). Корреляция между осями всегда становятся все сильнее и сильнее.

Графические представления

В RGCCA есть два возможных представления для каждого блока: диаграммы ординации объектов, которые показывают положение соответствующих объектов с координатами из каждой исходной таблицы на факториальной плоскости, состоящей из двух осей, и круг корреляций (см. п. **89**), который позволяет интерпретировать распределение этих объектов.

График ординации объектов. Чтобы представить ординационную диаграмму для первой таблицы используется функция : `MVA.plot(RGCCA, space=1)*`, для второй таблицы – `MVA.plot(RGCCA,space=2)*` и т.д. По умолчанию представлены оси 1-й (по горизонтали) и 2-й (по вертикали) главных компонент. Они могут быть изменены с помощью аргументов `xax` и `yax`.

Корреляционный круг. Строится функцией `MVA.plot(RGCCA,"corr",space=1)*` для первого блока и `MVA.plot(RGCCA,"corr",space=2)*` для второго блока и так далее. Чтобы удалить стрелки, добавьте аргумент `arrows=FALSE`. См. `?MVA.corplot` для многих других графических опций.

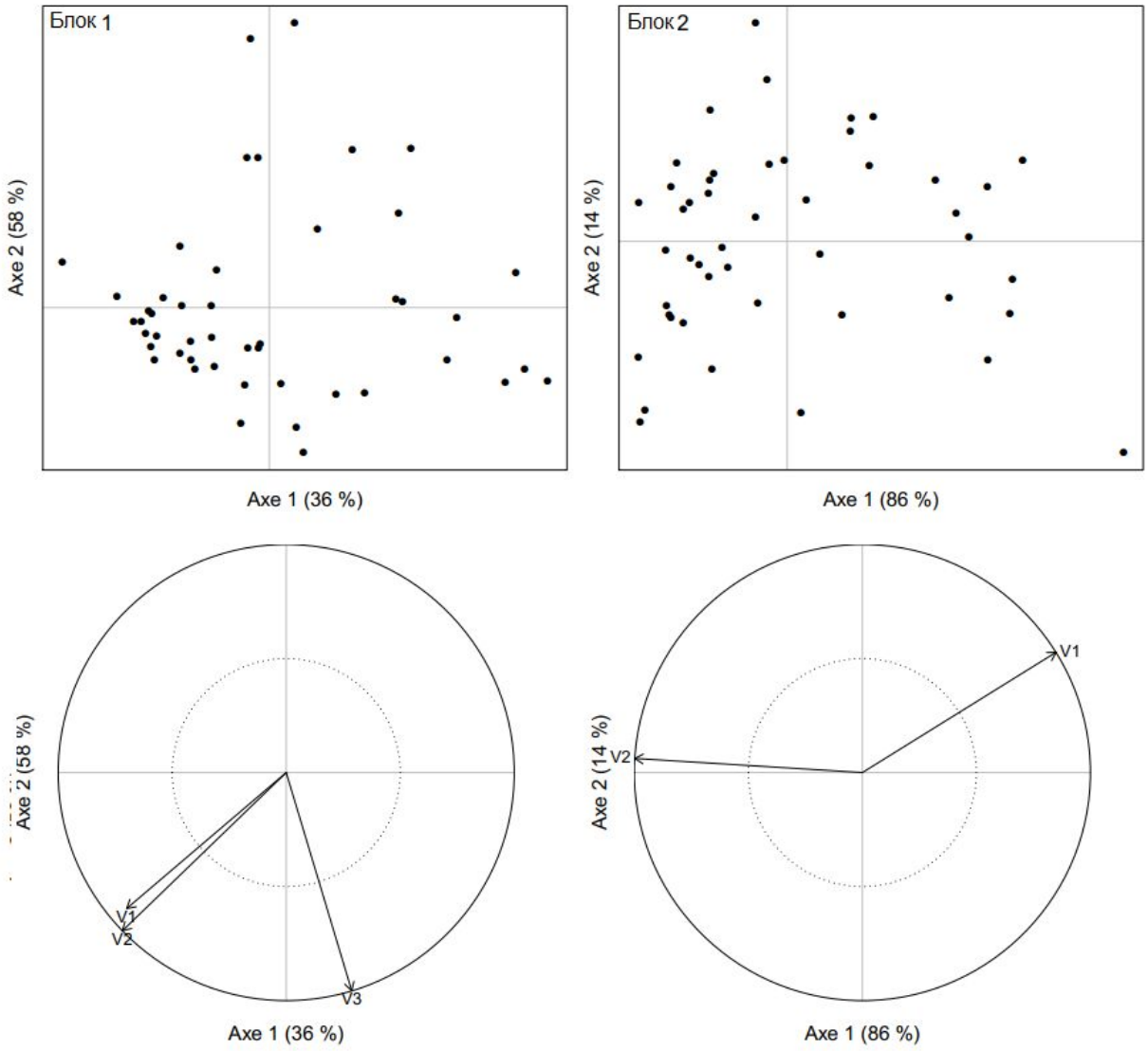
Интерпретация

Цель RGCCA чаще всего заключается не только в том, чтобы определить переменные каждой таблицы, которые связаны между собой, но и (прежде всего) ассоциации между переменными из разных таблиц. В этом качестве интерпретация

основывается главным образом на кругах корреляции.

Межблочные корреляции (т. е. между осями одного и того же уровня) указывают на то, какие оси наиболее важны для интерпретации. Именно эти высококоррелированные оси содержат максимум информации, общей для нескольких массивов, что и является целью анализа и, следовательно, и основой интерпретации. Если, например, корреляция является сильной для осей 1, то для каждой таблицы будут указаны переменные, наиболее коррелированные с этой первой осью (см. п. **89**). Именно эти переменные наиболее связаны между собой.

Для одной таблицы интерпретация проста: переменные на противоположных концах оси отрицательно коррелируют (см. п. **89**). Для оценки направленности взаимосвязи между переменными двух разных массивов также есть логичные правила. Если корреляция для первой пары осей положительна, то переменные, направленные слева от оси 1 для первой таблицы, положительно связаны с переменными, направленными слева от оси 1 для второй таблицы. Если корреляция для первой пары осей отрицательная, то эта закономерность меняется на противоположную.



116. Обобщенный прокрустовый анализ (GPA)

Англ. – *Generalized Procrustes analysis* (GPA)

Подготовка данных

Обобщенный прокрустовый анализ GPA требует, чтобы все переменные были количественными. Это могут быть исходные переменные, реально наблюдаемые или измеренные, и/или оси предыдущей ординации (см. п. **90** для получения координат отдельных объектов в последнем случае). Различные массивы не обязательно должны содержать одинаковое количество переменных.

Анализ работает корректнее, если переменные имеют приблизительно нормальное (по крайней мере, симметричное) распределение и в каждом массиве связаны между собой линейными отношениями. Предварительное преобразование данных может в значительной степени помочь улучшить ситуацию (см. п. **88**).

Наконец, все частные таблицы должны быть объединены в один массив. Для этого подайте команду: `таблица <- data.frame(таблица1, таблица2...)` , где `таблица1`, `таблица2` и т.д. являются различными массивами, которые расположатся слева направо в окончательной таблице.

Проведение анализа

Чтобы реализовать GPA, используется функция из пакета `FactoMineR`:
`GPA <- GPA(таблица, group=столбцы, plot=FALSE)`, где `столбцы` – вектор, определяющий количество столбцов в каждой таблице (слева направо), чтобы функция могла разделить их внутри. Чтобы дать имя каждому массиву (что может облегчить последующую графическую интерпретацию), добавьте аргумент `name.group=имена`, где `имена` – вектор, содержащий имя каждого массива (слева направо). По умолчанию они называются `group.1`, `group.2`...

Примечание: по умолчанию функция стандартизирует все наборы данных, что часто полезно, поскольку это придает одинаковый вес всем переменным (см. п. **88**). Чтобы не стандартизировать, добавьте аргумент `scale=FALSE`.

Оценка качества анализа

Тест. Цель анализа GPA состоит в том, чтобы наилучшим образом синтезировать консенсусную конфигурацию, т.е. "среднюю" из ординаций, соответствующих каждой таблице. Для этого ординации предварительно проходят шаги стандартизации положения, масштаба и поворота координатных осей, чтобы наилучшим образом обеспечить их сопоставление друг с другом. Часто именно эта консенсусная конфигурация является целью анализа для интерпретации или использования в последующем анализе (см. п. **90** для получения координат).

Примечание: первый шаг в GPA на самом деле состоит в том, чтобы сделать анализ главных компонент для каждой таблицы, поэтому существует требование, чтобы переменные были количественными (см. п. **96**). Это те самые ординации PCA, которые затем подвергаются прокрустовым преобразованиям.

Можно проверить, является ли консенсусная конфигурация статистически значимой, т. е. есть ли на самом деле соответствие между различными массивами. Для выполнения теста используется функция:
`GPA.test(таблица, group=столбцы)*`. Время расчета, необходимое для выполнения этого теста, относительно велико.

Обобщенная оценка. Если уровень совпадения различных таблиц является значимым, то для оценки качества анализа используется ряд показателей. Механизм того, как GPA определяет консенсусную конфигурацию, заключается в получении максимума дисперсии, общей для всех массивов. В связи с этим возникает вопрос о доле общей дисперсии, получаемой в целом, а именно для каждой оси этого консенсуса. Эти доли можно получить с помощью `MVA.synt(GPA)*` в первых двух таблицах, возвращаемых функцией. Третий возвращаемый массив отличается от второго, поскольку он дает доли консенсусной дисперсии, объясненные каждой осью. Обычно вместо этого используются доли от общей дисперсии.

Еще один способ оценки качества анализа заключается в том, чтобы определить, представлены ли консенсусной конфигурацией все исходные таблицы. Для этого рассматривается остаточная дисперсия, т. е. дисперсия, не объясненная консенсусной конфигурацией. Если обобщение является эффективным для всех таблиц, то эта остаточная дисперсия должна равномерно распределяться между ними. Если одна или несколько таблиц имеют гораздо более высокий процент остаточной дисперсии, чем другие таблицы, то это означает, что консенсусная конфигурация не может хорошо их представлять. Здесь, однако, важно учитывать долю общей консенсусной дисперсии и, если она очень высока, то можно считать, что все таблицы довольно хорошо представлены, хотя некоторые из них менее хорошо, чем другие. Распределение остаточной дисперсии предоставляется в последнем массиве, возвращаемом функцией `MVA.synt()`.

Графические представления

В GPA есть два возможных представления: диаграмма ординации объектов, которая показывает положение объектов для консенсусной конфигурации или каждой из исходных таблиц на факториальной плоскости, состоящей из двух осей, и круг корреляций (см. п. **89**), который позволяет интерпретировать распределение объектов.

График ординации объектов. Для представления ординационной диаграммы, показывающей консенсусную конфигурацию, используется функция: `MVA.plot(GPA)*`. По умолчанию представлены 1-е (по горизонтали) и 2-е (по вертикали) главные оси. Они могут быть изменены с помощью аргументов `xax` и `yax`.

Чтобы добавить на график разделение объектов на группы, используйте аргумент `fac=фактор`, где `фактор` определяет принадлежность каждого объекта к группе. Аргументы `col`, `pch`, `fac.lab`, `contours`, `stars` и `barycenters` позволяют настроить эстетику изображения.

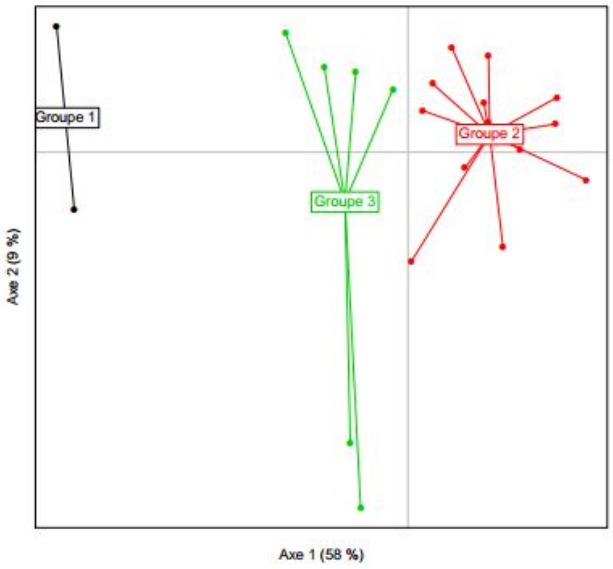
Чтобы представить отдельные объекты во всех частных ординациях, подайте команду: `plot(GPA)`. Точки одного и того же объекта будут связаны с точкой объекта в консенсусной конфигурации. Таким образом, можно будет определить объекты, которые "похожи" в различных таблицах, или, наоборот, очень различаются (поэтому консенсус здесь, вероятно, наименее актуален).

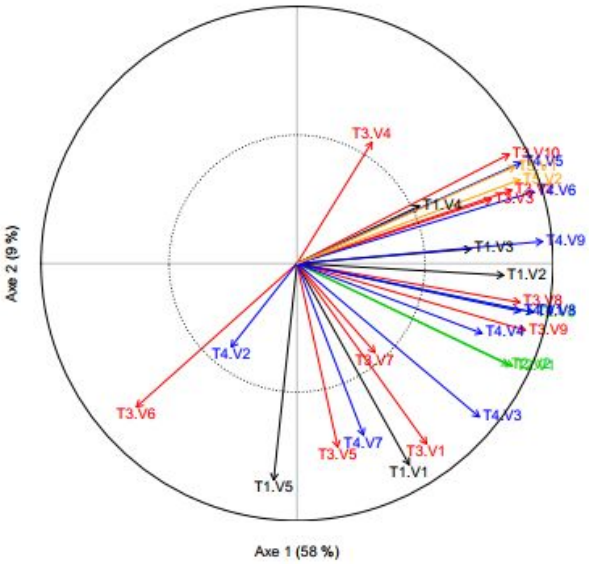
Корреляционный круг. Строится функцией `MVA.plot(GPA,"corr")*`. Чтобы удалить стрелки, добавьте аргумент `arrows=FALSE`.

Чтобы по-разному представлять переменные различных массивов, добавьте аргумент `fac=фактор`, где `фактор` является фактором, определяющим группу (т. е. таблицу) каждой переменной. Аргументы `col`, `pch`, `fac.lab`, `contours`, `stars` и `barycenters` позволяют настроить эстетику изображения. См. `?MVA.corplot` для многих других графических опций.

Интерпретация

Если цель анализа заключается в истолковании консенсусной конфигурации, то ее интерпретации идентична PCA (см. п. **96**).





КОММЕНТАРИИ*

1. М.Эрве в своем путеводителе предлагает осуществлять анализ данных на основе свободно распространяемой системы R (www.r-project.org/). Этот программный продукт является наиболее полной, надежной и динамично развивающейся статистической средой, объединяющей язык программирования высокого уровня и мощные библиотеки программных модулей вычислительной и графической обработки. Сегодня статистическая среда R является безусловным лидером среди некоммерческих систем статистического анализа и постепенно становится незаменимой при проведении научно-технических расчетов в большинстве западных университетских центров и многих ведущих фирмах.

До недавнего времени основным препятствием для русскоязычных пользователей при освоении R являлось представление почти всей ее документации на английском языке. В последние годы эта проблема потеряла актуальность в связи с появлением достаточно полных пособий, как отечественных авторов, так и переводов лучших зарубежных монографий [1-8]. Интересующийся читатель может легко найти эти книги, а также другую необходимую ему литературу из списка на нашем сайте stok1946.blogspot.com.

6. Для того, чтобы подробнее проиллюстрировать некоторые алгоритмы, представленные в путеводителе, мы используем в качестве примера для дальнейших упражнений таблицу `algae`, включенную в пакет `DMwR` и содержащую данные гидробиологических исследований обилия водорослей в различных реках.

Каждое из 200 наблюдений содержит информацию о 18-ти переменных, в том числе:

- три номинальных переменных, описывающих время года, сопряженное с моментом взятия проб, тип и скорость течения реки;
- 8 переменных, составляющих комплекс наблюдаемых гидрохимических показателей (pH, содержание кислорода, хлорофилла и различных анионов);
- среднюю численность каждой из 7-ми групп водорослей (видовой состав не идентифицировался).

Выполним загрузку данных:

```
library(DMwR)
data(algae)
```

Проверим наличие строк с пропущенными значениями

```
nrow(algae[!complete.cases(algae),])
[1] 16
```

К сожалению, 16 строк оригинальной таблицы данных имели пропущенные значения. Заполним пропуски в данных на основе алгоритма бэггинга [8, с. 73]:

```
library(caret)
pPbI <- preProcess(algae[, 4:11], method='bagImpute')
algae[,4:11] <-  predict(pPbI, algae[,4:11])
```

Создадим копию таблицы с "русификацией" уровней факторов:

```
Водоросли <- algae
colnames(Водоросли)[1:3] <- c("сезон","река","течение")
levels(Водоросли[,1]) <- c"весна","лето",("осень","зима")
levels(Водоросли[,2]) <- c("большая", "средняя", "малая")
levels(Водоросли[,3]) <- c("быстрое","среднее","медленное")
```

* Комментарии составлены В.К.Шитиковым и Н.А.Цейтлиным

```
cbind(Водоросли[,1:3], signif(Водоросли[,4:12],3))
summary(Водоросли[1:3])
   сезон         река         течение
 осень:40   большая:45   быстрое  :84
 весна:53   средняя:84   медленное:33
 лето :45   малая  :71   среднее  :83
 зима :62
```

Сохраним таблицу для использования в дальнейшем:

```
save (Водоросли, file="algaeR.RData")
```

Читатель может не выполнять вышеперечисленные действия, а сразу скачать с нашего сайта RData-файл с уже готовой таблицей данных:

```
algaeR <-
"http://www.ievbras.ru/ecostat/Kiril/R/Erve/algaeR.RData"
load(file=url(algaeR))
```

11. Решение, отнести тот или иной фактор к фиксированным или случайным, принимается в зависимости от цели исследования, т. е. вопроса, на который полученные данные должны дать ответ. В рамках приведенного выше примера тип реки и скорость ее течения – фиксированные факторы, поскольку важно знать, как влияют на отклик конкретные его уровни (большая река, средняя или малая). Фактор сезон можно интерпретировать как случайный фактор и оценить долю дисперсии за счет различий во времени взятия проб. Однако при других обстоятельствах, когда в центре внимания анализа ставится сезонная динамика, он должен рассматриваться как фиксированный.

13. Проблемы планирования исследований многоплановы и неисчерпаемы. Теория составления рациональных планов была впервые обоснована Р.Фишером в книге «Планирование опытов» (1935), а его подходы, использующие комбинаторные принципы, используются и по сей день. Если воздействующие факторы являются управляемыми и независимыми (т.е. экспериментатор может устанавливать и постоянно поддерживать требуемые их значения в течение всего опыта и в любых сочетаниях), то появляется возможность построения матриц полного (или дробного) факторного эксперимента. Такие планы обладают важнейшими свойствами ортогональности и ротатабельности, что дает возможность оценить все коэффициенты регрессионной модели независимо друг от друга и "прозрачно" их интерпретировать. К сожалению, такие ситуации достаточно уникальны.

Реальные матрицы результатов пассивного эксперимента (в частности, полевых наблюдений) свойством ортогональности не обладают, и из-за этого часто возникает проблема мультиколлинеарности факторов. В этих условиях оценки регрессионных коэффициентов модели оказываются неустойчивыми и становится трудно анализировать вклад каждого предиктора в объясняемую величину. Например, можно столкнуться с парадоксальной ситуацией, когда, например, все коэффициенты множественной регрессионной модели статистически незначимы, тогда как сама модель оказывается значимой (т.е. проверяемая при помощи *F*-теста гипотеза о равенстве нулю *всех* коэффициентов отвергается).

Типичное явление большинства неуправляемых экспериментов – наличие в структуре полученных данных различных неоднородностей и "дрейфов" [9], искажающих оценки реального влияния факторов. Например, в сельскохозяйственных исследованиях источниками неоднородностей могут быть различия в плодородности участков земли, удобрениях, сортах семян, способах ирригации, а также текущие климатические флуктуации, антропогенные воздействия и многое другое. На

практике пытаются уменьшить влияние неконтролируемых возмущений и сделать их воздействие "случайным" путем рандомизации условий проведения эксперимента.

«*Псевдорепликация* (или использование мнимых повторностей) – есть некорректная проверка статистических гипотез при оценке эффекта влияния фактора по экспериментальным данным, когда группы с разными уровнями воздействий не могут быть признаны повторными, или эти повторности не являются статистически независимыми» – так определил С.Хёлберт (Hurlbert, 1984) еще одну из важнейших проблем планирования эксперимента. Подробное обсуждение статьи Хёлберта и других связанных с ней аспектов приводится в сборнике [10]. Для «демистификации псевдорепликации» предлагается использовать обобщенную линейную модель со смешанными эффектами (LMEM) [5, с. 131] как статистическое решение по анализу данных полевых исследований с псевдоповторностями. В этом случае можно корректно разделить между собой различные источники изменчивости зависимой переменной и получить правильные выводы из данных, собранных в исследованиях с ограничениями на рандомизацию.

Наконец, в середине прошлого века Дж. Бокс и Дж.Кифер заложили основы развития такого направления как оптимальное планирование эксперимента. Нахождение оптимальных планов в общем случае связано с построением нелинейных регрессионных моделей и решением систем уравнений больших размерностей [11]. Далеко не всегда решение таких задач удается представить в явном виде, однако выбор подходящей структуры множества планов позволяет существенно упростить выбор стратегии эксперимента.

14. Уточним, при каких ситуациях может быть сделан такой статистический вывод, как «поскольку найденное *p*-значение выше априори заданного критического порога α, то нет оснований отклонить H_0»:

- эффект влияния тестируемого фактора действительно слишком мал, чтобы его принимать во внимание;
- выборка недостаточно репрезентативна и поэтому, например, в формуле для статистики Стьюдента ошибка выборочного параметра $S_{\overline{X}}$ слишком велика;
- выбранный критерий *T* не соответствует предпосылкам его применения или имеет недостаточную мощность (1 - β).

Напомним, что тестовый критерий *T* – это некоторая выборочная статистика с известной функцией распределения, которая при определенных условиях может быть монотонно связана с генеральными параметрами случайных величин. Пусть $t_{эмп} = |t_{эмп}|$ – абсолютная величина критерия, рассчитанная по данным конкретных выборок. Тогда *p*-значение – это *условная вероятность* получить такое же значение $t_{эмп}$ (или большее) при справедливости нулевой гипотезы: $p = \Pr(T^* \geq t_{эмп} \mid H_0)$ [12].
Если H_0 верна, то *p*-значение имеет непрерывное равномерное распределение на отрезке [0, 1]. Если $t_{эмп}$ сильно отличается от нуля, то *p*-значение становится меньше критического порога α и нулевую гипотезу принято отклонять.

Однако, если мы не отвергли H_0 (т.е. не нашли различий), то нельзя говорить о том, что различий действительно нет. Например, это – повод провести дополнительное исследование, увеличив объем выборки *N*. *Внимание*: статистическую значимость всегда можно "купить" большим размером выборки!!! Приводится пример, как dr.Nostat изобрёл гипнотическое устройство для похудения: если его положить под подушку, оно за месяц понижает массу тела на 1 г. Доктор испытал устройство на выборке из $N = 6000$ пациентов и получил статистически значимый результат.

Если используются параметрические критерии, то необходимо также обязательно проверить, отвечают ли тестируемые выборки постулируемым предположениям. Например, при использовании критерия Стьюдента обе выборки должны иметь

нормальное распределение, а их дисперсии равными. Корректно проверить эти предпосылки удается далеко не всегда. Определенные ограничения мы имеем и при использовании непараметрических критериев.

Международный комитет редакторов биомедицинских журналов справедливо предостерегает [13]: «Не следует полагаться исключительно на проверку статистических гипотез, например, на использование *p*-значений, которые не содержат важную информацию о размере эффекта и точности оценки».

Отметим также, что статистика, считающаяся точной наукой, претендует и на лексическую точность. Вот некоторые правила (которых, к сожалению, не всегда придерживались М.Эрве и его переводчики), в основном вполне обоснованные, однако не лишенные некоторой казуистики:

- параметрические гипотезы формулируются относительно *генеральных* параметров распределения случайных величин, и следует писать не «групповые средние μ_A и μ_B», а «генеральные средние значения» или «математические ожидания»;
- статистический анализ оценивает вероятность, с которой *не отклоняют* нулевую гипотезу H_0, если она верна, так как меры надежности «*принять* гипотезу H_0» не существует.
- исследователь должен избегать категорических формулировок ("доказано" и т. п.), так как статистический анализ ничего не "доказывает", а лишь "не противоречит" принятым гипотезам;
- следует также категорически избегать слова "*достоверный*", поскольку по «самым *достоверным слухам*», полученным специалистами в области языкознания [14], оно означает «подлинный, не вызывающий сомнения, имеющий вероятность Pr = 1», что в статистическом анализе считается нонсенсом.

Наконец, распространено заблуждение, что «*p*-значение является вероятностью нулевой гипотезы». Это не может быть так, поскольку *p* сама является условной вероятностью $\Pr(t_{эмп}|H_0) \neq \Pr(H_0|t_{эмп})$. В статье С.Гудмана [15] перечислена полностью вся «грязная дюжина» заблуждений относительно интерпретации *p*-значений.

15. Результаты статистического теста зависят не только от природы изучаемого явления, результатов эксперимента или методики проверки гипотез, но и от субъективных факторов, связанных с выбором исследователем предпочтительной гипотезы и заданного им критического уровня значимости α. По обстоятельствам, описанным выше, на практике рассчитанное *p*-значение является точечной оценкой *уровня значимости*, который, в свою очередь, является случайной величиной с некоторыми моментами распределения (дисперсией, коэффициентами асимметрии и эксцесса) и иногда достаточно широким доверительным интервалом. Близость *p*-значения к 0.05 не является сильным свидетельством против нулевой гипотезы: диапазон от 0.1 до 0.01 обычно интерпретируют просто как "серую зону" неопределенности.

При разработке достаточно крупного научно-исследовательского проекта обычно имеется чёткое разделение ролей в теории принятия решений. Ответственную персону, фактически осуществляющую выбор наилучшего варианта действий с точки зрения финансирования проекта, называют *лицом, принимающим решения* (в 70-х годах его часто называли кратко **ЛПР**, а нынче – *эффективным менеджером*). Разумеется, в своей деятельности ЛПР не может быть совершенно беспристрастным, что может оказать влияние (часто даже никем не осознаваемое) на планирование исследований и выбор критических величин. В контексте проверки значимости выделяется два типа стратегий (http://statsoft.ru/home/textbook/modules/stpowan_rus.html): «поддерживаем отклонение H_0 (RS *reject-support*)» и «поддерживаем принятие H_0 (AS *accept-support*)».

Для RS-стратегии характерна вера исследователя в то, что эффект существует, и он ищет подтверждение своим предположениям с помощью теста, который бы значимо отверг нулевую гипотезу. Например, если производитель создал новый товар, то всегда предпочтёт альтернативную гипотезу H_1, утверждающую, что параметры нового товара лучше параметров прототипа. В этом исследовании назначаются умеренные величины α, ошибка I рода не считается проблемой, в то время как ошибка II рода является трагедией. Большой объем выборки работает на исследователя, и он старается максимизировать мощность критерия, не считаясь с тем, что иногда вполне тривиальные эффекты могут быть объявлены как "высоко значимые".

В AS исследовании нулевая гипотеза – это существующая теория, в которую исследователь верит и отрицает все возможные отклонения от нее. Например, если исследователь создал математическую модель устройства, и она должна быть адекватной описываемому объекту, то предпочтение отдается H_0 о равенстве модельных параметров, которые имитируют работу объекта, параметрам самого устройства. Для этого назначается экстремально низкие величины α и внимательно контролируется ошибка I рода. Большой объем выборки играет против исследователя, и он психологически готов согласиться с малой мощностью теста, не считаясь с тем, что реально существующие различия будут пропущены.

Критический уровень значимости α, вообще говоря, также является случайной величиной, поскольку формально определяется как обобщённая экспертная оценка, зависящая от многих случайных факторов. Разумеется, существует гнет существующих привычек назначать во всех случаях $\alpha = 0.05$. Но еще Н.А. Плохинский ввел нематематическое (экспертное) понятие «*уровень ответственности за выводы*» при отклонении гипотезы H_0: чем он выше, тем меньше α. Разработаны формулы и градации шкал, гибко оценивающих α в зависимости от экспертных уровня ответственности [16, 17].

Если принять p и α как случайные величины, то условие $p < \alpha$ также следует рассматривать как новую гипотезу, например, в виде $M\{p\} < M\{\alpha\}$ против другой гипотезы $M\{p\} > M\{\alpha\}$, где $M\{\}$ оператор математического ожидания. Если нам удастся проверить последнюю пару гипотез, то нам придется определять "новые" уровни значимости p_1 и α_1, а для их сравнения опять формулировать новую пару гипотез и т. д. Таким образом, процедура проверки гипотез может войти в "порочную спираль" [17]…

16. Обратимся к примеру множественных сравнений, в котором оцениваются параметры k популяций по результатам n независимых наблюдений в каждой группе. При сравнении генеральных средних μ_i и μ_j в группах i и j можно применить обычную t-статистику для проверки нулевой гипотезы H_{0ij}: $\mu_i = \mu_j$, или H_{0ij}: $(\mu_i - \mu_j) = 0$ против альтернативной гипотезы H_{1ij}: $\mu_i \neq \mu_j$. Разность $(\mu_i - \mu_j)$ находится в интервале

$$(\overline{x}_i - \overline{x}_j) \pm t_{\alpha/2,n-1}(s_i^2 + s_j^2)^{0.5} / n^{0.5},$$

где $\overline{x}$ и s – групповые оценки среднего и стандартного отклонения. Если этот доверительный интервал включает значение 0, то с вероятностью ошибки α генеральные средние μ_i и μ_j можно считать равными.

Однако если мы рассматриваем *групповую* (*family*) гипотезу равенства генеральных средних во всех группах объектов H_0**:** $\mu_1=\mu_2=\ldots=\mu_k$, против гипотезы H_1: «не все генеральные средние совпадают», то вероятность совершить ошибку хотя бы в одном сравнении существенно возрастает. Неравенство Бонферрони утверждает [6, 10], что при m независимых испытаний, каждое с доверительным уровнем $(1 - \alpha)$, верхний предел групповой вероятности ошибки α_E, задаваемый в эксперименте, равен

вероятности того, что не все эти утверждения истинны (т.е. ложно хотя бы одно из них): $\alpha_E \le 1 - \Pr(\text{все суждения истинны}) = 1 - \prod_{i=1}^{m}(1-\alpha_i) = 1 - (1-\alpha)^m \approx \sum_{i=1}^{m}\alpha_i$.

Следовательно, если удельная вероятность ошибки одного сравнения составляет α/m, где $m = k\cdot(k-1)/2$ – число всех возможных парных сравнений групповых средних, то общий уровень ошибки α_E, принятый в эксперименте, не превысит α. Другой вариант проверки групповой нулевой гипотезы о равенстве всех k средних заключается в расчете матрицы p-значений множественных парных сравнений, которая затем умножается на m. При этом H_0 отклоняется на уровне значимости α только в том случае, если все скорректированные значения $p\cdot m$ будут меньше α.

Оценка групповой вероятности ошибки с использованием поправки Бонферрони подвергается многочисленной критике – см презентацию с впечатляющим названием «Кошмар Бонферрони» [18]. Вот, например, аргументы Т.Пернежера (Perneger, 1998) из статьи «В чем вред поправки Бонферрони»:

- Интерпретация данных зависит от числа тестов. Это противно нашей интуиции. Данные не могут терять значимость от того, что их кто-то подтвердил еще раз!
- При большом количестве тестов гипотеза о том, что все наблюдаемые различия неслучайны, никому не нужна
- При коррекции Бонферрони вероятность упустить существенные различия столь велика, что лучше просто перечислить какие тесты дали значимые результаты и, главное, почему!
- Коррекция Бонферрони, в лучшем случае, не нужна и, в худшем случае, вредна для "здоровья" статистического анализа.

Во многих случаях архиконсервативная коррекция Бонферрони делает выявление значимых результатов попросту невозможным. Мощность единичного теста резко снижается и при 100 сравнениях вряд ли может превысить 0.12, т.е. ради того, чтобы гарантировать отсутствие хотя бы одного ложного результата, мы упускаем 88% открытий! [18]

Не столь консервативным является FDR-контроль (*false discovery rate*) или коррекция Бенджамини-Хохберга, которая оценивает среднюю долю фальшивых открытий от их общего числа. Например, пусть для 7 проведенных тестов [18] рассчитаны частные значения p_i, которые приведены ниже в отсортированном виде. Тогда в результате поправки Бонферрони статистически значимыми окажется только два теста, а после коррекции по FDR – шесть из семи.

Тест	p_i	Бонферрони	FDR
1	0,001	0.0071	$\alpha/7 = 0.0071$
2	0,0055	0.0071	$2\alpha/7 = 0.0143$
3	0,01	0.0071	$3\alpha/7 = 0.0214$
4	0,015	0.0071	$4\alpha/7 = 0.0286$
5	0,02	0.0071	$5\alpha/7 = 0.0357$
6	0,04	0.0071	$6\alpha/7 = 0.0429$
7	0,3	0.0071	$7\alpha/7 = 0.05$

Можно также принять во внимание, что функция `p.adjust()` выполняет коррекцию p-значений пятью различными способами:

```
P <- c(0.001, 0.0055, 0.01, 0.015, 0.02, 0.04, 0.3)
p.adjust.M <- p.adjust.methods
p.adj <- sapply(p.adjust.M,
                function(meth) p.adjust(as.vector(P), meth))
```

```
      holm hochberg hommel bonferroni    BH    BY   fdr  none
[1,] 0.007    0.007  0.007      0.007 0.007 0.018 0.007 0.001
[2,] 0.033    0.033  0.030      0.038 0.019 0.050 0.019 0.006
[3,] 0.050    0.050  0.040      0.070 0.023 0.061 0.023 0.010
[4,] 0.060    0.060  0.045      0.105 0.026 0.068 0.026 0.015
[5,] 0.060    0.060  0.060      0.140 0.028 0.073 0.028 0.020
[6,] 0.080    0.080  0.080      0.280 0.047 0.121 0.047 0.040
[7,] 0.300    0.300  0.300      1.000 0.300 0.778 0.300 0.300
```

где `holm` – коррекция методом Хольма, `hochberg` – Хохберга, `hommel` – Хоммеля, `bonferroni` – Бонферрони, `BH` и `fdr` – Бенджамини-Хохберга, `BY` – Бенджамини-Иекуетели, `none` – без коррекции. Описание большинства из этих методов дано в [6].

С приведенными рассуждениями можно соглашаться или нет, но "дефект логики" приведенных критических замечаний состоит в том, что исследователь всего-навсего каждый раз должен четко акцентировать, какую из гипотез он проверяет: групповую или локальную. Если задача состоит в том, чтобы оценить групповую вероятность ошибки, то для этого мы можем использовать любой метода контроля (даже `none`). Этот результат можно сравнить с выводами других общих тестов, полученными, например, на основе дисперсионного анализа статистической модели.

Однако предположим, что мы в ходе анализа не собираемся выполнять множественную проверку гипотез, а только собираемся оценить, отличаются ли групповые средние для конкретной пары выборок *i* и *j*. Должны ли мы тогда вносить в обычную методику расчета *t*-статистики для двух совокупностей поправки Бонферрони или FDR? Нет, а зачем? Ведь групповой вероятностью ошибки мы больше не интересуемся!!! Ничто нам не мешает анализировать и публиковать результаты сравнений локальных пар групп наблюдений для этой и всех остальных возможных сочетаний *i* и *j* по обычным формулам для двух совокупностей. Нужно только объявить, что мы *не ставим своей задачей* оценивать групповую вероятность.

Например, оценим значимость влияния сезонного периода на содержание растворенного кислорода в воде по результатам взятия проб в таблице `Водоросли` – см. комментарий к п. **6**. По результатам дисперсионного анализа линейной модели (см. комментарии к п. **42**) фактор сезонности статистически значим:

```
library(car)
Anova(lm(mnO2~сезон,data=Водоросли))
Anova Table (Type II tests)
Response: mnO2
          Sum Sq  Df F value    Pr(>F)
сезон     155.79   3   10.45 2.078e-06 ***
Residuals 974.03 196
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Выполним теперь множественные сравнения всех сезонных периодов с использованием теста Стьюдента с пермутацией:

```
library(RVAideMemoire)
Pt <- pairwise.perm.t.test(Водоросли$mnO2,Водоросли$сезон,
                                p.method="none")
Pairwise comparisons using permutation t tests 999 permutations
      осень весна лето
весна 0.002 -     -
лето  0.004 0.002 -
зима  0.004 0.116 0.224
```

Оба теста не противоречат друг другу, и большинство гидрологов сделает вывод, что содержание кислорода в водотоках меняется в течение вегетационного периода. Однако отсутствие статистически значимых изменений зимой по сравнению с весной и летом не позволяет формально отклонить групповую нулевую гипотезу (даже без каких-либо коррекций *p*-значений), но здесь это практического смысла не имеет.

17. В определении мощности следует говорить об альтернативной гипотезе H_1, ибо именно по ней и определяется вероятность β: «Значение величины 1-β численно равно вероятности правильно принять гипотезу H_1, если она фактически верна». Это определение справедливо для простой гипотезы H_1. Поскольку обычно гипотеза H_1 – сложная, то принято говорить о функции мощности [16, 17].

18. С помощью статистического *анализа мощности* (Statistical Power Analysis) выполняется исследование взаимосвязи между четырьмя компонентами:

(а) стандартизованной величиной ожидаемого эффекта δ – например, для теста Стьюдента используется нормированная разность средних $\delta = (\mu_i - \mu_i)/\sigma$;

(б) числом выполняемых измерений *n*;

(в) задаваемой величиной уровня значимости α;

(г) необходимым соотношением между рисками ошибки I (α) и II (β) рода.

Если проверка H_0 и оценка ошибки I рода выполняется с использованием выборок наблюдений и зависит от характера распределения эмпирических данных, то анализ мощности является формализованной процедурой, в значительной мере абстрагированной от самого эксперимента. Это лишь оценка потенциальной чувствительности выбранного вами метода, включая все его предпосылки и условия реализации, на что необходимо обязательно обращать внимание при интерпретации результатов.

В частности, для оценки мощности теста для многих экспериментальных исследований может быть использована формула (http://egap.org/methods-guides/10-things-you-need-know-about-statistical-power), где используется предположение о нормальном распределении опытной и контрольной групп:

$$1\text{-}\beta = \Phi\left(\frac{|\mu_1 - \mu_2|\sqrt{N}}{2\sigma} - \Phi^{-1}(1-\alpha/2)\right),$$

где Φ – функция нормального распределения, а Φ^{-1} – ее обратная функция, *N* – общее число наблюдений ($n = N/2$).

Пусть, например, в контрольной группе из 20 спортсменов была получена средняя частота пульса 68 ударов в мин., а в группе после проведения соревнования – 70 уд/мин. Примем также, что усредненное для разных групп стандартное отклонение $s = 3$ уд/мин. Оценим по представленной формуле мощность теста при данных условиях и критической значимости $\alpha = 0.05$.

```
#  Функция оценки мощности выявления эффекта
#  при сравнении двух групповых средних
power_calculator <- function(mu_t, mu_c, sigma,
                             N, alpha=0.05){
  lowertail <- (abs(mu_t - mu_c)*sqrt(N))/(2*sigma)
  uppertail <- -1*lowertail
  beta <- pnorm(lowertail- qnorm(1-alpha/2),
     lower.tail=TRUE) + 1- pnorm(uppertail- qnorm(1-alpha/2),
      lower.tail=FALSE)
  return(beta)
  }
```

```
power_calculator(70, 68, 3, 40)
[1] 0.5589396
```

Аналогичные расчеты мы можем выполнить с помощью функции `pwr.t.test` пакета `pwr` (см. [3, 6]):

```
library(pwr)
pwr.t.test(power = NULL, n=20,
       d=(70-68)/3, sig.level=.05)  , type="two.sample",
       alternative="two.sided")
Two-sample t test power calculation
              n = 20
              d = 0.6666667
      sig.level = 0.05
          power = 0.5377868
```

Некоторые различия в оценках мощности можно отнести, например, за счет того, что во втором случае аппроксимация проводилась распределением Стьюдента.

Обычно при проведении анализа можно варьировать всеми переменными δ, n, α и β в разных их сочетаниях, но таким образом, чтобы значения трех любых параметров были бы зафиксированы для оценки неизвестного четвертого. В путеводителе обсуждается оценка числа измерений n, которая обычно проводится перед началом эксперимента.

Не меньший интерес представляет также оценка размера эффекта после проведения анализа и тестирования гипотезы H_0. Дж. Коен (Cohen, 1988) предложил при сравнении средних значений использовать в качестве оценки стандартизованного эффекта величину $d = (\overline{X}_1 - \overline{X}_2)/SD$, где SD – объединенная оценка стандартного отклонения. Интерпретация статистики d идентична t-статистики, но, в отличие от t, она не зависит от размера выборки. Из этих соображений размер эффекта от 0.2 до 0.5 считается малым, а от 0.5 до 0.8 – большим.

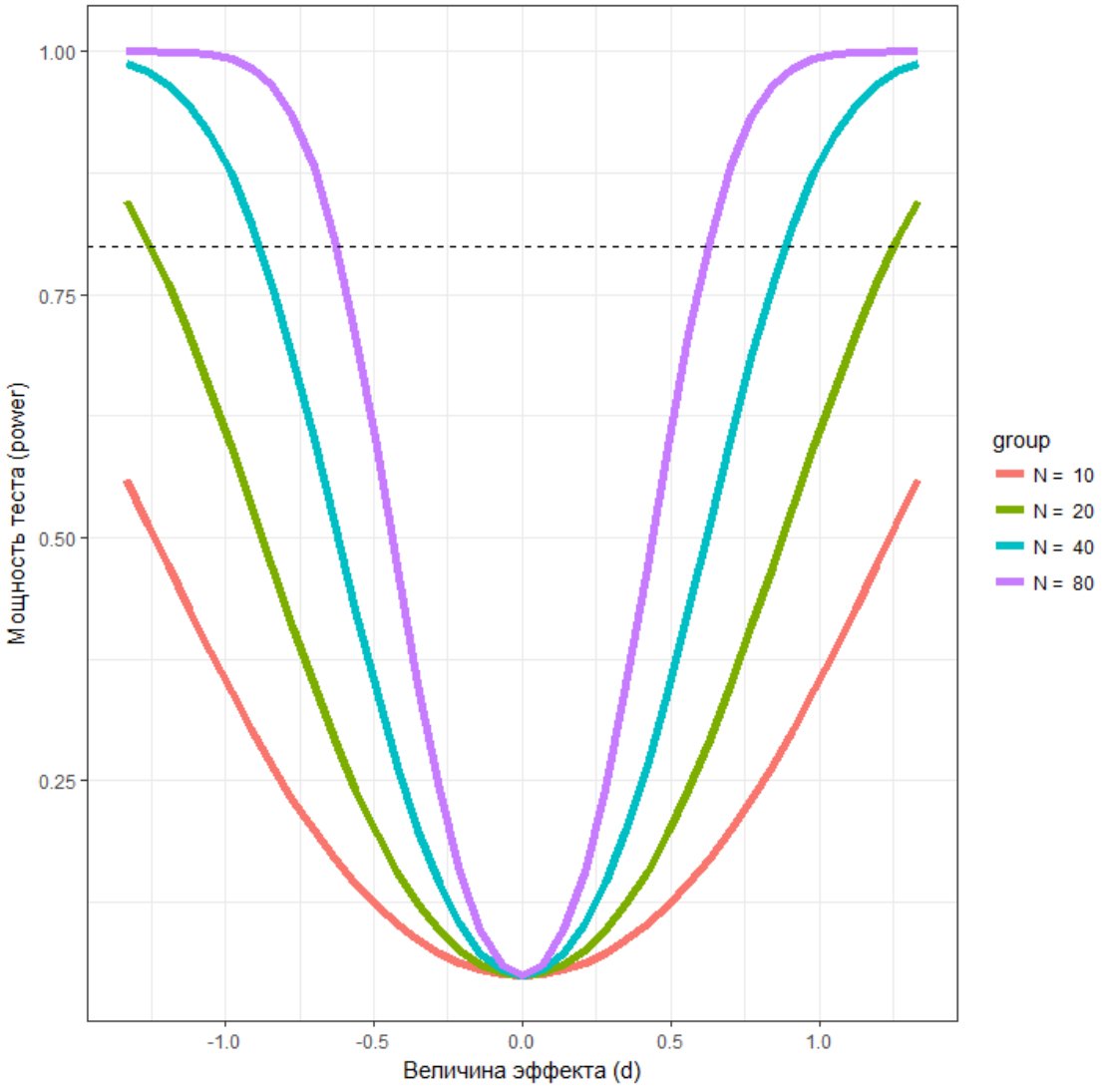
Анализ мощности с оценкой величины эффекта δ позволяет уточнить сделанный статистический вывод https://docplayer.ru/52122188-Moshchnost-statisticheskogo-testa-razmer-effekta-transformaciya-dannyh.html :

1. Если H_0 не отклонена, то
   - при большом δ нельзя говорить, что различий действительно нет: это повод провести дополнительное исследование, увеличив n;
   - при малом δ различий, скорее всего, действительно нет.
2. Если H_0 отклонена, то
   - при большом δ, скорее всего, различий действительно есть;
   - при малом δ надо хорошо подумать, имеют ли найденные различия биологический смысл или тут, скорее всего, велика ошибка измерения, а статистическая значимость эффекта "куплена" большим числом наблюдений.

С помощью приведенной функции `power_calculator()` смоделируем значения мощности `power` t-критерия, которые мы можем достичь при различных объемах выборок, чтобы выявить эффект заданного размера при уровне значимости $\alpha = 0.05$. Диаграмму построим для четырех фиксированных значений $N = 10, 20, 40$ и 80.

```
N <- c(10,20,40,80)
d.max = 4
sigma = 3
# Формирование таблицы значений мощности для разных d и N
ptab <- cbind(NULL, NULL)
```

```
for (i in 1:length(N)) {
   for (mu_c in seq(70,70+d.max, length.out = 20)){
     ptab <- rbind(ptab,c(
         group = N[i], d=(mu_c-70)/sigma,
         power = power_calculator(70, mu_c, sigma, N[i]))) }
}
# Дублирование таблицы для отрицательной ветви графика
ptab.neg <- ptab
ptab.neg[,2] <- 0 - ptab.neg[,2]
ptab <- rbind(ptab,ptab.neg)
ptab <- as.data.frame(ptab)
ptab$group <- as.factor(paste("N = ",ptab$group))
# Формирование графика кривых мощности
library(ggplot2)
ggplot(ptab, aes(x = d, y = power, color = group)) +
    xlab("Величина эффекта (d)") +
    ylab("Мощность теста (power)") +
    geom_line(size=2) +
    geom_hline(yintercept = 0.80, linetype = 2) +
    theme_bw()
```



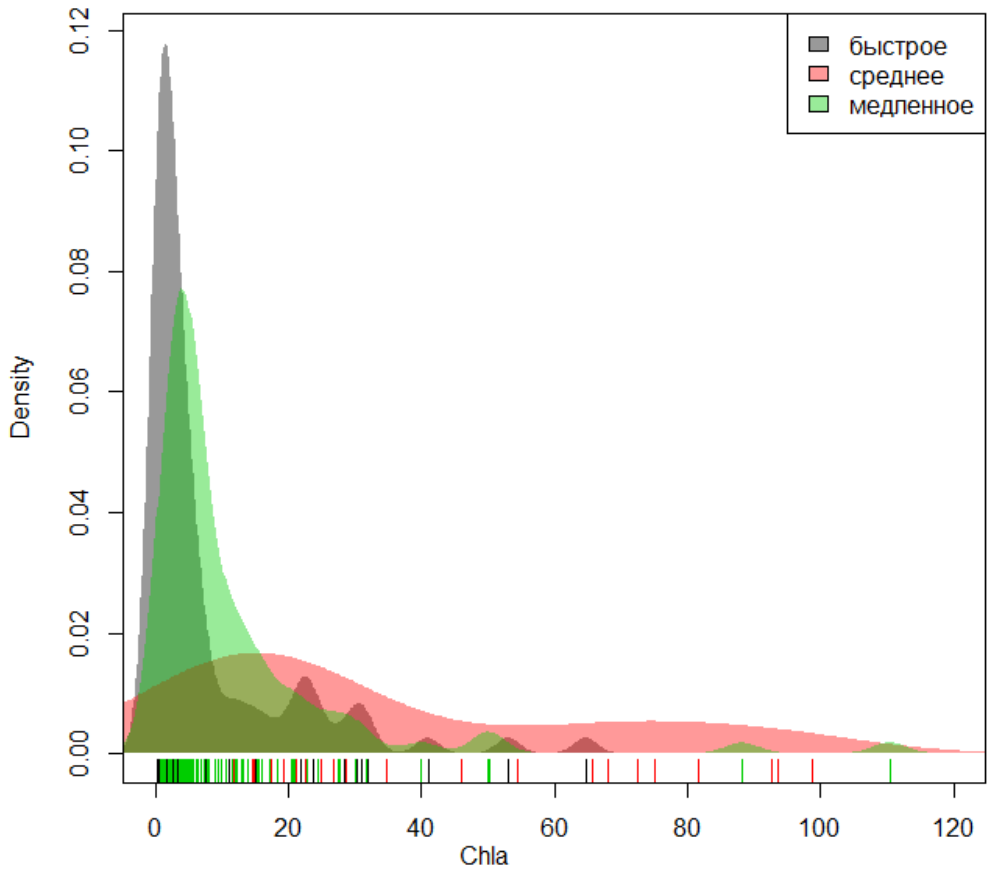
Очевидно, что малый эффект (δ = 0.2) не обнаруживается с приемлемой вероятностью даже при достаточно больших объемах выборок, а идентификация эффекта среднего размера (δ = 0.4 ÷ 0.6) требует более 80 измерений пульса.

28. Отметим, что все распределения **26-28**, используемые для проверки гипотез, тесно связаны между собой и с "нормированным нормальным" (или гауссовым) распределением z ~ N(0, 1):

- "хи-квадрат" Пирсона с f степенями свободы, $\chi^2_f = \sum_{i=1}^{f} z_i^2$;
- распределение Стьюдента с f степенями свободы, $t_f = z/(\chi^2_f/f)^{0,5}$;
- распределение Фишера-Снедекора с f_1 и f_2 степенями свободы $\upsilon^2_{f_1,f_2} = (\chi^2_{f_1}/f_1)/(\chi^2_{f_2}/f_2)$.

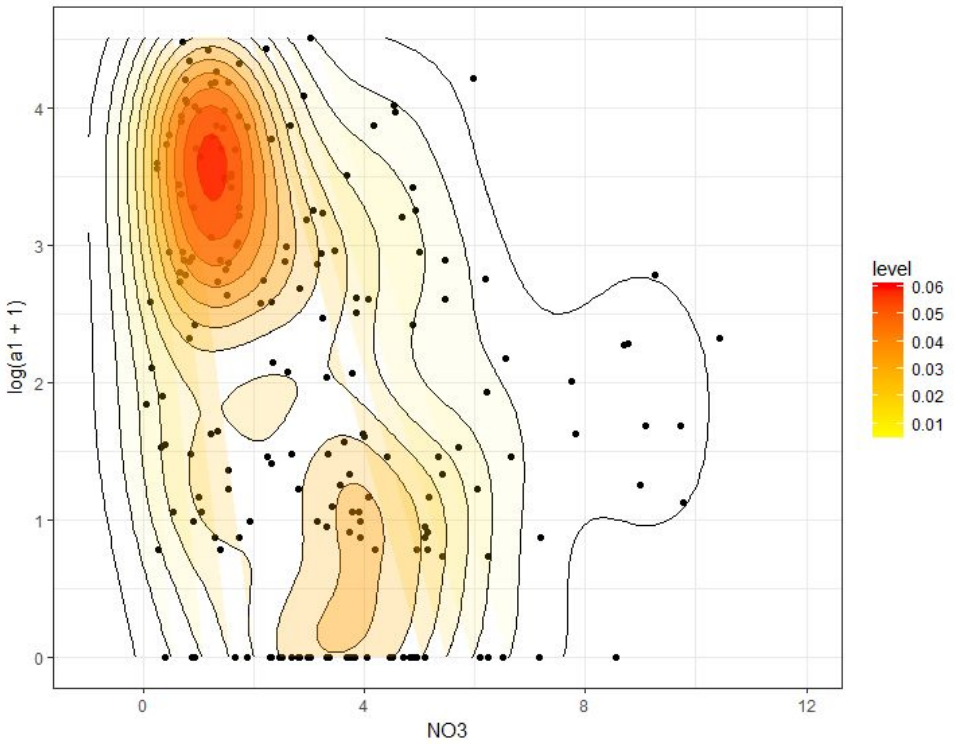
31. Построим график зависимости ядерной плотности распределения хлорофилла A от скорости течения (см. комментарий к п. **6**):

```
library(RVAideMemoire)
byf.hist(Chla ~ течение, data=Водоросли)
```



Отметим, что для визуализации данных превосходными возможностями обладает один из, несомненно, лучших графических пакетов для R – `ggplot2` [19-20], Это – прекрасная альтернатива отдельным функциям разных пакетов, которая на единой методической основе позволяет строить как всевозможные тривиальные диаграммы, так и гораздо более сложные графики, включающие, например, двумерные диаграммы распределения плотности. Построим такой график совместного распределения концентрации нитратов и обилия водорослей вида `a1` (в логарифмах):

```
library(ggplot2)
ggplot(Водоросли, aes(x = NO3, y = log(a1+1))) +
   geom_point() +  geom_density2d(colour="black") +
   stat_density2d(aes(alpha=..level.., fill=..level..),
       size=2, geom="polygon") + xlim(-1, 12) + theme_bw() +
   scale_fill_gradient(low = "yellow", high = "red") +
   scale_alpha(range = c(0.00, 0.5), guide = FALSE)
```



31. Построим график типа "фасоль" для сезонной изменчивости содержания кислорода в воде, позволяющий объяснить матрицу множественных сравнений – п. **16**:

```
library("beanplot")
beanplot (mnO2 ~ сезон, data=Водоросли)
```

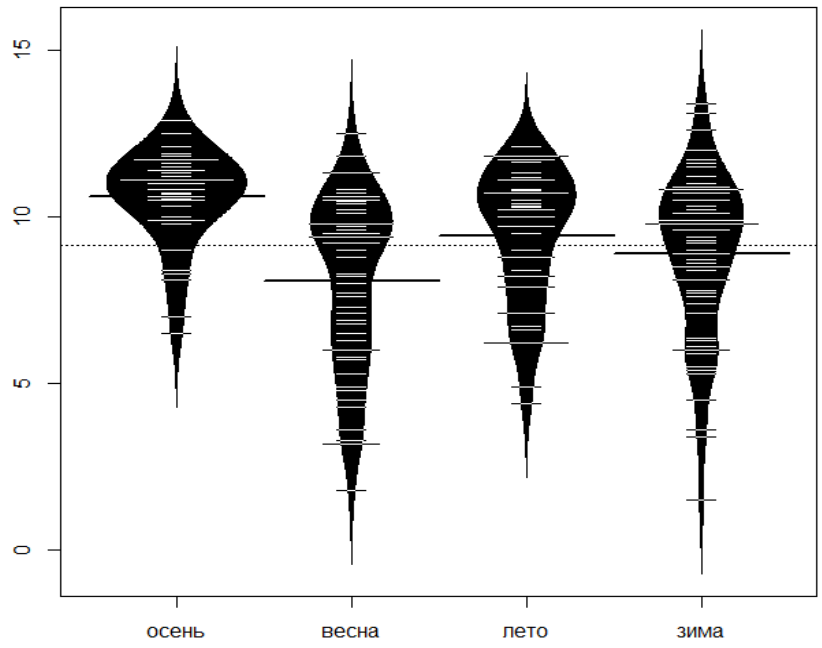


График типа "фасоль" можно сравнить с рис. 3.4 на стр. 74 [8], где представлена так называемая "*скрипичная диаграмма*" (violin plot), которая построена с использованием пакета `ggplot2` и также объединяет в себе идеи диаграмм размахов и кривых распределения вероятности.

31. Пакет `rcompanion`, разработанный С.Мангифико (Mangiafico) и детально представленный в электронной книге на http://rcompanion.org/, является прекрасным дополнением `RVAideMemoire`. В частности, он предоставляет расширенные возможности получить выборочные оценки доверительных интервалов различных параметров как с использованием традиционных (параметрических) методов, так и бутстрепа.

Например, для среднего арифметического содержания ионов аммония `NH4` в реках разного типа:

- традиционным методом – `traditional = TRUE`

```
library("rcompanion")
groupwiseMean(NH4~река, data=Водоросли,conf = 0.95,digits = 3)
     река  n Mean Conf.level Trad.lower Trad.upper
1 большая 45  215       0.95      111.0        320
2 средняя 84  671       0.95       80.6       1260
3   малая 71  475       0.95      148.0        802
```

- с использованием бутстрепа – `boot = TRUE` – и процедуры BCa (*bias correction and acceleration*) [5, с. 25] – `bca = TRUE`

```
groupwiseMean(NH4~река, data=Водоросли, R = 10000,
              boot = TRUE, traditional = FALSE, bca = TRUE)
     река  n Mean Boot.mean Conf.level Bca.lower Bca.upper
1 большая 45  215       216       0.95       138       354
2 средняя 84  671       674       0.95       328      2040
3   малая 71  475       475       0.95       251       972
```

- с учетом второго фактора – скорости течения (для малых рек с низким течением нашлось только одно измерение и доверительные интервалы найдены не были)

```
groupwiseMean(NH4 ~ река + течение, data=Водоросли)
     река течение  n   Mean Conf.level Trad.lower Trad.upper
1 большая высокое  7   74.0       0.95     -0.241       148.0
2 большая  низкое 17  316.0       0.95    100.000       532.0
3 большая среднее 21  181.0       0.95     33.600       328.0
4 средняя высокое 34  875.0       0.95   -556.000      2310.0
5 средняя  низкое 15  384.0       0.95    102.000       666.0
6 средняя среднее 35  595.0       0.95    170.000      1020.0
7   малая высокое 43   54.8       0.95     31.500        78.1
8   малая  низкое  1   10.0       0.95         NA          NA
9   малая среднее 27 1160.0       0.95    336.000      1980.0
```

Для доверительных интервалов параметра центра распределения ионов аммония с использованием М-оценок Хубера (модификация метода наибольшего правдоподобия):

```
groupwiseHuber(NH4 ~ река ,data=Водоросли)
     река  n   M.Huber  lower.ci  upper.ci
1 большая 45 100.70439  76.00367 125.40510
2 средняя 84 175.14644 145.32836 204.96452
3   малая 71  63.16985  47.16628  79.17342
```

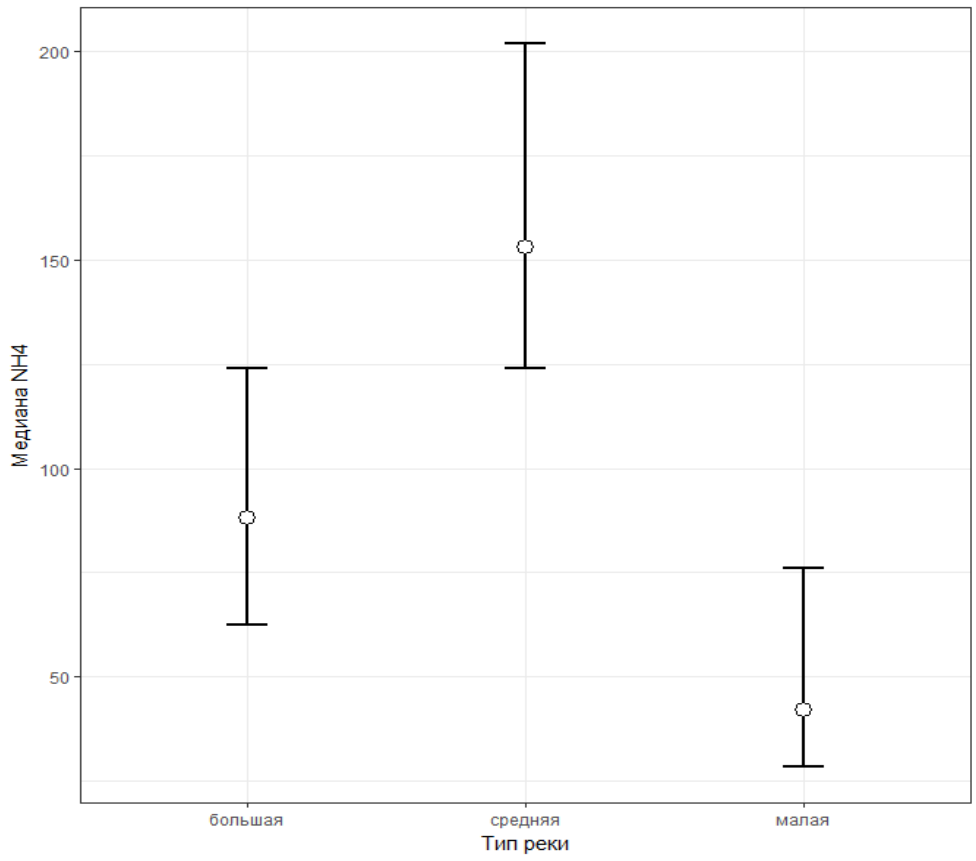
Для медианы (на основе бутстрепа и метода процентилей):

```
Ms <- groupwiseMedian(NH4 ~ река, data=Водоросли,
      R = 5000,percentile = TRUE,bca = FALSE)
     река  n Median Conf.level Percentile.lower Percentile.upper
1 большая 45   88.2       0.95            60.1             124
2 средняя 84  153.0       0.95           124.0             201
3   малая 71   42.0       0.95            28.3              76
```

Представленные результаты показывают сильные различия выборочных параметров положения (обоих средних и медианы), а также доверительных интервалов, рассчитанных параметрическим методом и бутстрепом. Например, верхняя граница доверительного интервала для средних рек оценивается от 201 (медиана) до 2040 (бутстреп). Такое явление характерно для выборок с асимметричным распределением и/или при наличии выбросов.

Различия между группами проиллюстрируем графиком оценок доверительных интервалов медианы:

```
library(ggplot2)
ggplot() + labs(x='Тип реки', y='Медиана NH4') + theme_bw() +
   geom_errorbar(data=Ms, mapping=aes(x=река,
      ymin= Percentile.lower, ymax=Percentile.upper),
      width=0.15, size=1) +
   geom_point(data=Ms, mapping=aes(x=река, y=Median),
       size=4, shape=21, fill="white")
```



37. Анализ выбросов обычно проводится с использованием критерия Граббса или правила Тьюки. Оба метода предполагают нормальное распределение данных.

Функция, реализующая тест Граббса, проверяет нулевую гипотезу, что самое максимальное выборочное значение не является выбросом. Если нулевая гипотеза отклоняется, можно исключить выявленный выброс и продолжить проверку. Эту процедуру можно реализовать функцией (https://stackoverflow.com/questions/22837099/how-to-repeat-the-grubbs-test-and-flag-the-outliers) :

```
grubbs.P <- function(x) {
  outliers <- NULL
  parray <- NULL
  test <- x
  grubbs.result <- grubbs.test(test)
  pv <- grubbs.result$p.value
  while(pv < 0.05) {
    outliers <- c(outliers,
   as.numeric(strsplit(grubbs.result$alternative," ")[[1]][3]))
    parray <- c(parray, pv)
    test <- x[!x %in% outliers]
    grubbs.result <- grubbs.test(test)
    pv <- grubbs.result$p.value   }
   return(data.frame(X=outliers, P = parray))
}
library(outliers)
grubbs.P(Водоросли$NH4)
           X            P
1  24064.000 0.000000e+00
2   8777.600 0.000000e+00
3   6400.000 6.594725e-14
4   5738.330 0.000000e+00
5   4073.330 1.382436e-10
6   3515.000 5.921530e-10
…
```

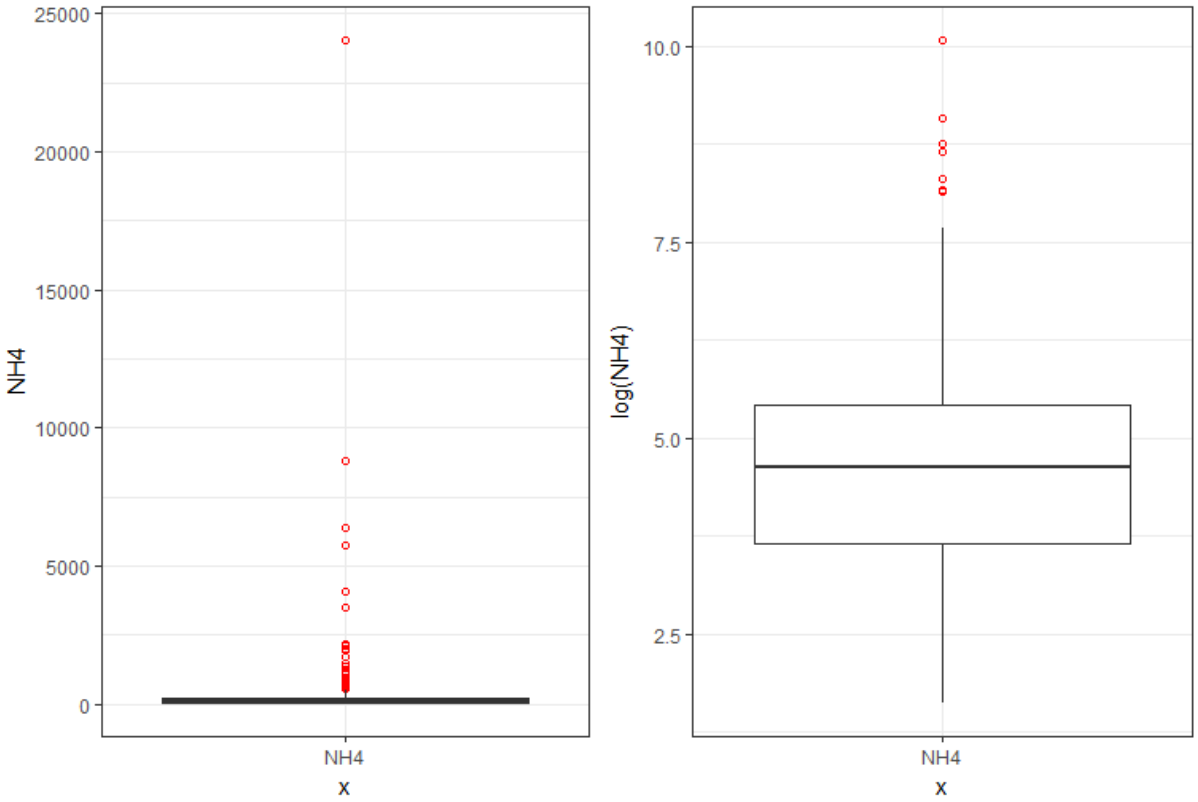
Было идентифицировано в качестве выбросов 30 значений содержания ионов аммония, что довольно много. Можно, конечно, объявить наш тест "групповой проверкой гипотез" и применить поправку Бонферрони α/30 = 0,0017, но и тогда число выбросов сократится только до 21. Прологарифмируем выборочные значения:

```
grubbs.P (round(log(Водоросли$NH4),4))
        X          P
1 10.0885 0.03012944
```

После преобразования выборочных данных выбросом считается только одно значение.

Построим диаграмму «ящик с усами» до и после логарифмирования выборки.

```
library(ggplot2)
ggplot(data=Водоросли, mapping=aes(x="NH4", y=log(NH4))) +
geom_boxplot(outlier.color="red", outlier.shape=1) + theme_bw()
```



Если использовать правило Тьюки, то выбросами считаются выборочные значения, находящие за пределами "усов" построенного графика `boxplot`. Напомним, что наиболее популярная длина "уса" – полтора интерквартильного размаха IQR, что приводит к излишне жестким условиям отсева выбросов (см. диаграмму справа):

```
x <- log(Водоросли$NH4)
qnt <- quantile(x, probs=c(.25, .75), na.rm = T)
H <- 1.5 * IQR(x, na.rm = T)
rev(sort(x[x > (qnt[2] + H)]))
[1] 10.0885  9.0800  8.7641  8.6549  8.3122  8.1648  8.1509
```

Если использовать более компромиссное значение параметра `range = 2.5`, то то мы получим тот же результат, что и для теста Граббса:

```
boxplot(x, range=2.5)$out
[1] 10.0885
```

После индентификации выбросов в одномерном массиве данных возникает вопрос, как заблокировать их негативное влияние. Возможно множество решений, из которых чаще всего используются следующие три:

- *бескомпромиссное*, связанное с удалением выбросов из выборки, которое считается наименее целесообразным (особенно, если выборка мала, а в статусе выбросов имеются сомнения);
- *виндзоризация* или исправление выбросов путем их замены на некоторые экстремальные, но допустимые значения, такие как 97.5%-й квантиль, утроенный интерквартильный размах, утроенное стандартное отклонение или иное (см. также [16]);
- удаление выбросов и последующее *заполнение пропущенных значений* с использованием тех или иных "интеллектуальных" алгоритмов (см. [8, с. 73]).

Исключение выброса может быть связано с потерей информации о виде распределения случайной величины. Поэтому виндзоризация и использование бэггинг-моделей для "исправления" данных считаются наиболее рациональными приёмами.

В нашем примере мы планируем построить модель зависимости биомассы водорослей от восьми гидрохимических показателей и трех факторов. В этих условиях выглядит не вполне корректным объявлять всё наблюдение выбросом, основываясь только на одном показателе. Тем более, что методы ранжирования многомерных выбросов представлены Л.Тарго как в его монографии [21], так и в пакете `DMwR`.

Функция `lofactor()` находит для каждого наблюдения оценку того, насколько оно является выбросом (*outlyingness*). Реализованный в ней алгоритм LOF (*local outlier factor* – Breunig et al., 2000) основан на понятии многомерной плотности наблюдений и рассчитывает степень изоляции объекта по отношению к своему кластеру, составленному из *k* ближайших соседей. Значения LOF обычно варьируют около 1 и объекты с коэффициентами, превышающими 1.5 ÷ 2, можно считать многомерными выбросами.

Выше показано, что удачная трансформация позволяет привести разброс данных в рамки некоторых статистических предположений, обуславливающих применение большинства методов анализа. Поэтому прологарифмируем 6 из 8 гидрохимических показателей (за исключением pH и содержания O_2). Метод LOF требует стандартизации данных и задания числа ближайших соседей *k*:

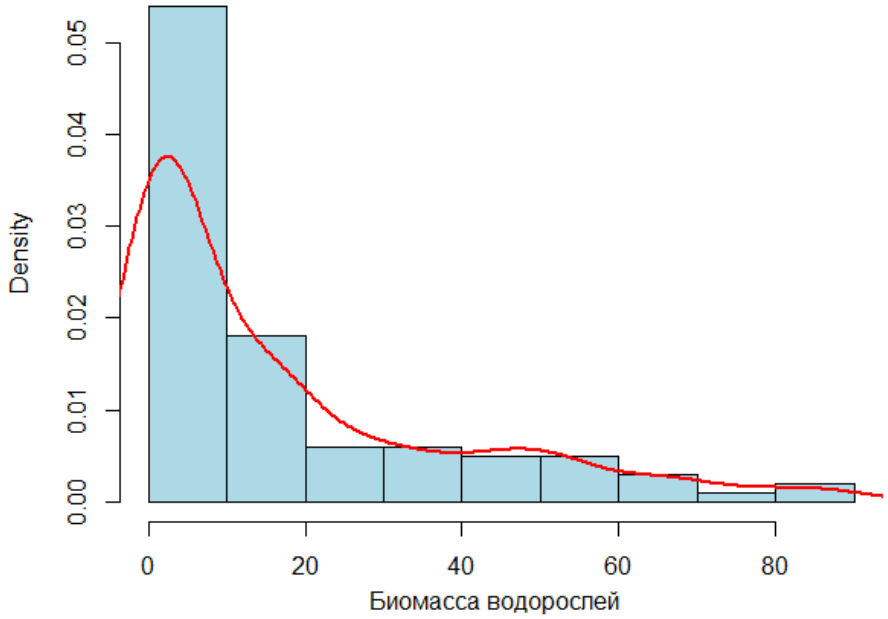
```
ГХП_мод <- cbind(Водоросли[,4:5],log(Водоросли[,6:11]))
ГХП_стан <- scale(ГХП_мод, center = TRUE, scale = TRUE)
library(DMwR)
W.lof <- lofactor(ГХП_стан, k = 5)
range(W.lof)
[1] 0.9318469 2.7336440
print(which(W.lof > 2))
[1]  35 116 153
print(Водоросли[which(W.lof > 2),4:11])
    mxPH mnO2     Cl    NO3   NH4   oPO4     PO4      Chla
35  8.27  7.8 29.200  0.050  6400  7.400 23.000  0.90000
116 9.70 10.8  0.222  0.406    10 22.444 10.111 64.75598
153 7.30 11.8 44.205 45.650 24064 44.000 34.000 53.10000
```

Для трех "подозрительных" наблюдений экстремальные значения показателей, обуславливающие, вероятно, величину LOF, выделены жирным шрифтом.

39-42. Если число наблюдаемых в эксперименте показателей достаточно велико, то составлению формулы модели обычно предшествует кропотливая работа по предварительной оценке их корреляционной связи, а также характера распределения как отклика, так и отдельных предикторов (чтобы, например, принять решение об их предварительном преобразовании). Строгих правил для выбора структуры функциональной части регрессии чаще всего нет. Обычно это – основанные на опыте предположения специалистов-экспертов о том, что факторы X_i по своей природе могут оказать существенное влияния на отклик Y. Как правило, на структуру модели накладываются прагматические ограничения: она должна быть по возможности лаконичной и линейной по параметрам. Несомненную помощь при составлении формулы модели может оказать процедура селекции комплекса независимых переменных, оптимального относительно информационных критериев (п. **44**).

Рассмотрим поэтапно возможные действия при построении модели линейной регрессии зависимости биомассы водорослей `a1` от наблюдаемых номинальных и количественных показателей качества воды в реке. Вначале отметим асимметричный характер распределения значений отклика:

```
hist(Водоросли[,12],  breaks = 12, freq = FALSE, col =
    "lightblue", xlab= "Биомасса водорослей", main="")
lines(density(Водоросли[,12]), col = "red", lwd = 2)
```



Простые нормализующие преобразования, такие как логарифмирование или двойное извлечение квадратного корня также не приводят в нашем случае к формированию симметричного распределения.

Далее выполним анализ статистической значимости влияния каждого из трех фиксированных факторов, чтобы определить, какие из них имеет смысл включить в общую модель (п. **42**). Осуществим для этого построение линейной модели трехфакторного дисперсионного анализа и ее тестирование по типу I, II и III:

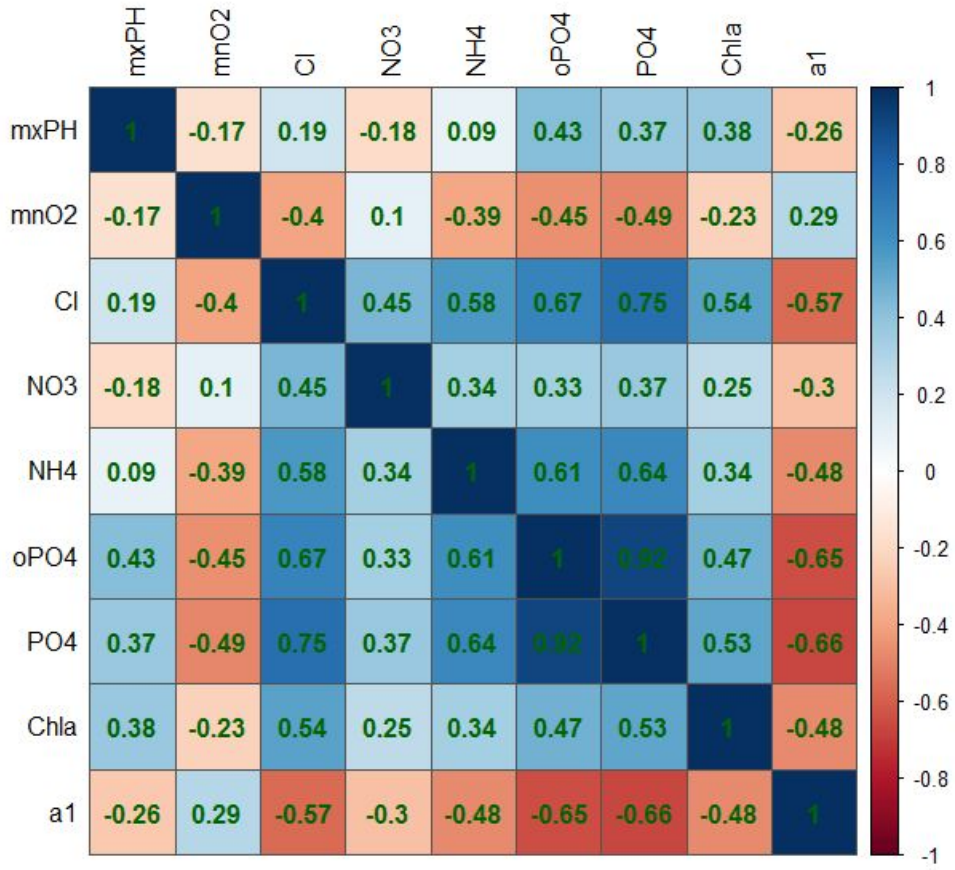
```
lm2f.a1 <- lm(a1 ~ река*течение + сезон,data=Водоросли)
anova(lm2f.a1)
Analysis of Variance Table
              Df Sum Sq Mean Sq F value    Pr(>F)
река           2  11501  5750.6 14.7054 1.165e-06 ***
течение        2   4000  1999.9  5.1141  0.006875 **
сезон          3    120    40.1  0.1025  0.958524
река:течение   4   1556   389.0  0.9948  0.411632
Residuals    188  73518   391.1

library(car)
Anova(lm2f.a1)
Anova Table (Type II tests)
              Sum Sq  Df F value    Pr(>F)
река            7246   2  9.2649 0.0001454 ***
течение         4028   2  5.1506 0.0066415 **
сезон            167   3  0.1422 0.9345748
река:течение    1556   4  0.9948 0.4116323
Residuals      73518 188
```

```
Anova(lm2f, type ="III")
Anova Table (Type III tests)
              Sum Sq  Df F value    Pr(>F)
(Intercept)     4008   1 10.2486 0.0016062 **
река            6251   2  7.9920 0.0004664 ***
течение         1595   2  2.0394 0.1329792
сезон            167   3  0.1422 0.9345748
река:течение    1556   4  0.9948 0.4116323
Residuals      73518 188
```

По результатам теста типа III единственно значимым является главный фактор `река`, который мы будем учитывать в дальнейшем при построении моделей, тогда как остальные факторы, включая их взаимодействия, признаются незначимыми. Здесь важно и биологическое соображение: малые реки обычно имеют более быстрое течение, т. е. размер реки тесно связан со скоростью водотока и нет необходимости включать в модель два тесно связанных фактора.

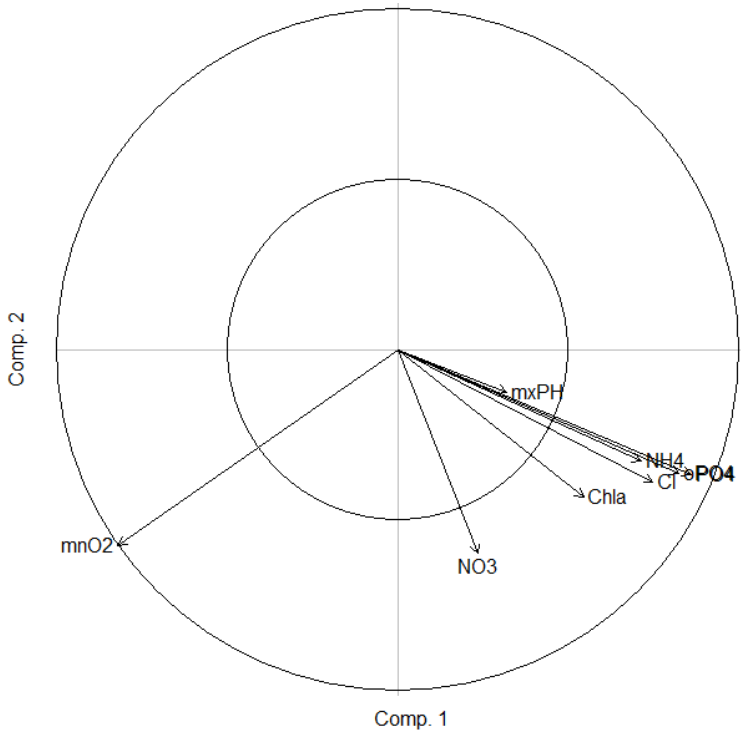
Важные предварительные выводы можно сделать, сформировав корреляционную матрицу количественных независимых переменных. Чтобы учесть нелинейный характер зависимостей и уменьшить влияние выбросов, прологарифмируем 6 из 8 гидрохимических показателей. Отобразить результаты можно в виде раскрашенной корреляционной матрицы:

```
ГХП_кор <- cbind(Водоросли[,4:5],log(Водоросли[,6:11]))
P_кор <- cbind(ГХП_кор,a1 = Водоросли[,12])
Mcor <-cor(P_кор)
library(corrplot)
corrplot(Mcor, method="color", addCoef.col=" darkgreen",
         addgrid.col = "gray33", tl.col = "black")
```



Другой вариант – диаграмма в виде корреляционного круга (см. п. **89**).

```
library(vegan)
PCA <- rda(ГХП_кор)
library(RVAideMemoire)
MVA.plot(PCA,"corr")
```



Очевидно, что хотя между предикторами существуют корреляционные связи достаточно умеренной силы, оси не менее 5 показателей из 8 имеют в многомерном пространстве примерно идентичную направленность.

Выполним отбор подмножества наиболее информативных предикторов модели с использованием функции `dredge()` пакета `MuMIn` (см. подробное описание пакета на stok1946.blogspot.com/2018/08/blog-post_19.html). Вначале построим полную модель на базе 8 гидрохимических показателей, 6 из которых прологарифмируем.

```
library(MuMIn)
options(na.action = "na.fail")
P_мод <- cbind(Водоросли[,c(12,4:5)], log(Водоросли[,6:11]))
lm.full <- lm(a1~., data=P_мод)
```

Выполним полный перебор всех 256 возможных частных регрессионных моделей:

```
ms1 <- dredge(lm.full)
subset(ms1,delta < 1.4)
```

```
Global model call: lm(formula = a1 ~ ., data = P_мод)
Model selection table
    (Intrc)   Chla     Cl     NH4   oPO4    PO4 df   logLik   AICc delta weight
196   59.54 -2.386 -2.250         -4.055 -3.915  6 -830.758 1674.0  0.00  0.223
194   59.05 -2.747                -3.828 -5.498  5 -831.963 1674.2  0.28  0.194
68    55.07 -2.589 -3.202         -6.654         5 -832.018 1674.3  0.39  0.183
210   60.94 -2.743       -1.2010 -3.622 -4.804  6 -831.159 1674.8  0.80  0.150
84    57.24 -2.620 -2.749 -1.1320 -6.156         6 -831.321 1675.1  1.13  0.127
212   60.96 -2.429 -1.955 -0.9392 -3.863 -3.580  7 -830.281 1675.1  1.20  0.123
Models ranked by AICc(x)
```

Три модели, лучшие с точки зрения информационного критерия AICc, включают в качестве независимых переменных прологарифмированные концентрации хлорофилла А (Chla), хлоридов (Cl) и двух модификаций фосфатов (oPO4).

Перейдем теперь к завершающему этапу анализа: построению регрессионной модели, оценке ее адекватности, поиске приемлемых оценок вектора коэффициентов **β** и проверке гипотез относительно этих параметров (п. **41**). Остановимся на модели, включающей установленный выше набор независимых переменных: фактор река и прологарифмированные концентрации хлоридов, фосфатов и хлорофилла.

```
lm.opt <- lm(a1~река+log(Chla)+log(Cl)+log(oPO4),
                                        data=Водоросли)
summary(lm.opt)
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)    51.034      4.231  12.062  < 2e-16 ***
рекасредняя     1.833      2.941   0.623   0.5337
рекамалая       5.128      3.241   1.582   0.1152
log(Chla)      -2.093      0.994  -2.106   0.0365 *
log(Cl)        -3.385      1.351  -2.504   0.0131 *
log(oPO4)      -6.323      1.082  -5.844 2.12e-08 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
Residual standard error: 15.64 on 194 degrees of freedom
Multiple R-squared:  0.4771,    Adjusted R-squared:  0.4636
F-statistic:  35.4 on 5 and 194 DF,  p-value: < 2.2e-16
Anova(lm.opt)
Response: a1
           Sum Sq  Df F value    Pr(>F)
река          653   2  1.3355  0.26544
log(Chla)    1084   1  4.4333  0.03653 *
log(Cl)      1533   1  6.2719  0.01309 *
log(oPO4)    8350   1 34.1556 2.12e-08 ***
Residuals   47427 194
---
```

Полученная модель имеет среднюю для биологических моделей объясняющую силу, оцениваемую по коэффициенту детерминации R^2 = 0.46, и основана на трех статистически значимых гидрохимических показателях. Фактор, определяющий тип реки, оказался незначимым.

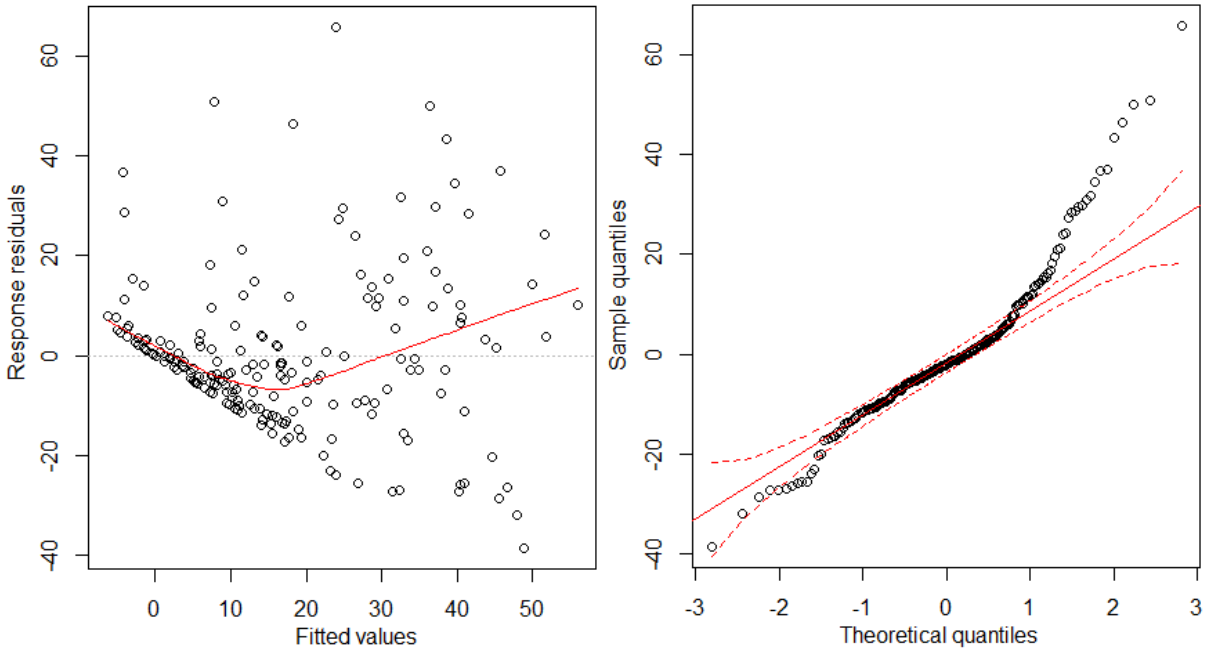
Аналогичная модель, полученная пошаговым алгоритмом на основе непрологарифмированных гидрохимических показателей (см. [8], с. 90), существенно уступает модели lm.opt по основным характеристикам (*RSE*, R^2, AIC_c)

```
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) 55.7204080 20.5365454   2.713  0.00727 **
рекасредняя  3.3401764  3.3721698   0.991  0.32316
рекамалая   10.9864235  3.7636536   2.919  0.00393 **
mxPH        -3.7829230  2.4260402  -1.559  0.12056
NO3         -1.5247600  0.4931425  -3.092  0.00228 **
NH4          0.0014816  0.0009336   1.587  0.11416
PO4         -0.0691868  0.0102496  -6.750 1.68e-10 ***
---
```

```
Residual standard error: 17.49 on 193 degrees of freedom
Multiple R-squared: 0.3494,     Adjusted R-squared: 0.3292
  F-statistic: 17.27 on 6 and 193 DF,  p-value: 5.882e-16
```

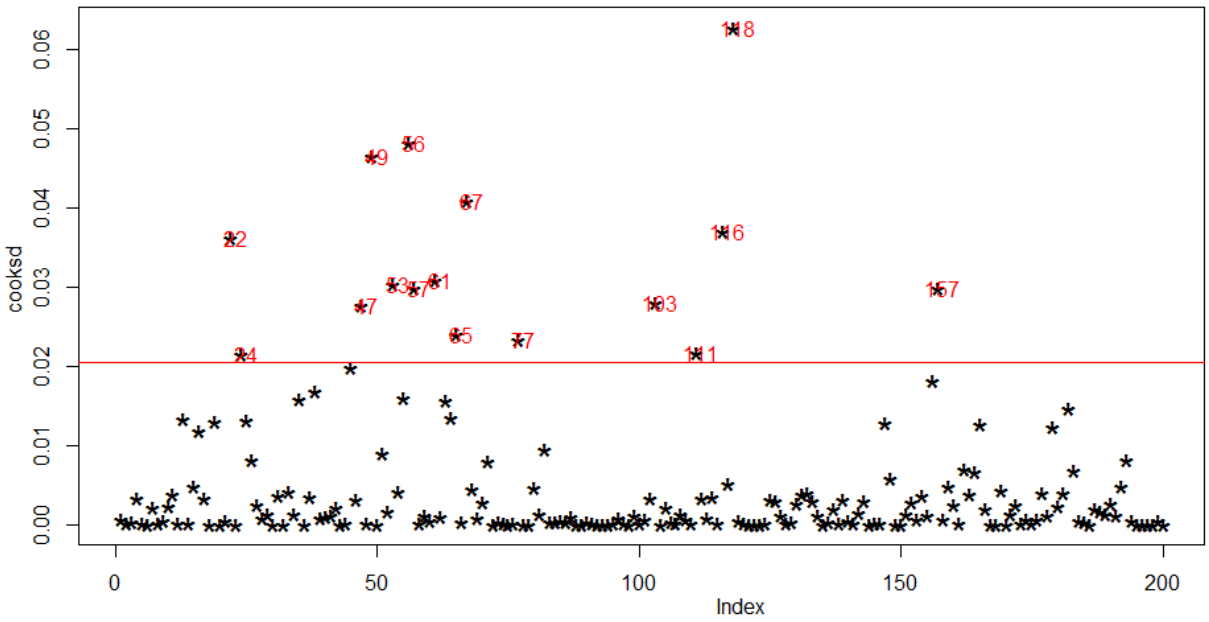
Выполним проверку адекватности модели `lm.opt` (подробнее см. раздел 7.7 [6]). Дисперсия остатков модели не является гомоскедастичной: на графике слева видно, что вариация остатков увеличивается с ростом биомассы водорослей. Распределение остатков также не соответствует нормальному распределению (см. квантиль-квантильную диаграмму справа):

```
library(RVAideMemoire)
plotresid(lm.opt)
```



Можно проанализировать, как каждое из наблюдений влияет на общий прогнозируемый результат, для чего используется расстояние Кука (Cook's distance). Оно оценивает среднее изменение значений прогнозируемого отклика, если исключить из таблицы данных для подгонки модели i-е наблюдение: $D_i = \sum_j (\hat{Y}_j - \hat{Y}_{j(i)})^2 / p \cdot MSE$, где $\hat{Y}_j$ и $\hat{Y}_{j(i)}$ – значение прогноза до и после исключения i-й строки данных, p – количество параметров, MSE – среднеквадратичная ошибка. Если извлечь и исследовать каждую из наиболее влияющих строк исходных данных, то можно понять причину «дрейфа» остатков модели (например, какая-то из переменных X, включенных в модель, имеет экстремальные значения). В адекватной модели все расстояния Кука должны быть примерно одинаковыми. Поэтому те наблюдения, для которых расстояние Кука превышает среднее значение в 4 раза, можно классифицировать как экстремально влияющие (разумеется, это не жесткая граница).

```
cooksd <- cooks.distance(lm.opt)
plot(cooksd, pch="*", cex=2)
abline(h = 4*mean(cooksd), col="red")
text(x=1:length(cooksd)+1, y=cooksd,
      labels=ifelse(cooksd>4*mean(cooksd),names(cooksd),""),
      col="red")  # расставляем метки
```



Важнейшей и единственно объективной характеристикой качества модели является ошибка ее предсказания на объектах, не участвующих в построении самой модели. Подробное объяснение процедуры тестирования моделей с помощью функции `train` из пакета `caret` представлено в [8].

```
library(caret)
train(a1~., data=P_мод, method='glm', family = gaussian,
        trControl = trainControl(method = "cv"))
Resampling: Cross-Validated (10 fold)
  RMSE      Rsquared   MAE
  15.51392  0.4980113  11.21432
```

В ходе тестирования все множество наблюдений случайным образом разбивалось на 10 частей по 20 строк, после чего последовательно 10 раз каждые 9 частей использовалось для построения модели, а одна часть – для тестирования. Ошибка при кросс-проверке `RMSE` является независимой и объективной оценкой устойчивости модели: для хорошо сконструированной модели `RMSE` незначительно (например, менее, чем на 20%) превышает стандартные отклонения для остатков модели, построенной на полных данных. В нашем случае внутренняя ошибка регрессии даже превышает ошибку на внешнем дополнении (15.64 против 15.51) из-за разницы в числе степеней свободы.

Таким образом, несмотря на высокую статистическую значимость модели в целом и большинства ее коэффициентов, графический анализ остатков свидетельствует о ее неполной адекватности. Причины невыполнения предположений о гомоскедастичности и нормальности распределения ошибок регрессии связаны, вероятнее всего, с асимметричностью распределения значений биомассы водорослей, имеющимися выбросами гидрохимических показателей и их высокой мультиколлинеарностью. Возможные варианты решения этой проблемы таковы:

- разделить исходную таблицу на две части (например, с биомассой водорослей до и после 30 мг/л) и построить две модели для каждого диапазона значений отклика;
- подобрать такое преобразование биомассы, которое бы обеспечило симметричность распределения;
- использовать робастные методы построения регрессионных моделей;
- сформировать модель в предположении об ином типе распределения (например, Пуассона или отрицательного биномиального).

43. Метод, используемый в пакете `lsmeans`, было бы целесообразно интерпретировать как «анализ средних предсказаний модели на основе МНК». *Средние наименьшие квадраты* – это обычные средние значения эталонных предсказаний линейной модели, полученные на тривиальной регулярной сетке. Опорная или эталонная сетка (*reference grid*) определяет для каждого предиктора модели набор из одного или нескольких базовых уровней значений предикторов. Если иное не указано явно, то по умолчанию устанавливаются следующие уровни:

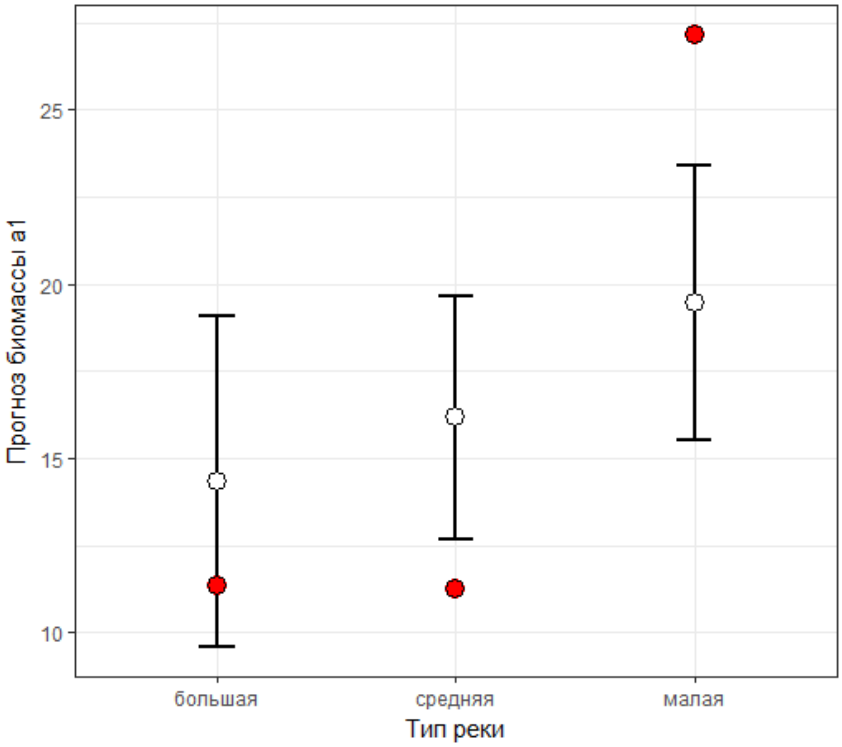
- для каждого предиктора, являющегося фактором, его базовые значения являются уникальными уровнями этого фактора;
- каждый числовой предиктор (ковариата) имеет только один опорный уровень, равный его среднему значению в наборе данных.

```
P_мод <- cbind(Водоросли[,c(2,12)], log(Водоросли[,c(6,9,11)]))
lm.opt <- lm(a1~., data=P_мод)
library(lsmeans)
 (ref.w <- ref.grid(lm.opt))
'ref.grid' object with variables:
    река = большая, средняя, малая
    Cl = 3.1988
    oPO4 = 3.5192
    Chla = 1.7314
```

После того, как опорная сетка установлена, оцениваются средние прогнозы по этой сетке и формируются таблицы, включающие сами прогнозы и их интервальные оценки. Для простых моделей с количественными переменными средние прогнозы совпадают со средними, вычисленными по базовым значениям ковариат. Для несбалансированных факторных наблюдений средний прогноз для каждого уровня фактора определяет средний основной эффект, скорректированный на возможную величину дисбаланса числа наблюдений.

Если выполняются предположения относительно распределения ошибки ϵ в модели линейной регрессии, то для заданного набора значений $\mathbf{X}$ интервальная оценка среднего значения зависимой переменной y является ее доверительным интервалом `CL`. Построим график `CL` прогнозов отклика для средних значений предикторов по каждому уровню фактора `река`. Для сравнения вычислим средние значения `obs` по исходной таблице наблюдений и покажем их на графике красными точками.

```
lsmeans.w <- lsmeans(ref.w, "река")
 Confidence level used: 0.95
(lsm.out <- cbind(data.frame(
    summary(lsmeans.w, type="response")),
    obs = with(Водоросли, tapply(a1, река, mean))))
     река   lsmean       SE  df  lower.CL upper.CL      obs
1 большая 14.33297 2.403731 194  9.592172 19.07377 11.35333
2 средняя 16.16627 1.760166 194 12.694748 19.63778 11.26786
3   малая 19.46127 2.004142 194 15.508562 23.41397 27.14507
library(ggplot2)
ggplot() + labs(x = 'Тип реки', y = 'Прогноз биомассы a1') +
   geom_errorbar(data=lsm.out, mapping=aes(x=река,
       ymin= lower.CL, ymax=upper.CL), width=0.15, size=1) +
   geom_point(data=lsm.out, mapping=aes(x=река, y=lsmean),
       size=4, shape=21, fill="white") +
   geom_point(data=lsm.out, mapping=aes(x=река, y=obs),
       size=4, shape=21, fill="red") + theme_bw()
```



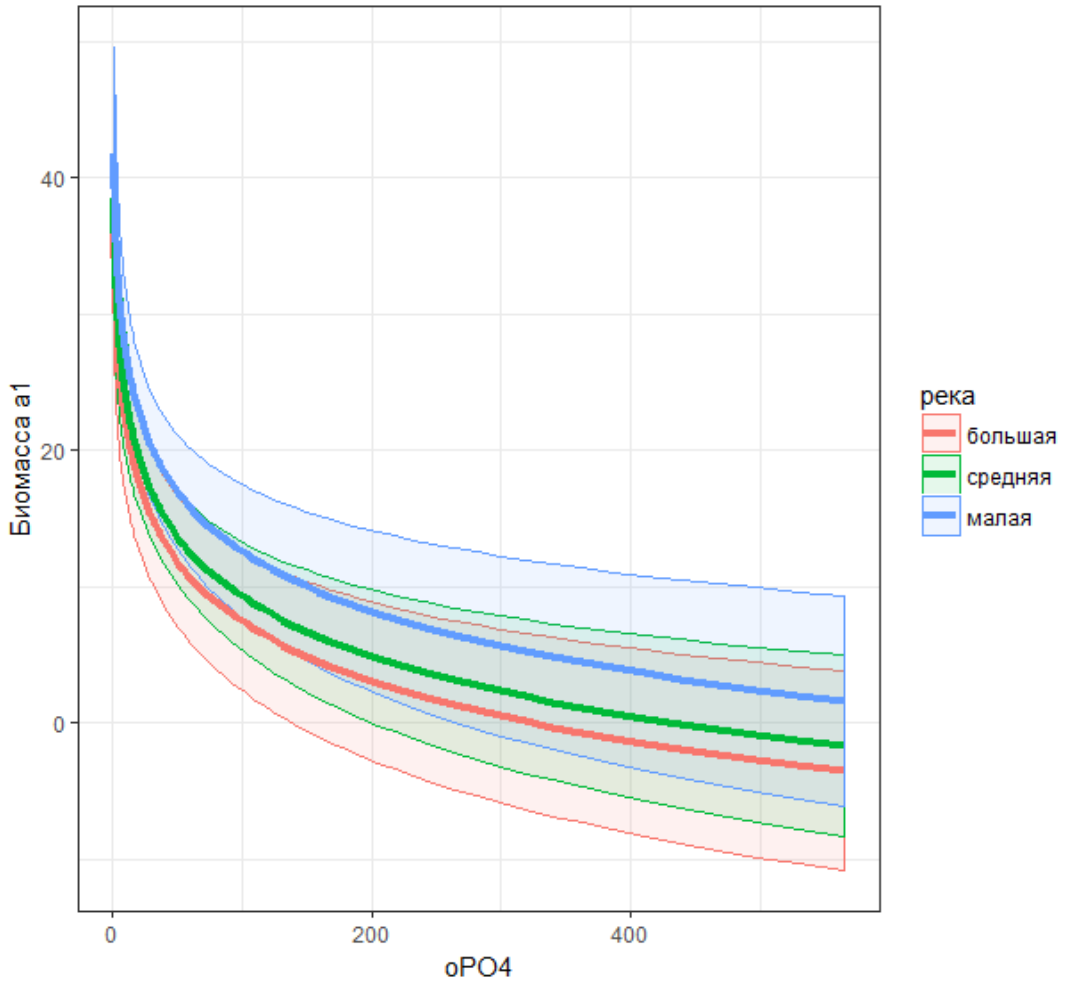
Функция `contrast()` пакета `lsmeans` позволяет выполнить парные сравнения модельных средних 12 различными методами, задаваемыми параметром `method`, в том числе:

- `"pairwise"`,`"revpairwise"` и `"tukey"` выполняет все возможные парные сравнения между оценками средних для каждого уровня фактора; по умолчанию используется метод множественной коррекции уровней значимости `"tukey"`;
- `"poly"` создает ортогональные полиномиальные контрасты для сравнений средних по уровням упорядоченных факторов (при условии равных расстояний между уровнями);
- `"trt.vs.ctrl"` выполняет сравнения одного уровня (контрольного, обозначенного аргументом `ref`) с каждым из остальных уровней (т. е. с некоторым воздействием); по умолчанию используется метод множественной коррекции уровней значимости `"dunnettx"`;
- `"eff"` и `"del.eff"` сравнивают каждый уровень со средним по всем уровням (`"eff"`) или со средним по всем другим уровням (`"del.eff"`); по умолчанию используется метод множественной коррекции уровней значимости `"fdr"`.

```
contrast(lsmeans.w, "pairwise")
 contrast           estimate       SE  df t.ratio p.value
 большая - средняя -1.833294 2.940594 194  -0.623  0.8074
 большая - малая   -5.128294 3.240858 194  -1.582  0.2557
 средняя - малая   -3.295000 2.775682 194  -1.187  0.4623
P value adjustment: tukey method for for 3 tests
contrast(lsmeans.w, "eff")
 contrast         estimate       SE  df t.ratio p.value
 большая effect -2.3205292 1.843791 194  -1.259  0.3146
 средняя effect -0.4872356 1.570557 194  -0.310  0.7567
 малая effect    2.8077647 1.756513 194   1.598  0.3146
P value adjustment: fdr method for 3 tests
```

Для того, чтобы графически отобразить зависимость отклика `a1` от ковариаты `oPO4` для разных уровней фактора `река` с учетом доверительных интервалов, создадим массив, состоящий из последовательности 100 значений переменной `oPO4` и сформируем сетку из 100*3 ячеек:

```
library(RVAideMemoire)
x <- seq2(log(Водоросли$oPO4), int = 100)
ref.w <- ref.grid(lm.opt,at=list(oPO4=x))
lsmeans.w <- lsmeans(ref.w, "река", by = "oPO4")
df <- data.frame(summary(lsmeans.w, type="response"))
library(ggplot2)
p <- ggplot(df, aes(x=exp(oPO4), y=lsmean, colour = река))
p + geom_ribbon(data=df, aes(y=lsmean, ymin=lower.CL,
    ymax=upper.CL, fill=река), alpha=0.1) +
  geom_line(data=df, aes(y=lsmean, colour=река), size=1.5) +
  labs(x="oPO4", y="Биомасса a1") +   theme_bw()
```



44. В условиях высокой коррелированности предикторов процедура построения модели не всегда приводит к корректным результатам. Следствием мультиколлинеарности является плохая обусловленность матрицы $\mathbf{X}^T\mathbf{X}$, ее определитель становится близок к нулю, в результате чего оценки коэффициентов модели и их дисперсий становятся неустойчивыми. При соответствующем повышении тесноты линейной связи между факторами интерпретация их влияния на отклик часто становится приближённой, грубой или даже неверной. Сами по себе точечные оценки

(численные значения) коэффициентов b_i уже не могут служить однозначной мерой направления и силы влияния соответствующих переменных. Например, если исключить из модели один из предикторов, входящий в состав мультиколлинеарного подмножества, то оставшиеся коэффициенты b_i часто могут не только сильно измениться по величине, но и поменять свой знак на противоположный, вследствие чего бывает трудно установить истинное направление влияния соответствующих переменных X_i.

Степень мультиколлинеарности многомерного комплекса переменных можно оценить с помощью фактора инфляции дисперсии (*Variance Inflation Factor*, VIF), вычисляемого для каждого предиктора.

```
ГХП_кор <- cbind(Водоросли[,4:5],log(Водоросли[,6:11]))
P_кор <- cbind(ГХП_кор,a1 = Водоросли[,12])
library(car)
print(vif(glm(a1~., data=P_кор)),4)
 mxPH  mnO2    Cl   NO3   NH4  oPO4   PO4  Chla
1.671 1.611 2.808 1.773 1.888 7.055 8.716 1.633
```

Превышение VIF над некоторым пороговым значением (обычно это критическое значение принимается равным 5) свидетельствует о наличии проблемы мультиколлинеарности для X_j. Эти переменные обычно рекомендуется удалить из модели.

Выполнить селекцию информативного комплекса независимых переменных и одновременно в некоторой степени компенсировать негативное влияние выбросов позволяют робастные методы построения моделей (см. [8, с. 96]). Они заключаются во введении дополнительного слагаемого регуляризации в функционал оптимизации модели, что часто позволяет получать более устойчивое решение. Смысл процедуры заключается в "стягивании" в ходе настройки вектора коэффициентов $\boldsymbol{\beta}$ таким образом, чтобы они в среднем оказались несколько меньше по абсолютной величине, чем это было бы при оптимизации по МНК. В методе регрессии "лассо" (*LASSO, Least Absolute Shrinkage and Selection Operator*) условие минимизации квадратов ошибки при оценке параметров $\hat{\beta}$ выражается следующей формулой:

$$\hat{\beta} = \arg\min[\sum_{i=1}^{n}(y_i - \sum_{j=1}^{m}\beta_i x_{ij})^2 + \lambda\,|\,\boldsymbol{\beta}\,|],$$

где λ – параметр регуляризации, имеющий смысл штрафа за сложность. Аналогичный метод гребневой ("ридж") регрессии вместо абсолютных значений вектора коэффициентов $\boldsymbol{\beta}$ использует их квадраты.

При значении параметра регуляризации $\lambda = 0$, робастная регрессия сводится к обычному методу наименьших квадратов, а при увеличении λ формируемая модель становится все более "лаконичной", пока не превратится в нуль-модель. Оптимальная величина λ находится с использованием перекрестной проверки, т. е. ей соответствует минимальная ошибка прогноза $\widehat{y}_i$ на наблюдениях, не участвовавших в построении самой модели.

Построим модель оптимальной сложности на основе регуляризации с использованием функции `train()` из пакета `caret`. Для метода `glmnet` эта функция выполняет оптимизацию для двух моделей регуляризации: при `alpha` = 1 подгоняется модель по методу лассо, а при `alpha` = 0 – гребневая регрессия. Осуществим построение лассо-модели и, поскольку точное значение `lambda` нам неизвестно, построим 10 моделей в интервале λ от 0.5 до 2, после чего примем наилучшую из них в соответствии с минимумом ошибки регрессии при кросс-проверке. Все три фактора

сезон,река,течение преобразуем в бинарную форму с использованием функции model.matrix:

```
x <- model.matrix(a1 ~ .,data=Водоросли[,c(1:5,12)])[,-1]
x <- cbind(x, log(Водоросли[,6:11]))
grid.train = seq(0.5,2,length=10)
library(caret)
lasso.a1.train <- train(as.data.frame(x), Водоросли$a1,
    method='glmnet',
    tuneGrid = expand.grid(.lambda = grid.train, .alpha = 1),
    trControl = trainControl(method = "cv"))
Resampling: Cross-Validated (10 fold)
  lambda     RMSE      Rsquared   MAE
  0.5000000  15.70948  0.4598771  11.21274
  0.6666667  15.62003  0.4659426  11.16961
  0.8333333  15.55489  0.4708289  11.13706
  1.0000000  15.51004  0.4746440  11.13147
  1.166667   15.50639  0.4757183  11.14632
  1.3333333  15.51900  0.4758157  11.16641
  1.5000000  15.54035  0.4753622  11.19542
  1.6666667  15.56722  0.4747737  11.22659
  1.8333333  15.60255  0.4738394  11.26183
  2.0000000  15.64426  0.4726571  11.30227
The final values used for the model were alpha = 1 and lambda
= 1.166667.
```

Выведем протокол со значениями коэффициентов модели:

```
coef(lasso.a1.train$finalModel,
          lasso.a1.train$bestTune$lambda)
 (Intercept)      55.3655687
сезонвесна         .
сезонлето          .
сезонзима          .
рекасредняя        .
рекамалая          1.9935372
течениемедленное   .
течениесреднее     .
mxPH               .
mnO2               .
Cl                -1.7540595
NO3                .
NH4               -0.5342783
oPO4              -3.4015630
PO4               -3.6948378
Chla              -1.7307055
```

Мы получили робастную модель примерно с одинаковой ошибкой кросс-проверки, что и обычная линейная модель и с тем же составом "полезных" коэффициентов. Существует мнение, что использование моделей с регуляризацией позволяет избежать проблем мультиколлинеарности. Это справедливо только в той части, что улучшается обусловленность матриц и оценки коэффициентов становятся более устойчивыми. Но регуляризация не всегда влияет на выбор оптимального состава предикторов: наша модель включила в себя оба сильно коррелированных показателя oPO4 и PO4.

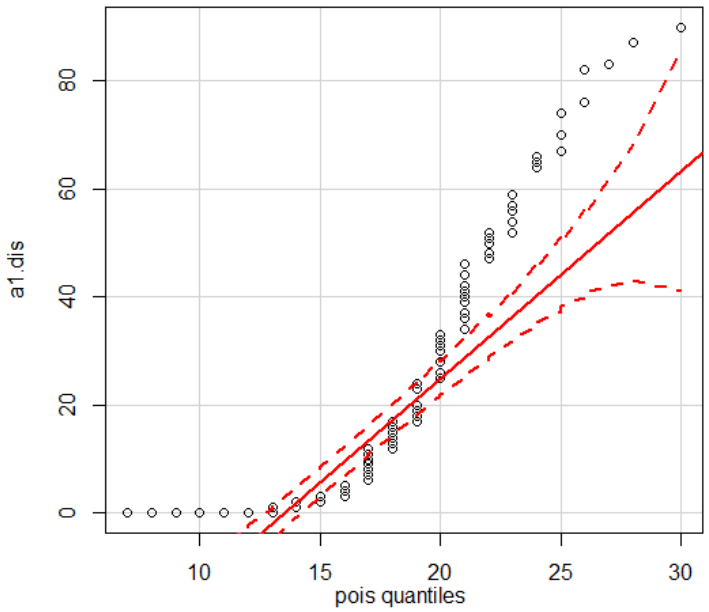
53. Результаты теста на адекватность в вышеприведенных комментариях заставляют нас усомниться в правильности предположений о нормальном распределении остатков при построении линейной модели зависимости биомассы водорослей от гидрохимических показателей. Поскольку биомасса – непосредственная функция от числа обнаруженных экземпляров, то рассмотрим возможность использования при построении дискретных распределений, в первую очередь – распределения Пуассона.

Оценим выборочные параметры для распределения Пуассона и выполним тест Крамера – фон Мизеса:

```
a1.dis <- round(Водоросли$a1,0)
library(MASS)
parms <- fitdistr(a1.dis, "poisson")
    lambda
  16.9050000
 (0.2907318)
lambda <- parms$estimate
t.test(a1.dis)$conf.int
[1] 13.92244 19.88756
library(goftest)
library(RVAideMemoire)
cvm.test(a1.dis, cdf.discrete(a1.dis, "pois", lambda))
        Cramer-von Mises test of goodness-of-fit
data:  a1.dis
omega2 = 14.354, p-value = 8.834e-10
```

Нулевая гипотеза о принадлежности выборки к распределению Пуассона отклоняется. Этот вывод также можно подтвердить, построив квантиль-квантильный график:

```
library(car)
qqPlot(a1.dis,dist="pois",lambda)
```



55. Рассмотрим методику построения моделей на основе предположения о дискретном распределении данных. Построим вначале обобщенную линейную модель с гауссовым распределением, которая отличается от обычной линейной модели (см. комментарий к п. **41**) только формой итогового резюме:

```
P_мод <- cbind(Водоросли[,c(2,12)],log(Водоросли[,c(6,9,11)]))
m.norm <- glm(a1~., data=P_мод, family="gaussian")
summary(m.norm)
    Null deviance: 90695  on 199  degrees of freedom
Residual deviance: 47427  on 194  degrees of freedom
AIC: 1675.3
```

Модели на основе пуассоновского и квазипуассоновского распределений отличаются между собой только тем, что *p*-величины значимости коэффициентов модели во втором случае оцениваются значительно реалистичней:

```
m.p <- glm(a1~., data=P_мод, family="poisson")
AIC(m.p)
AIC: 3202.6
m.qp <- glm(a1~., data=P_мод, family="quasipoisson")
summary(m.qp)
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  4.31810    0.23852  18.104   <2e-16 ***
рекасредняя  0.23834    0.22130   1.077   0.2828
рекамалая    0.26051    0.21493   1.212   0.2270
Cl          -0.19557    0.07905  -2.474   0.0142 *
oPO4        -0.33321    0.06348  -5.249    4e-07 ***
Chla        -0.12834    0.06273  -2.046   0.0421 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
(Dispersion parameter for quasipoisson family taken to be
15.47304)
    Null deviance: 4847.1  on 199  degrees of freedom
Residual deviance: 2506.3  on 194  degrees of freedom
AIC: NA
```

Однако дисперсионный параметр, равный 15.5, очень высок, что не позволяет использовать эту модель в качестве рабочей.

Выполним теперь построение модели на основе отрицательного биномиального распределения (`"nbinom"`), для чего воспользуемся функцией `glm.nb()` из пакета `MASS`:

```
library(MASS)
m.nb <- glm.nb(a1~., data=P_мод)
summary(m.nb)
Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  4.67912    0.31566  14.823  < 2e-16 ***
рекасредняя  0.06540    0.22434   0.292   0.7706
рекамалая    0.23161    0.24480   0.946   0.3441
Cl          -0.20622    0.10125  -2.037   0.0417 *
oPO4        -0.44120    0.08146  -5.416 6.09e-08 ***
Chla        -0.08633    0.07501  -1.151   0.2498
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
(Dispersion parameter for Negative Binomial(0.784) family taken
to be 1)
    Null deviance: 352.45  on 199  degrees of freedom
Residual deviance: 234.07  on 194  degrees of freedom
AIC: 1403.5
```

```
              Theta:  0.7840
          Std. Err.:  0.0901
 2 x log-likelihood:  -1389.5210
```

Отметим, что полученная модель значительно лучше по AIC-критерию, чем гауссова модель `m.norm`.

Выше нами отмечалось, что важнейшей и наиболее объективной характеристикой качества модели является ошибка ее предсказания на объектах, не участвовавших в построении самой модели. Для негативной биномиальной модели с помощью функции `train` из пакета `caret` можно протестировать также возможные функции связи `link`.

```
library(caret)
train(a1~., data=P_мод, method='glm.nb',
      trControl = trainControl(method = "cv"))
  link       RMSE      Rsquared   MAE
  log        18.04217  0.5385605  12.09403
  sqrt       14.38826  0.5666867  10.09259
```

Очевидно, что эта модель существенно надежнее, чем аналогичная гауссова модель: ошибка регрессии при кросс-проверке `RMSE` уменьшилась с 15.5 до 14.4, а коэффициент детерминации R^2 увеличился до 0.56 против 0.49.

Обычно считается, что непременно следует удалять из модели незначимые независимые переменные. Рассмотрим, к какому результату это приведет в нашем случае:

```
m2.nb <- glm.nb(a1~Cl + oPO4, data=P_мод)
summary(m2.nb)
Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  4.97505    0.24985  19.912  < 2e-16 ***
Cl          -0.24080    0.09494  -2.536   0.0112 *
oPO4        -0.50194    0.07911  -6.344 2.23e-10 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
(Dispersion parameter for Negative Binomial(0.7661) family
taken to be 1)
    Null deviance: 345.93  on 199  degrees of freedom
Residual deviance: 233.25  on 197  degrees of freedom
AIC: 1400.7
              Theta:  0.7661
          Std. Err.:  0.0872
 2 x log-likelihood:  -1392.7080
train(a1 ~ Cl + oPO4, data=P_мод, method='glm.nb',
      trControl = trainControl(method = "cv"))
  link       RMSE      Rsquared   MAE
  log        18.17087  0.4508462  11.83800
  sqrt       15.47209  0.5439898  10.67631
```

Действительно, AIC -критерий несколько уменьшился, но одновременно увеличилась ошибка при кросс-проверке модели. Таким образом, можно полагать, что некоторые члены модели, считающиеся незначимыми, на самом деле играют существенную роль для стабилизации ошибки прогнозов.

102. По мнению Н. Кенкеля [22], при использовании многомерных методов для статистической обработки наблюдений в биологических науках, «*сохраняется значительная неопределённость в выборе оптимальной стратегии анализа, которая включает важные решения относительно преобразования данных, стандартизации переменных и общего методологического подхода. Эта стратегия должна основываться на тщательном и точном учёте цели исследования, структуры обрабатываемых данных и основных статистических предпосылках. К сожалению, даже относительно свежие обзоры многомерных аналитических стратегий основываются в значительной степени на эмпирических представлениях сомнительной достоверности, которые появились в экологической литературе 1970-х годов и некритически увековечены по сей день…*». Н. Кенкель поставил перед собой задачу «*рассеять вводящие в заблуждение рекомендации и ошибочные обобщения относительно эффективности доступных многомерных методов...*».

Что касается «*вводящих в заблуждение рекомендаций*», то некоторые заключения М. Эрве представляются нам излишне категоричными. В первую очередь это касается его весьма негативных взглядов на роль неметрического многомерного шкалирования, которое, как показывает наш опыт [23, 24], является наиболее устойчивым и интерпретируемым методом ординации в различных областях исследований. Рассмотрим подробно и поэтапно особенности его практической реализации.

Отметим, что форма численного представления показателей популяционной плотности в формулах для подсчёта расстояний d должна быть адекватна задачам исследования. Например, биомасса разных видов водных беспозвоночных может различаться в десятки тысяч раз и непосредственное использование этого показателя при формировании матрицы **D** приведёт к тому, что итоговая ординационная диаграмма будет целиком определяться численностью 1–2 видов крупных моллюсков. Если же выполнить обычную стандартизацию, приведя биомассу каждого вида к нулевому среднему и единичной дисперсии, как это предлагает М. Эрве, то это также не будет способствовать выяснению сути дела, поскольку доминирующие и функционально важные виды будут иметь ту же относительную значимость, что и виды-маргиналы, играющие незначительную роль в экосистеме. Поэтому, чтобы "приглушить" чересчур сильную вариацию показателя, используют общие правила построения хорошо интерпретируемых ординаций [25]. В таблице `Водоросли`, используемой ранее для наших примеров, семь столбцов `a1 - a7` представляют биомассу различных таксономических групп водорослей:

```
summary(Водоросли[,12:18])
       a1               a2               a3               a4        
 Min.   : 0.00   Min.   : 0.000   Min.   : 0.000   Min.   : 0.000  
 1st Qu.: 1.50   1st Qu.: 0.000   1st Qu.: 0.000   1st Qu.: 0.000  
 Median : 6.95   Median : 3.000   Median : 1.550   Median : 0.000  
 Mean   :16.92   Mean   : 7.458   Mean   : 4.309   Mean   : 1.992  
 3rd Qu.:24.80   3rd Qu.:11.375   3rd Qu.: 4.925   3rd Qu.: 2.400  
 Max.   :89.80   Max.   :72.600   Max.   :42.800   Max.   :44.600  
       a5               a6               a7        
 Min.   : 0.000   Min.   : 0.000   Min.   : 0.000  
 1st Qu.: 0.000   1st Qu.: 0.000   1st Qu.: 0.000  
 Median : 1.900   Median : 0.000   Median : 1.000  
 Mean   : 5.064   Mean   : 5.964   Mean   : 2.495  
 3rd Qu.: 7.500   3rd Qu.: 6.925   3rd Qu.: 2.400  
 Max.   :44.400   Max.   :77.600   Max.   :31.600
```

Нетрудно увидеть обычный для социологических и биологических наблюдений феномен негауссовости, который заключается в том, что распределение выборок носит отчетливый гипергеометрический характер (см. законы Ципфа-Мандельброта) [26].

Мы рекомендуем выполнять преобразование численности видов, приводящее к χ^2-дистанции, которое является, по всей вероятности, наиболее разумным компромиссом при учёте, как роли ведущих компонент, так и вклада редких (малочисленных) таксонов.

```
A.chi <- decostand(Водоросли[,12:18], "chi.square")
Env <- Водоросли[,4:11]
rankindex(Env, A.chi, c("euc","man","bray","jac","kul"))
      euc       man      bray       jac       kul
0.2830556 0.2910761 0.2642564 0.2642563 0.2650146
```

Функция `rankindex` для различных метрик (евклидовой, манхэттенской, Брея-Кёртиса, Жаккара и Кульчицкого) рассчитала коэффициенты корреляции Спирмена между двумя таблицами: семью гидрохимическими показателями `Env` и трансформированними биомассами сообщества водорослей `A.chi`.

Задача неметрического многомерного шкалирования состоит в том, чтобы создать такую *p*-мерную (*p* = 2 и более) "проекцию" изучаемых объектов, на которой взаимные попарные расстояния оказались бы наименее искажены по сравнению с исходным состоянием **D**. Главные оси геометрической метафоры данных в пространстве меньшей размерности находятся путем минимизации критерия "стресса": $\Delta = \sum_{i,j=1}^{n} d_{ij}^{\alpha} \left| \hat{d}_{ij} - d_{ij} \right|^{\beta}$, где d_{ij} и $\hat{d}_{ij}$ – расстояния между объектами i и j в исходном и редуцированном пространствах, α и β – задаваемые коэффициенты. Для поиска оптимального решения методом nMDS выбирается некоторая начальная конфигурация точек и выполняются последовательные приближения для достижения максимального сходства между исходной и моделируемой матрицами расстояний (с точки зрения критерия стресса).

```
A.mds <- metaMDS(A.chi,distance = "man", trace = FALSE)
global Multidimensional Scaling using monoMDS
Data:     A.chi
Distance: manhattan
Dimensions: 2
Stress:     0.2165243
Stress type 1, weak ties
No convergent solutions - best solution after 20 tries
A.mds <- MDSrotate(A.mds, Env$oPO4)
```

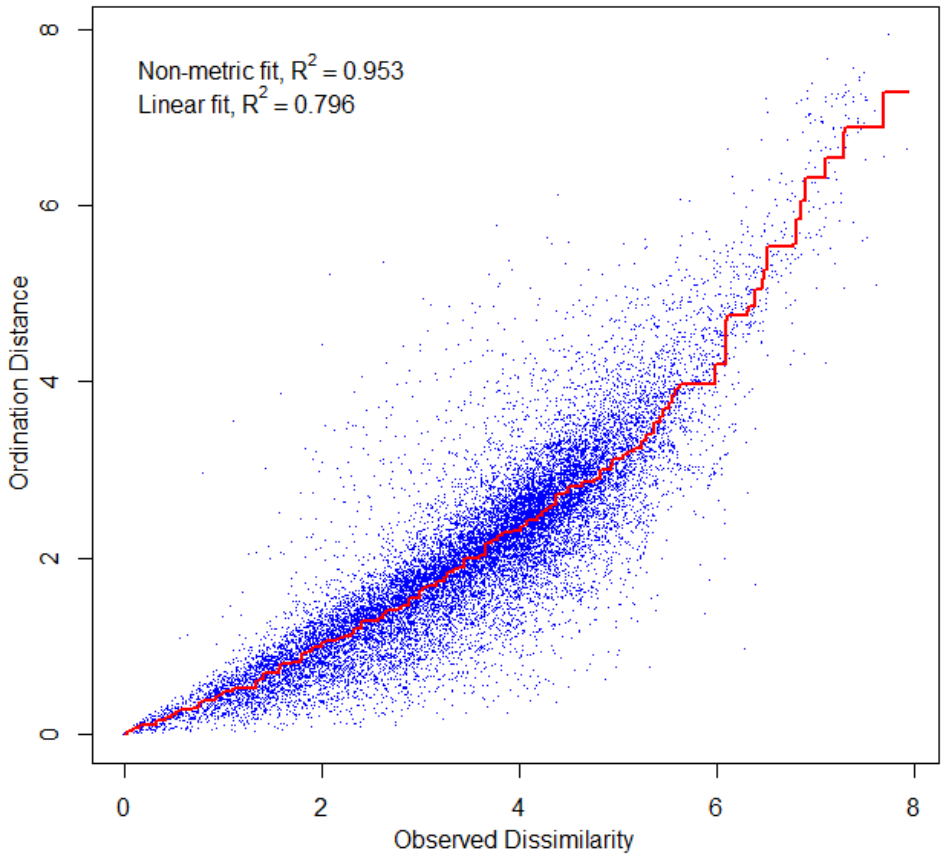
Разработчики функции `metaMDS` (J.Oksanen) сделали все возможное, чтобы алгоритм эффективно сходился в окрестностях глобального минимума стресса (`Stress=0.216`). Оптимизация стартует с линейной метрической модели (т. е. с ординации PCoA, которая часто находит достаточно хорошее решение), после чего выполняется итерационная процедура спуска по градиенту. Чтобы алгоритм не "застрял" в локальных минимумах, процесс повторяет 20 (по умолчанию) циклов, стартуя каждый раз с наилучшего решения, достигнутого на предыдущих шагах.

Оси полученной ординации могут быть ориентированы любым заданным образом: в нашем случае задано совмещение оси NMDS1 с направлением оси одного из внешних факторов – концентрации ортофосфатов из таблицы `Env`.

Стресс на самом деле является обобщенным индикатором сгущения точек вокруг кривой диаграммы Шепарда. При этом функция `stressplot` показывает два коэффициента корреляции, играющих роль показателя качества подгонки. Корреляция, основанная на стрессе $R^2 = 1\text{-}S^2$ – это корреляция между подогнанными значениями $\hat{d}_{ij}$

и фактическими расстояниями, т. е. между точками и ступенчатой линией. Линейная корреляция предполагает, что эти подогнанные значения лежат на прямой линии:

```
stressplot(A.mds)
```



«*Результаты неметрического многомерного шкалирования nMDS имеют полуколичественный (т. е. относительный) характер…поскольку nMDS использует не фактические расстояния, а только их полуколичественные оценки (ранги)*» /М. Эрве/.

Из изложенного ясно, что координаты точек объектов в осях ординации nMDS и расстояния между этими точками нельзя расценивать как «*полуколичественные или относительные*». Можно предположить, что это мнение М. Эрве основывается на статье Р.Шепарда (Shepard, 1962), в которой, действительно, при выполнении итераций оценивается порядок рангов. Но уже немного позднее Дж.Краскел (Kruskal, 1964) описал другой более общий алгоритм подгонки данных: моделируемые расстояния предполагаются монотонно связанными с исходными дистанциями в любом пространстве Минковского.

«*Расстояния, наблюдаемые на диаграмме, сформированной PCoA, могут быть непосредственно интерпретированы как расстояния, представленные в матрице дистанций, тогда как для ординации, выполненной nMDS, единственная доступная информация имеет тип "это расстояние больше, чем иное*" /М. Эрве/.

Действительно, поиск решения в PCA/PCoA осуществляется на множестве линейных функций (с точностью до ортогональных преобразований) и основан на операциях с собственными числами и собственными векторами. Это означает, что можно аналитически точно оценить расстояние между точками в пространстве, например, двух главных осей. Но это ординационное расстояние никак не соответствует фактическому расстоянию между объектами уже хотя бы потому, что эти две главные оси объясняют чаще всего лишь 45-65% общего статистического разброса. Метод nMDS не является аналитически точным, но воспроизводит ординацию на

основе априори заданного числа осей (на практике 2 или 3), поэтому интерпретируется *целиком вся имеющаяся* информация, причем, с учетом нелинейного характера имеющихся зависимостей. Таким образом, достаточно очевидно, что неметрическое шкалирование, по сравнению с иными методами ординации, выполняет оптимальное проецирование произвольных матриц дистанций в пространство малой размерности с минимально возможными искажениями фактических расстояний.

«*В матрице расстояний нет больше никакой информации о возможных исходных переменных, поэтому единственным графическим представлением является диаграмма ординации объектов*…» /М. Эрве/.

Поскольку с каждым объектом в исходной таблице связаны конкретные значения переменных (биомассы отдельных таксонов водорослей), то не представляет труда оценить координаты каждого таксона в пространстве ординационных шкал методом "взвешенных средних". Достаточно вспомнить из физики задачу о нахождении центра тяжести сложной фигуры. Именно эту процедуру выполняет функция `wascores` и вызывающая ее "в темную" функция `metaMDS`.

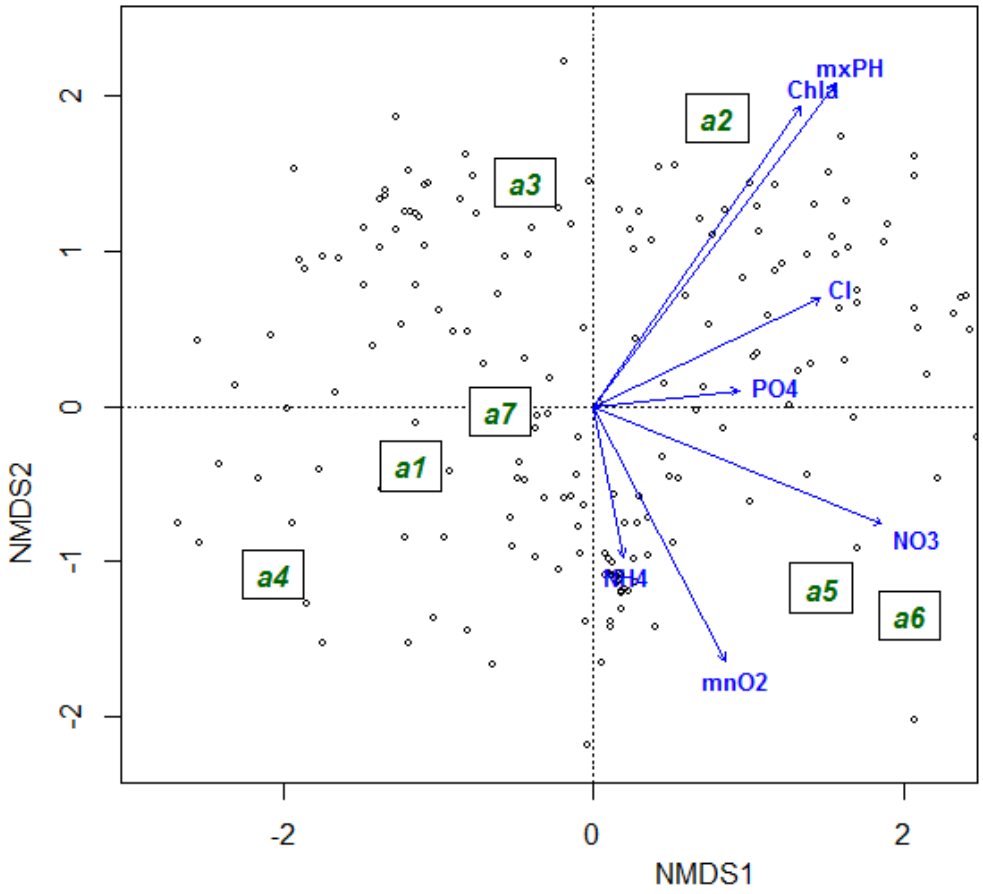
«*RDA – это единственный способ корректно интерпретировать результаты о связи многомерного отклика с внешними переменными*.» /М. Эрве/.

Функция `envfit` оценивает направление вектора внешнего фактора для любой ординации, в том числе, для nMDS. Мы не вполне согласны с утверждением, что нахождение корреляций (см. п. **91**) между той или иной ординацией и внешним фактором менее корректно, чем построение модели PCA с инструментальными переменными.

```
ef <- envfit(A.mds, Env, permu = 999)
***VECTORS
        NMDS1    NMDS2     r2 Pr(>r)
mxPH  0.60078  0.79942 0.2037  0.001 ***
mnO2  0.46144 -0.88717 0.1029  0.001 ***
Cl    0.90378  0.42801 0.0791  0.002 **
NO3   0.92582 -0.37795 0.1215  0.001 ***
NH4   0.19301 -0.98120 0.0297  0.040 *
oPO4  1.00000  0.00000 0.0125  0.297
PO4   0.99404  0.10901 0.0270  0.060 .
Chla  0.56925  0.82217 0.1668  0.001 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
Permutation: free
Number of permutations: 999
```

Вызывает неподдельный интерес то обстоятельство, что такой ключевой показатель `oPO4`, подавляющий рост водорослей `a1`, оказался статистически незначимым в рамках всего сообщества. На основе полученных результатов мы можем построить "триплот", т. е. диаграмму, объединяющую распределение точек объектов и видов, а также дополнительные оси факторов (со статистической значимостью менее 0.1).

```
plot(A.mds,  type="n")
abline(h = 0, lty = 3)
abline(v = 0, lty = 3)
points(ord.mds, disp="sites",cex=0.5)
ordilabel(A.mds,display = "sp", font=4,
          col="darkgreen",      add=TRUE, cex=1)
plot(ef, p.max = 0.1, col="blue", font=2, lwd=2, cex=0.8)
```



Значительный практический интерес [27] может представлять аппроксимация значений внешнего фактора, распределенного в пространстве главных шкал, двумерной сглаживающей поверхностью. Для каждого показателя загрязнения воды Y по его эмпирическим значениям в точках наблюдений можно рассчитать трехмерную обобщенную аддитивную модель GAM [28]:

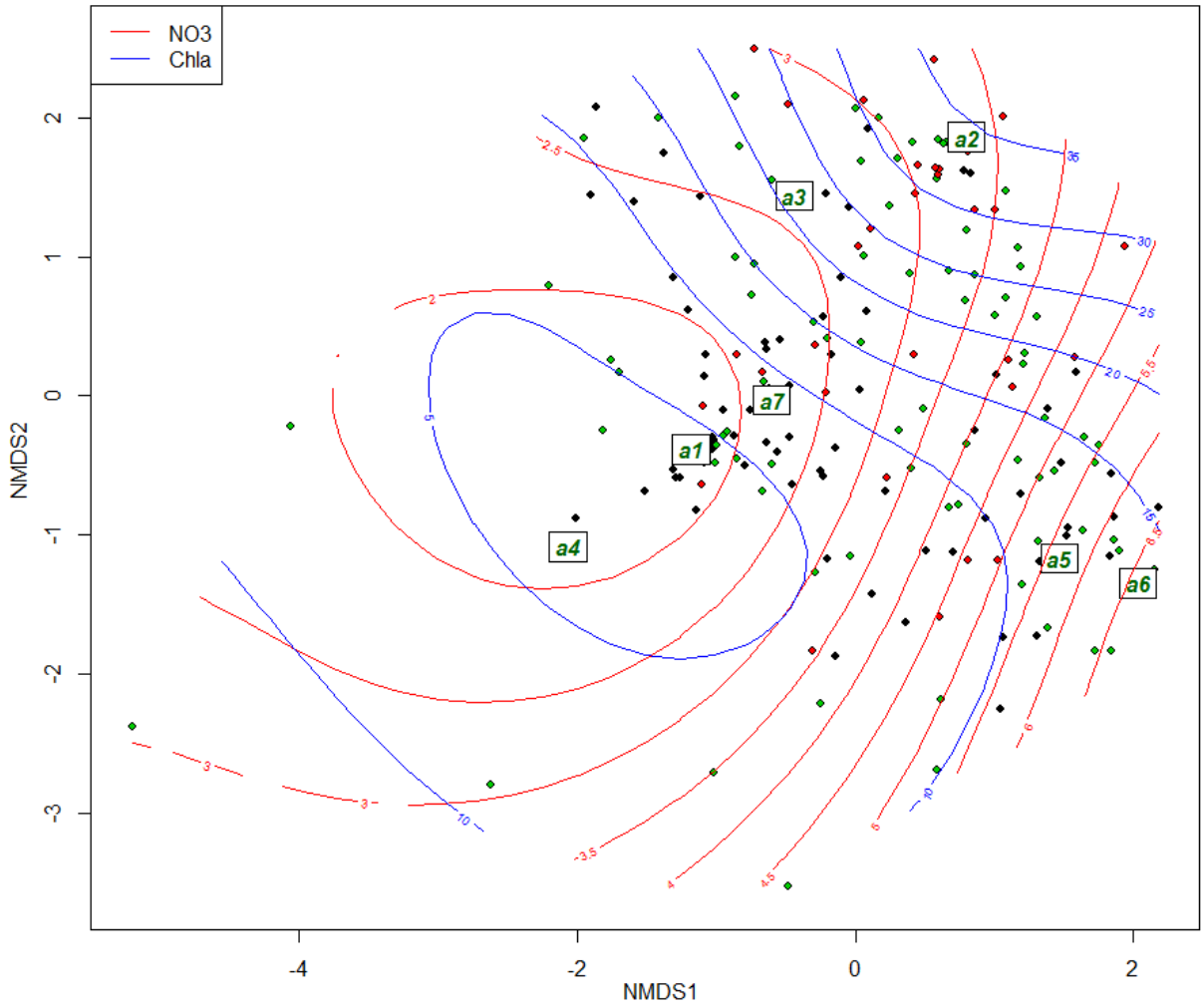
$$Y = \alpha + f_1(s_1) + f_2(s_2) + f_3(s_1, s_2) + \varepsilon,$$

где f_1, f_2, f_3 – сглаживающие функции в форме сплайнов или полиномов от NMS-координат s_1 и s_2.

Эта процедура выполняется функцией `ordisurf`:

```
plot(A.mds,  type="n")
points(A.mds, disp="sites",cex=0.8,pch=21,
        bg=as.numeric(Водоросли$течение))
ordilabel(A.mds,display = "sp", font=4, col="darkgreen",
        add=TRUE, cex=1)
with(Env, ordisurf(A.mds, NO3, add = TRUE, col =2))
with(Env, ordisurf(A.mds, Chla, add = TRUE, col =4))
legend("topleft", c("NO3","Chla"), lwd=1, col=c(2,4))
```

На полученной диаграмме легко усмотреть, что виды водорослей `a1` и `a4` (в отличии от `a2`) предпочитают водотоки с низкими значениями выбранных предикторов. Аналогично можно заключить, что эти показатели чаще имеют высокое значение в водоемах с медленным течением (красные точки), чем с быстрым (черные точки).



103. Выполним расчеты той же ординационной модели с использованием анализа избыточности RDA:

```
A.rda <- rda(A.chi~., data=Env)
summary(A.rda)
Partitioning of variance:
              Inertia Proportion
Total          1.9747     1.0000
Constrained    0.5671     0.2872
Unconstrained  1.4076     0.7128
```

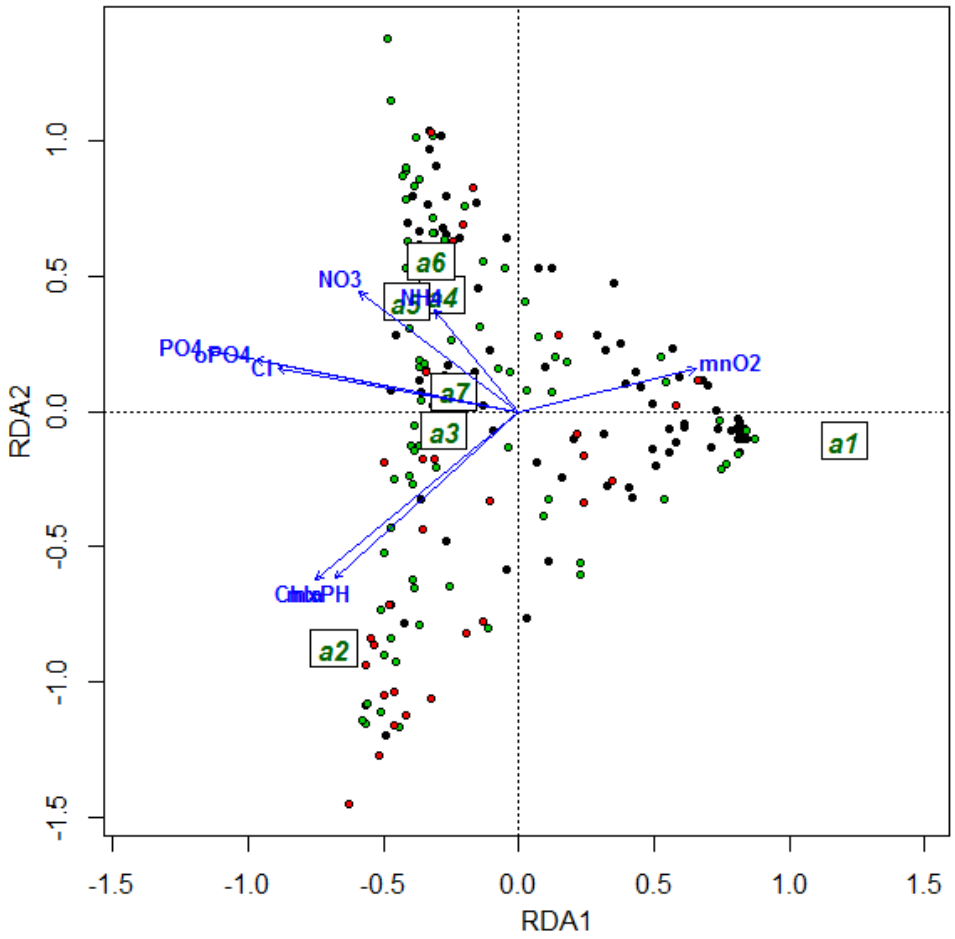
Внешними факторами может быть объяснено 28.7% вариации биомассы водорослей.

```
Importance of components:
                         RDA1    RDA2    RDA3    RDA4    RDA5     RDA6    PC1    PC2
Eigenvalue             0.2396 0.14445 0.12278 0.03264 0.01863 0.008963 0.3518 0.3040
Proportion Explained   0.1213 0.07315 0.06218 0.01653 0.00944 0.004540 0.1782 0.1539
Cumulative Proportion 0.1213 0.19448 0.25666 0.27319 0.28262 0.287160 0.4653 0.6193
```

Дисперсия, объясняемая внешними факторами, раскладывается на шесть главных компонент `RDA1-RDA6`. Часть необъясняемой вариации (33.6%) связывается с главными компонентами `PC1-PC2` (в отличие от nMDS, остальные оси RDA просто отбрасывает). Итого ординация RDA объясняет 61.9% вариации исходных данных.

По результатам дисперсионного анализа с использованием пермутационного теста ординационная модель с инструментальными переменными является статистически значимой и включает все предикторы как значащие:

```
anova(A.rda, by = "term", step=200)
Model: rda(formula = A.chi ~ mxPH + mnO2 + Cl + NO3 + NH4 +
               oPO4 + PO4 + Chla, data = Env)
          Df Variance       F Pr(>F)
mxPH       1  0.11437 15.5183  0.001 ***
mnO2       1  0.12333 16.7342  0.001 ***
Cl         1  0.07715 10.4689  0.001 ***
NO3        1  0.06878  9.3332  0.001 ***
NH4        1  0.03832  5.1999  0.001 ***
oPO4       1  0.05753  7.8056  0.001 ***
PO4        1  0.02754  3.7363  0.004 **
Chla       1  0.06004  8.1474  0.001 ***
Residual 191  1.40763
ef.rda <- envfit(A.rda, Env, permutations = 999)
plot(A.rda,  type="n")
points(A.rda, display = "site", pch=21, cex=0.7,
      bg=as.numeric(Водоросли$течение))
ordilabel(A.rda,display   =   "sp",   font=4,   col="darkgreen",
    add=TRUE, cex=1)
plot(ef.rda, p.max = 0.1, col="blue", font=2, lwd=2, cex=0.8)
```

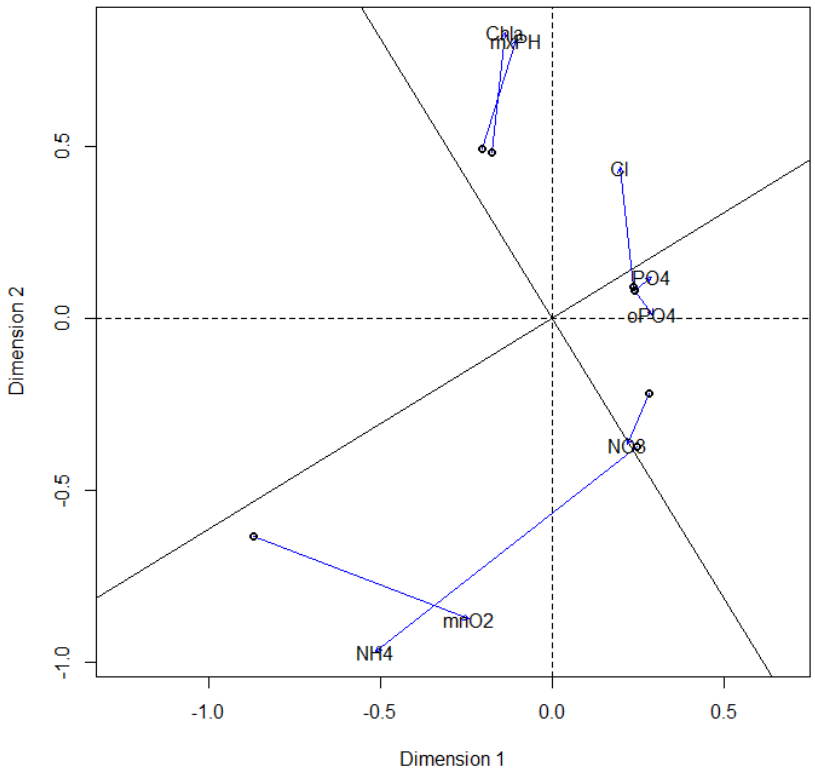


Принципиальное отличие ординаций nMDS и RDA заключается в том, что во втором случае распределение точек на факториальной плоскости определяется как внутренней биологической вариацией, так и влиянием внешних факторов. Однако одинаково высокая значимость всех гидрохимических показателей, скорее всего,

объясняется тем, что метод не слишком хорошо разобрался с эффектом их мультиколлинеарности (а также возможно тем, что сама ординационная структура была сформирована с участием этих внешних факторов). В случае nMDS коэффициенты корреляции отражают тесноту связи внешних факторов только с "чистой" (необъясненной) вариативностью экологической структуры, что может представлять отдельный интерес для объяснения механизмов формирования альгоценозов.

Возникает естественный вопрос: а отличаются ли статистически ординации, полученные этими двумя столь разными методами. Используем для этого прокрустов анализ. Начнем с оценки сдвига концов стрелок гидрохимических показателей:

```
env.mds <- as.data.frame(ef$vectors$arrows)
env.rda <- as.data.frame(ef.rda$vectors$arrows)
plot(prov <- procrustes(env.mds, env.rda))
text(prov, display= "target")
```



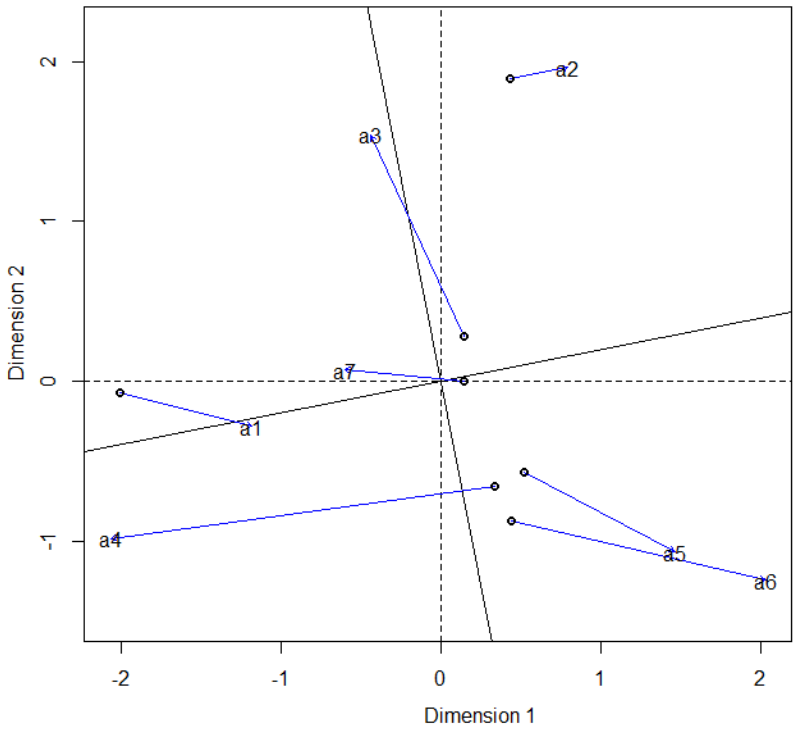

```
protest(env.mds, env.rda)
Procrustes Sum of Squares (m12 squared):        0.4439
Correlation in a symmetric Procrustes rotation: 0.7457
Significance:  0.006
```

На графике видно, что значительно изменилась ориентация только двух осей: растворенного кислорода и ионов аммония. Функция `protest` вычислила средний квадрат расстояний *ss* между точками обеих ординаций и прокрустову корреляцию $r = (1 - ss)^{0.5}$. Если многократно размещать точки этих ординаций на плоскости, используя случайные координаты, то только в 6 случаях из 1000 рандомизированный коэффициент прокрустовой корреляции превысит эмпирическое значение. Это означает, что расположение точек является статистически близким.

Теперь рассмотрим, есть ли различия в конфигурации точек отдельных таксонов водорослей:

```
spec.mds <- as.data.frame(A.mds$species)
spec.rda <- as.data.frame(A.rda$CCA$v[,1:2])
```

```
protest(spec.mds, spec.rda)
Procrustes Sum of Squares (m12 squared):        0.5651
Correlation in a symmetric Procrustes rotation: 0.6595
Significance:  0.12
plot(pro <- procrustes(spec.mds, spec.rda))
text(pro, display= "target")
```



Серьезным сдвигам оказались подвержены таксоны `a4`, `a6` и `a3`. Пермутационный тест показал, что между двумя структурами имеются статистически значимые отличия. Какая из ординаций лучше отражает существо дела? Видимо ответ на этот вопрос могут дать современные статистические тесты (Монте-Карло, рандомизация и кросс-проверка), позволяющие выполнять количественную проверку обоснованности построенных моделей и придать самому понятию "корректность" доказательный смысл.

106. Исторические традиции и характер обрабатываемых данных оказывают сильное влияние на стиль представления метода анализа соответствий. По сути, существует две различных версии интерпретации смысла CA (или его канонической реализации CCA), которые можно условно назвать как "экологическая" и "социологическая".

По первой версии анализ соответствий, разработанный несколькими учеными-экологами (Curtis, McIntosh, 1951; Williams, 1952; Hill, 1973), основан на процедуре *встречного осреднения* (или, в другой терминологии, реципрокного взвешивания RA *reciprocal averaging*). Он был предложен как альтернатива PCA для анализа таблиц численности видов, распределение которых характеризуется сильной асимметрией и длинным правосторонним "хвостом". При этом вместо матрицы ковариаций/корреляций рассчитывается нормированная матрица χ^2-статистик, которая оценивает расхождение между обилиями видов в многомерном пространстве. Общая сумма этих статистик называется *инерцией* [29]. Таким образом, если PCA использует евклидовы расстояния между объектами и линейные связи, то CA предполагает, что любое взаимодействие вида с факторами среды имеет гауссову зависимость [30].

Однако при большом числе видов, характерном для многих экологических сообществ, расстояния между точками в пространстве, образуемом осями соответствия СА, могут быть серьёзно искажены по сравнению с фактическими расстояниям между объектами в исходной таблице данных. Разработаны методы компенсации таких искажений, например, "эффекта арки", но они попутно могут "сглаживать" и реально существующие зависимости.

Исходя из изложенного, при анализе соответствий по "экологической" версии строки исходной таблицы рассматриваются как отдельные изучаемые объекты, а столбцы – как наблюдаемые показатели. Интерпретация биплотов/триплотов CCA также ничем не отличается от RDA. Метод обычно реализуется функцией `cca` из пакета `vegan`.

Одновременно с этими работами и независимо от них, другими учеными, далекими от экологии, анализ соответствий (*correspondence analysis*) рассматривался как поиск "корреспонденций" в таблице сопряженности (Hirschfeld, 1935, Benzécri, 1973). При таком подходе «*… строки и столбцы таблицы рассматриваются симметрично, т. е. нет принципиальных отличий между "объектами" и "переменными". Таким образом, строки и столбцы могут быть переставлены местами без изменения существа анализа*» /М. Эрве/. "Социологи" обычно используют либо ППП «Statistica», либо функции из других пакетов R – см. http://www.gastonsanchez.com/visually-enforced/how-to/2012/07/19/Correspondence-Analysis. При этом трудно сказать, чем "экологическая" и "социологическая" версии отличаются между собой в алгоритмическом плане.

Из дискуссии между со-комментаторами

Н. Ц. (***Натан Цейтлин):*** Я не уверен, что "простой биолог" сможет во всех этих примерах разобраться. Как правило, экспериментаторы не могут грамотно формализовать задачу исследования и эффективно обработать результаты наблюдений. В такой ситуации на помощь экспериментаторам обычно приходит аналитический статистик (**АС**), важнейшую роль которого мы подробно описали в работах [16, 17, 31].

В. Ш. (***Владимир Шитиков):*** Я встречал многих молодых экологов, которым вполне по плечу самостоятельное решение самых сложных задач, и этому способствуют углубленные курсы по планированию исследований в ведущих университетах. Более того, современные биофизики и генетики не только активно используют существующие математические методы, но и становятся решающей движущей силой развития прикладной статистики, разрабатывая новые эффективные модели и алгоритмы (см. например, метод DIABLO [32, 33]).

Н. Ц. Имеется чёткое разделение ролей в теории принятия решений: *лицо, принимающее решение* (**ЛПР**) осуществляет выбор наилучшего варианта действий, а готовит материалы для этого *аналитический статистик* (**АС**), к которому обращается ЛПР за оценками и рекомендациями.

Вы пишете (к п. **15**), что иногда «*большой объем выборки играет против исследователя, и он старается выбрать маломощный критерий*». Получается, что ваш исследователь – какой-то мошенник?!

В. Ш. Из далекого Гамбурга взгляд на имеющиеся у нас реалии может показаться слишком идеалистическим. Пусть, например, "честный АС" работает на химкомбинате, и ему дают задание обосновать нулевую гипотезу H_0, что промышленные сбросы не наносят вред экосистеме близлежащего озера. Но АС выбирает *мощный критерий*, отвергает H_0 в пользу альтернативы H_1 о наличии вредного воздействия (p = 0.045), и его тут же увольняют «за подрыв репутации завода». Тогда он устраивается в

природоохранную инспекцию и, чтобы начислить побольше штрафов, ему дают задание обосновать гипотезу H_1 о том, что сбросы приносят вред экосистеме. Располагая ограниченными данными, АС не находит оснований отклонить H_0 ($p = 0.055$) и его опять увольняют «за неполное служебное соответствие». Тяжело приходится "честной парочке ЛПР-АС" в условиях эффективного менеджмента!

Н. Ц. На будущее предлагаю ознакомиться с предложенным нами методом проверки статистических гипотез с учётом интересов двух конкурирующих сторон [16, раздел 1.1.12].

Теперь перейдем к п. **18**. «*Мощность должна быть разумно высокой*». А что является критерием «*разумности*»? Бог весть! «*Размер эффекта от 0.2 до 0.5 считается малым, а от 0.5 до 0.8 – большим*». Совершенно необоснованные утверждения!

«*При малом* δ *надо хорошо подумать, имеют ли найденные различия биологический смысл*». Все это какие-то абстрактные мудрствования! АС должен грамотно проверять гипотезы согласно принятой формальной теории. Если гипотеза H_0 отклонена, значит нет оснований для беспокойства – успокойся и публикуй результаты. Так наука устроена.

В. Ш. Наука стоит не только на теоремах Неймана-Пирсона и Фишера, но и (в основном) на учете предшествующего опыта, здравом смысле и оценке конкретной ситуации (т. е. том, что Вы называете «экспертным методом принятия решений»). Критический уровень значимости может приниматься в диапазоне от 0.1 до 0.01, мощность – от 0.8 до 0.95, величина эффекта – более 0.6 (там можно ориентироваться на аналогию с распределением *t*-статистики). Все эти заключения носят достаточно случайный характер. Отклонение или принятие H_0 на основе *p*-значения – не приговор суда, исключающий иные точки зрения, и использование иных подходов (например, байесовского) должно всегда приветствоваться.

Н. Ц. Что, опять – "на глазок"?! Надо просто проверять соответствующие гипотезы! В общем, Вы превращаете строгий статистический подход из науки в сплошной волюнтаризм.

В. Ш. М. Ерве на протяжении всего изложения выделяет две основные парадигмы статистического анализа: простые тесты (это то, что проф. А. И. Орлов назвал «убогой эконометрикой») и анализ с использованием моделей. Во втором случае все не так однозначно: критерии оценок могут быть разные, подходов к валидации много и процедуры селекции лучшей модели еще до конца не определены.

Кстати, к п. **18** неверно утверждение «*при использовании критерия Стьюдента обе выборки должны иметь нормальное распределение, а их дисперсии равными*». Достаточно того, что нормальны и гомоскедастичны остатки модели `lm(X~F)`.

Н. Ц. Что значит «*Если модель недостаточно адекватна...*»? До недавнего времени требование адекватности эмпирической функции регрессии (ЭФР) формулировалось косвенно, как требование получить максимальный коэффициент детерминации. Однако непосредственным критерием адекватности ЭФР результатам наблюдений является гипотеза о равенстве остаточной дисперсии отклика в модели и дисперсии воспроизводимости отклика. Нами [16, 31] описана методика вычисления смещённой оценки дисперсии воспроизводимости по данным пассивного регрессионного эксперимента. Когда модель *недостаточно адекватна* (т. е. не очень хорошо соответствует анализируемым данным), то достигнутая оценка уровня значимости *p* меньше критического значения α, и гипотеза об адекватности отклоняется. В таком случае структура ЭФР пересматривается и расчёты повторяются.

В. Ш. М. Ерве рассматривает в этом случае предположения о нормальном законе распределения случайной ошибки ε и ее гомоскедастичности. Наиболее надежный путь проверки этих предположений – анализ графиков, но там возможны субъективные

толкования "близко", "достаточно далеко" и проч. Ваш подход весьма интересен – ему, видимо, близок по основным идеям тест на потерю адекватности (*lack-of-fit tests* – Ritz, Martinussen, 2011). Надеемся рассмотреть его на страницах нашего блога https://stok1946.blogspot.com.

Н. Ц. Что за интервальные оценки рассматриваются в п. **43**?

В. Ш. Если использовать `predict(модель.lm, newdata=df, interval="prediction", level=.95)`, то получаем прогнозные интервалы для каждой заданной точки `x`, а если использовать `interval="confidence"`, то получаем доверительные интервалы отклика – см. [5], с. 108. Для того, чтобы получить толерантные интервалы, можно использовать пакет `tolerance`.

Н. Ц. Совсем не обязательно удалять отдельные предикторы из высоко скоррелированных подмножеств, как предлагается в п. **44**. См. другие методы преодоления эффекта мультиколлинеарности [16, 31] и примеры – там же.

В. Ш. В нашем примере попытка регуляризации регрессии привела как раз к одновременному выбору самых высоко коррелированных предикторов `oPO4` и `PO4`. Процедура ридж-регрессии "не борется" с мультиколлинеарностью, а просто обеспечивает устойчивую оценку коэффициентов в условиях плохо обусловленных матриц.

Н. Ц. Алгоритмы кросс-проверки – давно и широко распространяемая нелепость, с которой я категорически не согласен: в модели должны участвовать все данные! А критерий её адекватности только один – *равенство дисперсий* (остаточной и воспроизводимости).

В. Ш. В статистике категоричность, как нигде, вредна. Можно набрать с десяток процедур валидации моделей, которые, если не конкурируют, но дают "слабо коррелирующие" между собой результаты. Я в своей практике эти процедуры и критерии селекции наиболее полезных моделей обычно выстраиваю по следующему приоритету:

- ошибка при кросс-проверке, либо на ином внешнем дополнении (неустойчивая модель, у которой эта ошибка более чем в 1.5 раза превышает "табельное" стандартное отклонение остатков, вряд ли пригодится на практике);
- стандартное отклонение для остатков в сочетании с числом используемых параметров β (т. е. информационные критерии типа AIC);
- число статистически незначимых предикторов;
- отклонение от нормального закона распределения остатков или гомоскедастичности (например, по тесту Бройша-Пагана);
- коэффициенты детерминации;
- статистическая значимость модели в целом по критерию Фишера.

Не всегда соблюдается именно такой порядок, но общая стратегия обычно прагматична: лучшая модель – это та, которая лаконична, устойчива и меньше всего ошибается на широком множестве незнакомых данных. Все остальные критерии и соображения играют "роли второго плана".

Н. Ц. Незначимые коэффициенты, на самом деле, играют незначимую роль.

В. Ш. Предиктор становится статистически незначимым только на некотором шаге алгоритма построения модели, но может оставаться весьма полезным при иных обстоятельствах. Например, "тип реки" статистически незначим для линейной модели, но становится значимым для модели лассо. А для модели отрицательного биномиального распределения включение этого фактора стабилизирует прогнозы на внешнем дополнении. Хотя, разумеется, число незначимых предикторов должно быть по возможности минимальным, но это правило – только третье в моем списке.

Н. Ц. *А з*ачем вообще огород городить с выбором вида распределения, если уже используются робастные методы оценивания, пригодные для любого унимодального распределения?!

В. Ш. Слишком оптимистичное утверждение относительно моделей гребневой регрессии. А по сути, для "строительства огорода" у нас два аргумента: (а) биомасса водорослей зависит от их числа, т. е. природа отклика может интерпретироваться как счетная, и (б) модель отрицательного биномиального распределения оказалась существенно лучше остальных.

Н. Ц. Это опять – "на глазок". А надо проверять гипотезу о том, что среднеквадратичное отклонение (СО) остаточной ошибки регрессии не уменьшилась против альтернативы, что уменьшилась!

В. Ш. Да ничего не надо проверять!!! Строится просто пул возможных моделей, ранжированный по заданному критерию, а дальше – дело исследователя: (а) использовать наилучшую модель, (б) использовать любую другую модель из окрестности оптимума, исходя из предметных соображений или (в) построить "коллектив" и выполнять мультимодельный прогноз (см. подробности в https://stok1946.blogspot.com/2018/08/blog-post_19.html)

Н. Ц. У Вас получается, что статистика – не наука, а сплошной волюнтаризм.

В. Ш. Не исключено, что так оно отчасти и есть.

Н. Ц. Ещё в школе учат, что всякая математическая функция должна быть определена на какой-то области аргументов. А ни в оригинальной работе М. Эрве, ни в комментариях почему-то не говорится о методе окаймления области действия ЭФР. Этот недостаток можно восполнить, познакомившись с разделом 4.6. в книге [31].

В. Ш. Интересно, как этот алгоритм будет работать в условиях мультиколлинеарности комплекса из пары десятков переменных.

Н. Ц. Многое в нашей жизни еще не потеряло интерес…

Литературные ссылки

[1] *Зарядов И.С.* Введение в статистический пакет R: типы переменных, структуры данных, чтение и запись информации, графика. Москва: Изд-во РУДНБ, 2010. 207 с.

[2] *Зарядов И.С.* Статистический пакет R: теория вероятностей и математическая статистика. Москва: Изд-во РУДНБ, 2010. 141 с.

[3] *Шипунов А.Б.* и др. Наглядная статистика. Используем R! М.: ДМК Пресс, 2014. 298 с.

[4] *Кабаков Р.И.* R в действии. Анализ и визуализация данных в программе R / пер. с англ. П.А. Волковой. М.: ДМК Пресс, 2014. 588 с.

[5] *Шитиков В.К., Розенберг Г. С.* Рандомизация и бутстреп: статистический анализ в биологии и экологии с использованием R. Тольятти: Кассандра, 2014. 314 с.

[6] *Мастицкий С.Э., Шитиков В.К.* Статистический анализ и визуализация данных с помощью R. М.: ДМК Пресс, 2015. 496 с.

[7] *Шитиков В.К.* Экотоксикология и статистическое моделирование эффекта с использованием R. Электронная книга. 2016.

[8] *Шитиков В.К., Мастицкий С.Э.* Классификация, регрессия и другие алгоритмы Data Mining с использованием R. Электронная книга. 2017.

[9] *Маркова Е.Е., Лисенков А.Н.* Комбинаторные планы в задачах многофакторного эксперимента. М.: Наука, 1979. 349 с.

[10] Проблемы экологического эксперимента: (планирование и анализ наблюдений) / Под ред. чл.корр. РАН Г.С. Розенберга и д.б.н. Д.Б. Гелашвили; сост. и коммент. д.б.н. В.К. Шитикова. Тольятти СамНЦ РАН; Кассандра, 2008. 274 с. http://www.ievbras.ru/ecostat/Kiril/

[11] *Ермаков С.М., Жиглявский А.А.* Математическая теория оптимального эксперимента. М.:Наука, 1987. 392 с.

[12] *Хромов-Борисов Н.Н.* Синдром статистической снисходительности или значение и назначение *p*-значения // Телеконференция по медицине, биологии и экологии, 2011. №4. URL: http://tele-conf.ru .

[13] Международный комитет редакторов медицинских журналов: Единые требования к рукописям, представляемым в биомедицинские журналы: правила написания и редактирования материалов // Межд. журн. мед. практики, 2005. № 5. С. 10–23. http://www.mediasphera.ru/mjmp/2005/5/10.pdf.

[14] *Зорин Н.А.* О неправильном употреблении термина "достоверность" в российских научных психиатрических и общемедицинских статьях. 2000. http://www.biometrica.tomsk.ru/let1.htm

[15] *Goodman S.* A dirty dozen: Twelve *P*-value misconceptions // Semin. Hematol., 2008. Vol. 45. P. 135-140.

[16] *Цейтлин Н. А.* Из опыта аналитического статистика. М.: Солар, 2007. 906 с. www.cubematrix.com/oldsite/anlagen/as.pdf

[17] *Цейтлин Н. А.* Проблемы проверки статистических гипотез. Tseitlin N. Gorbach A. Der Probleme der statistischen Hypothesentests. Hamburg:CuBe Matrix Inh., 2015. 32 с. http://biometrica.tomsk.ru/A-metod-HPI-2.pdf

[18] *Рубанович А.В.* Статистика множественных сравнений в ассоциативных исследованиях полиморфизма ДНК: Кошмар Бонферрони. Институт общей генетики им. Н.И. Вавилова РАН. Презентация. www.biometrica.tomsk.ru/bonferroni.ppt

[19] *Wickham H.* `ggplot2`*:* Elegant graphics for data analysis. Springer, 2009. 260 p.

[20] *Мастицкий С.Э.* Визуализация данных с помощью ggplot2. М.: ДМК Пресс, 2016. 222 с.

[21] *Torgo L.* Data mining with R: learning with case studies. Chapman & Hall/CRC, 2011. 272 p.

[22] *Kenkel N.C.* On selecting an appropriate multivariate analysis // Canadian Journal of Plant Science. 2006. V. 86. P. 663–676.

[23] *Prentice I.C.* Non-metric ordination methods in ecology // Journal of Ecology. 1977. V. 65. P. 85–94.

[24] *Minchin P.R.* An evaluation of the relative robustness of techniques for ecological ordination // Vegetatio. 1987. V. 67. P. 1167-1179.

[25] *Legendre P., Anderson M.J.* Distance-based redundancy analysis: testing multispecies responses in multifactorial ecological experiments // Ecological Monographs. 1999. V. 69. P. 1–24.

[26] *Шитиков В.К., Зинченко Т.Д., Розенберг Г.С.* Макроэкология речных сообществ: концепции, методы, модели. Тольятти: Кассандра, 2012. 257 с. http://www.ievbras.ru/ecostat/Kiril/

[27] *Терехова В.А., Шитиков В.К., Иванова А.Е., Кыдралиева К.А.* Оценка экологического риска техногенного загрязнения почвы на основе статистического распределения встречаемости видов микромицетов // Экология. 2017. №5. С. 339–346

[28] *Wood S.N.* Generalized Additive Models: An Introduction with R. Chapman, Hall/CRC, 2006. 410 p.

[29] *Legendre P., Legendre L.* Numerical Ecology. Amsterdam: Elsevier Sci. BV, 2012. 1006 p.

[30] *Джонгман Р.Г.Г., тер Браак С.Дж.Ф., ван Тонгерен О.Ф.Р.* Анализ данных в экологии сообществ и ландшафтов. М.: РАСХН. 1999. 306 с.

[31] *Горбач А.Н., Цейтлин Н.А.* Покупательское поведение: анализ спонтанных последовательностей и регрессионных моделей в маркетинговых исследованиях. - Киев: Освіта України, 2011. 298 с. http://www.cubematrix.com/oldsite/anlagen/asp.pdf

[32] *Singh A., Gautier B., Shannon C.P., Vacher M., Rohart F., Tebutt S.J., Le Cao K-A.* DIABLO – an integrative, multi-omics, multivariate method for multi-group classification // BioRxiv. 2016. No. 067611. 50 p.

[33] *Hervé M.R., Nicolè F., Lê Cao K.A.* Multivariate Analysis of Multiple Datasets: a Practical Guide for Chemical Ecology // Journal of Chemical Ecology. 2018. V. 44. P. 215–234.