Basic Steps for ANOVA

Zircon

2022-11-19

目录

```
rm(list=ls())
```

1 加载与方差分析相关的包

```
# 前四个与方差分析最紧密相关
library(rstatix) # identify_outliers, and include good stats test functions
```

```
##
## Attaching package: 'rstatix'
```

```
## The following object is masked from 'package:stats':
##
##     filter
```

```
library(dplyr) # %>%
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(bruceR) # a good ANOVA related wrapper
```

```
##
## bruceR (version 0.8.9)
## BRoadly Useful Convenient and Efficient R functions
```

```
##
## Packages also loaded:
## √ dplyr      √ emmeans       √ ggplot2
## √ tidyr      √ effectsize    √ ggtext
## √ stringr    √ performance   √ cowplot
## √ forcats    √ lmerTest      √ see
## √ data.table
##
## Main functions of `bruceR`:
## cc()             Describe()  TTEST()
## add()            Freq()      MANOVA()
## .mean()          Corr()      EMMEANS()
## set.wd()         Alpha()     PROCESS()
## import()         EFA()       model_summary()
## print_table()    CFA()       lavaan_summary()
##
## https://psychbruce.github.io/bruceR/
##
## These R packages are dependencies of `bruceR` but not installed:
## pacman, lmtest, vars, phia, BayesFactor, GGally, GPArotation
## ***** Please Install All Dependencies *****
## install.packages("bruceR", dep=TRUE)
```

```
library(car) # leveneTest
```

```
## Loading required package: carData
```

```
##
## Attaching package: 'car'
```

```
## The following object is masked from 'package:dplyr':
##
##     recode
```

```
library(afex) # for factorial design anova
```

```
## ************
## Welcome to afex. For support visit: http://afex.singmann.science/

## - Functions for ANOVAs: aov_car(), aov_ez(), and aov_4()
## - Methods for calculating p-values with mixed(): 'S', 'KR', 'LRT', and 'PB'
## - 'afex_aov' and 'mixed' objects can be passed to emmeans() for follow-up tests
## - NEWS: emmeans() for ANOVA models now uses model = 'multivariate' as default.
## - Get and set global package options with: afex_options()
## - Set orthogonal sum-to-zero contrasts globally: set_sum_contrasts()
## - For example analyses see: browseVignettes("afex")
## ************

##
## Attaching package: 'afex'

## The following object is masked from 'package:lme4':
##
##     lmer
```

```
library(ggpubr)  # plot in publication-ready formats
```

```
##
## Attaching package: 'ggpubr'

## The following object is masked from 'package:cowplot':
##
##     get_legend
```

```
library(purrr)  # for functions dealing with functions
```

```
##
## Attaching package: 'purrr'

## The following object is masked from 'package:car':
##
##     some

## The following object is masked from 'package:data.table':
```

```
##
##     transpose
```

2 数据初步整理

数据通常有两种存储的形式，一种是长数据，一种是短数据，长数据比较普遍。长数据和短数据的对比就是，长数据还需要根据分组变量将所有的数据分组。

2.1 载入数据并且整理为数据框

anova 处理的对象是 Data Frame，因此数据都要使用 `data.frame` 函数转化为数据框。

```
# 这一组数据用于 one-way ANOVA，并且可以用于 repeated-measured ANOVA，strategy 表示自变量
score = c(3,3,4,6,6,8,5,3,5,7,8,8,8,5,8,9,8,10,8,9,7,10,10,10)
subj = rep(1:6,times=4) # 善于使用 rep 函数，创建标签
strategy=c(rep(1,6),rep(2,6),rep(3,6),rep(4,6))
long_data = data.frame(score,subj,strategy) # 构建 dataframe
long_data
```

```
##    score subj strategy
## 1      3    1        1
## 2      3    2        1
## 3      4    3        1
## 4      6    4        1
## 5      6    5        1
## 6      8    6        1
## 7      5    1        2
## 8      3    2        2
## 9      5    3        2
## 10     7    4        2
## 11     8    5        2
```

```
## 12     8    6         2
## 13     8    1         3
## 14     5    2         3
## 15     8    3         3
## 16     9    4         3
## 17     8    5         3
## 18    10    6         3
## 19     8    1         4
## 20     9    2         4
## 21     7    3         4
## 22    10    4         4
## 23    10    5         4
## 24    10    6         4
```

```
# 这一组数据用于 two-way ANOVA，设计的是 2*2 实验，研究 time 和 presentation 对 performanc
perf=c(11,8,9,10,7,4,4,8,5,4,10,10,7,6,7,10,6,10,10,9)
time=c(rep(1,10),rep(0,10))
pre=c(rep(1,5),rep(0,5),rep(1,5),rep(0,5))
id=1:20
two=data.frame(id,time,pre,perf)
two
```

```
##    id time pre perf
## 1   1    1   1   11
## 2   2    1   1    8
## 3   3    1   1    9
## 4   4    1   1   10
## 5   5    1   1    7
## 6   6    1   0    4
## 7   7    1   0    4
## 8   8    1   0    8
## 9   9    1   0    5
## 10 10    1   0    4
## 11 11    0   1   10
```

```
## 12 12    0   1   10
## 13 13    0   1    7
## 14 14    0   1    6
## 15 15    0   1    7
## 16 16    0   0   10
## 17 17    0   0    6
## 18 18    0   0   10
## 19 19    0   0   10
## 20 20    0   0    9
```

2.2 检验数据本身是否正常

拿到数据以后，应该首先检查是否具有缺失值，是否有极端值（outlier）。

2.3 检查缺失值

使用 `is.na` 函数，可以配合 `sum` 看全局。

```
sum(is.na(long_data))
```

```
## [1] 0
```

```
# 可以直接对所有数据检验是否有缺失值，如果没有直接跳过后续步骤

# 具体对每一个列检验缺失值
sum(is.na(long_data$score))
```

```
## [1] 0
```

```
sum(is.na(long_data$subj))
```

```
## [1] 0
```

```
sum(is.na(long_data$strategy))
```

```
## [1] 0
```

```
num_na=long_data %>% map(is.na)
# 也可以使用 map 函数直接对三个列进行检验，但结果比较不直观
summary(num_na)
```

```
##          Length Class  Mode
## score    24     -none- logical
## subj     24     -none- logical
## strategy 24     -none- logical
```

2.4 检查极端值

`outlier`: 值在 $Q_3 + 1.5IQR$ 和 $Q_1 - 1.5IQR$ 区间之外 `extreme`: 值在 $Q_3 + 3IQR$ 和 $Q_1 - 3IQR$ 区间之外

```
# 检查极端值
long_data %>% # 管道相当于把管道之前的结果传入之后函数的第一个参数
  group_by(strategy) %>% # 注意 group_by 的列并不需要实现转化为 factor 类型
  identify_outliers(score) # 极端值必定是在每组之内讨论的，因此必须先进行分组
```

```
## # A tibble: 2 x 5
##   strategy score  subj is.outlier is.extreme
##      <dbl> <dbl> <int> <lgl>      <lgl>
## 1        3     5     2 TRUE       TRUE
## 2        3    10     6 TRUE       FALSE
```

2.5 分组变量转化为因子

数据框中包含的分组变量应该转化为 factor 类型，并且让数据依据其分组。

虽然不是一定要用到，但会比较保险，比如 `leveneTest` 就需要依照分组变量计算，绘制盒须图等也需要合适的分组变量，更何况，分组变量本身的性质就应该是 factor。

分组变量转为因子类型的方法有：* `convert_as_factor`：管道友好，可以

传入多个参数 * as.factor：简单直白，注意必须是对数据框中的列操作，而不是对原始列的变量操作

```
# 将分组变量转化为因子
# 本例中，subj 和 strategy 都应该转化为因子

# 方法一：使用 convert_as_factor
long_data=data.frame(long_data) %>%
  convert_as_factor(strategy)
# 注意，convert_as_factor 函数是需要接收返回值的，它并不是一种方法，否则数据没有被处理
long_data # 发现 strategy 转变为了因子
```

```
##    score subj strategy
## 1      3    1        1
## 2      3    2        1
## 3      4    3        1
## 4      6    4        1
## 5      6    5        1
## 6      8    6        1
## 7      5    1        2
## 8      3    2        2
## 9      5    3        2
## 10     7    4        2
## 11     8    5        2
## 12     8    6        2
## 13     8    1        3
## 14     5    2        3
## 15     8    3        3
## 16     9    4        3
## 17     8    5        3
## 18    10    6        3
## 19     8    1        4
## 20     9    2        4
## 21     7    3        4
```

```
## 22    10    4         4
## 23    10    5         4
## 24    10    6         4
```

```
# convert_as_factor 可以传入多个参数，可以让所有的要因子化的列都转变为因子类型!

# 方法二: as.factor
# 错误示范，对原始列的变量直接操作
subj=as.factor(subj)
# 改变的只是当时构成 long_data 中的那一列变量，但是 long_data 中的 subj 实际上并没有更新!
long_data # 发现并没有任何改变发生
```

```
##    score subj strategy
## 1      3    1         1
## 2      3    2         1
## 3      4    3         1
## 4      6    4         1
## 5      6    5         1
## 6      8    6         1
## 7      5    1         2
## 8      3    2         2
## 9      5    3         2
## 10     7    4         2
## 11     8    5         2
## 12     8    6         2
## 13     8    1         3
## 14     5    2         3
## 15     8    3         3
## 16     9    4         3
## 17     8    5         3
## 18    10    6         3
## 19     8    1         4
## 20     9    2         4
## 21     7    3         4
```

```
## 22    10    4        4
## 23    10    5        4
## 24    10    6        4
```

```
# 正确示范：修改数据框的列
long_data$subj=as.factor(long_data$subj) # 注意必须是对数据框中的列操作
long_data # 发现 subj 转变为了因子
```

```
##    score subj strategy
## 1      3    1        1
## 2      3    2        1
## 3      4    3        1
## 4      6    4        1
## 5      6    5        1
## 6      8    6        1
## 7      5    1        2
## 8      3    2        2
## 9      5    3        2
## 10     7    4        2
## 11     8    5        2
## 12     8    6        2
## 13     8    1        3
## 14     5    2        3
## 15     8    3        3
## 16     9    4        3
## 17     8    5        3
## 18    10    6        3
## 19     8    1        4
## 20     9    2        4
## 21     7    3        4
## 22    10    4        4
## 23    10    5        4
## 24    10    6        4
```

综合以上介绍，convert_as_factor 是最通用、最便捷的！

```
long_data=data.frame(long_data) %>%
  convert_as_factor(strategy,subj)
# 注意，convert_as_factor 函数是需要接收返回值的，它并不是一种方法，否则数据没有被处理。
long_data # 发现 strategy 和 subj 都转变为了因子
```

```
##    score subj strategy
## 1      3    1        1
## 2      3    2        1
## 3      4    3        1
## 4      6    4        1
## 5      6    5        1
## 6      8    6        1
## 7      5    1        2
## 8      3    2        2
## 9      5    3        2
## 10     7    4        2
## 11     8    5        2
## 12     8    6        2
## 13     8    1        3
## 14     5    2        3
## 15     8    3        3
## 16     9    4        3
## 17     8    5        3
## 18    10    6        3
## 19     8    1        4
## 20     9    2        4
## 21     7    3        4
## 22    10    4        4
## 23    10    5        4
## 24    10    6        4
```

```
two=two %>% convert_as_factor(pre,time)
```

3 检查 ANOVA 的前提

数据基本无误或排除特殊情况后，根据 anova 的类型，检验需要的前提。比如正态性假设，方差同质假设、球形假设等。

3.1 正态性假设的检验

3.1.1 管道配合分组

注意这里的正态性检验的函数是 `shapiro_test`，而不是 `shapiro.test`

传入数据，使用 `group_by` 分组（可以同时按多个分组变量实现分组），进行正态性检验 `shapiro_test`

```
long_data %>%
  group_by(strategy) %>%
  shapiro_test(score)
```

```
## # A tibble: 4 x 4
##   strategy variable statistic     p
##   <fct>    <chr>        <dbl> <dbl>
## 1 1        score        0.896 0.352
## 2 2        score        0.896 0.352
## 3 3        score        0.873 0.238
## 4 4        score        0.831 0.110
```

3.1.2 tapply

```
tapply(X, INDEX, FUN = NULL)
```

- `X`: 函数的应用对象，在数据分析时，一般是长数据形式
- `INDEX`: 传入分组变量，有多个分组变量时，用列表承载。每个分组变量都应该与 `X` 有相同的长度
- `FUN`: 要应用的函数

tapply 的结果会以列表的形式存储，存储着分组后每一个 cell 中的处理结果，之后用 for 循环遍历即可。

```
normfit=tapply(perf,list(time,pre),shapiro.test)
for (i in 1:2){
  for (j in 1:2){
    print(normfit[i,j])
  }
}
```

```
## [[1]]
##
##  Shapiro-Wilk normality test
##
## data:  X[[i]]
## W = 0.70079, p-value = 0.009761
##
##
## [[1]]
##
##  Shapiro-Wilk normality test
##
## data:  X[[i]]
## W = 0.8173, p-value = 0.1113
##
##
## [[1]]
##
##  Shapiro-Wilk normality test
##
## data:  X[[i]]
## W = 0.70079, p-value = 0.009761
##
##
```

```
## [[1]]
##
##  Shapiro-Wilk normality test
##
## data:  X[[i]]
## W = 0.98676, p-value = 0.9672
```

3.2 方差同质假设的检验

- leveneTest: 输入公式、输入数据（默认 center=median，可以设置为 center=mean）
- levene_test: 管道友好，输入公式（默认 center=median，可以设置为 center=mean）

center=median 能提供更有鲁棒性的结果，不过一般还是设置为 center=mean.

注意，**分组变量需要调整为因子类型**，这一点建议在**数据的基本处理阶段**就要完成!

```
# leveneTest
leveneTest(score~strategy,data=long_data)
```

```
## Levene's Test for Homogeneity of Variance (center = median)
##       Df F value Pr(>F)
## group  3  0.9786 0.4226
##       20
```

```
# levene_test (%>%)
long_data %>%
  convert_as_factor(strategy) %>%
  # 如果前期没有将 strategy 转化为因子类型，此处需要加上这条语句
  levene_test(score~strategy)
```

```
## # A tibble: 1 x 4
##     df1   df2 statistic     p
```

```
##   <int> <int>     <dbl> <dbl>
## 1     3    20     0.979 0.423
```

```
# 也可以对比发现两种方法最终得到了一样的结果
```

3.3 协方差同质假设（球形假设）的检验

mauchly.test：第一个参数传入线性回归模型，第二个参数传入公式

```
matrix_data = matrix(long_data$score,nrow=6,ncol=4)
# 首先需要将数据转化为矩阵，其中行表示每个被试（subject），列表示不同的条件（treatment）
mlmfit = lm(matrix_data ~ 1)
# 变量 1~ 变量 2，代表变量 1 根据变量 2 分组，如变量 2 是常数 1，会返回一个拟合的线性模型
mauchly.test(mlmfit,X=~1)
```

```
##
##  Mauchly's test of sphericity
##  Contrasts orthogonal to
##  ~1
##
##
## data:  SSD matrix from lm(formula = matrix_data ~ 1)
## W = 0.55102, p-value = 0.8239
```

```
# 第二个参数指定为 X=~1，代表进行协方差同质性检
# 输出的 p-value 越大越好，认为球形假设成立
```

4 ANOVA

- aov,anova 等 R 语言原生函数
- MANOVA：bruceR 包中的强大函数

4.1 R 语言原生函数

4.1.1 aov

```
aov(formula, data = NULL)
```

aov 函数可以直接对两列数据进行方差分析，且注意第二列数据应该为因子类型（分组变量）。如果使用数据框，那么 formula 中的变量即为数据框的列名称。

aov 函数返回的是组内和组间的和方、自由度，也即最基本的方差分析的结果。这一结果通常用变量承接，可以认为是一种模型，可以用于后续的分析，比如传入 summary,anova 函数中去。

Residuals 指的是组内。

```
fit_model = aov(score ~ strategy,long_data)
# 前一个为因变量，后一个为分组变量
fit_model
```

```
## Call:
##    aov(formula = score ~ strategy, data = long_data)
##
## Terms:
##                  strategy Residuals
## Sum of Squares         60        62
## Deg. of Freedom         3        20
##
## Residual standard error: 1.760682
## Estimated effects may be unbalanced
```

```
# 要注意，aov 的用法是要分析的数据处于一列，分组数据处于另一列
# 因此，如果有两列数据想要用 aov，需要将两列数据合并，并单独添加一列变量对数据分组

# 例如有 3 组变量，想要通过 aov 拟合：
data1 = 1 : 11
```

```r
data2 = 5 : 15
data3 = c(10 : 19,21)

# 需要将三个变量合并到一列，并添加分组变量
# 以下代码是最 Raw R 语言的写法

data_y = c(data1, data2, data3)
factor_y = c(
  rep(1, length(data1)),
  rep(2, length(data2)),
  rep(3, length(data3))
)
factor_y = as.factor(factor_y)

aov(data_y ~ factor_y) # 可以直接对现成的两列数据跑 ANOVA
```

```
## Call:
##    aov(formula = data_y ~ factor_y)
##
## Terms:
##                   factor_y Residuals
## Sum of Squares  456.7273  340.9091
## Deg. of Freedom        2        30
##
## Residual standard error: 3.370999
## Estimated effects may be unbalanced
```

```r
# 更规范且常用的方法是将数据整理为数据框
long_test_data=data.frame(group=factor_y,num=data_y)
anova(aov(num~group,long_test_data))
```

```
## Analysis of Variance Table
##
## Response: num
```

```
##           Df Sum Sq Mean Sq F value    Pr(>F)
## group      2 456.73 228.364  20.096 2.901e-06 ***
## Residuals 30 340.91  11.364
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# 宽数据可以直接用 MANOVA 处理，迅速将三列 data 合并
fit_data=MANOVA(data.frame(data1,data2,data3),
                dvs='data1:data3',
                dvs.pattern = 'data(.)',
                within  = 'data')
```

```
##
## Note:
## dvs="data1:data3" is matched to variables:
## data1, data2, data3

##
## ====== ANOVA (Within-Subjects Design) ======
##
## Descriptives:
##
##  "data"   Mean    S.D.  n
##
##   data1  6.000 (3.317) 11
##   data2 10.000 (3.317) 11
##   data3 15.091 (3.477) 11
##
## Total sample size: N = 11
##
## ANOVA Table:
## Dependent variable(s):      data1, data2, data3
## Between-subjects factor(s): -
## Within-subjects factor(s):  data
## Covariate(s):               -
```

```
##
##            MS   MSE df1 df2        F     p      ²p [90% CI of  ²p]   ²G
##
## data  228.364 0.030   2  20 7536.000 <.001 ***   .999 [.998, .999] .573
##
## MSE = mean square error (the residual variance of the linear model)
##  ²p = partial eta-squared = SS / (SS + SSE) = F * df1 / (F * df1 + df2)
##  ²p = partial omega-squared = (F - 1) * df1 / (F * df1 + df2 + 1)
##  ²G = generalized eta-squared (see Olejnik & Algina, 2003)
## Cohen' s f² =  ²p / (1 -  ²p)
##
## Levene' s Test for Homogeneity of Variance:
## No between-subjects factors. No need to do the Levene' s test.
##
## Mauchly' s Test of Sphericity:
## The repeated measures have only two levels. The assumption of sphericity is always m
```

```
# 注意的是，宽数据只能进行被试内检验，也即重复测量方差分析。
```

值得一提的是，方差分析是一般线性模型的特例。因此，可以用 aov 模型拟合的数据，理论上都可以用线性模型 (Linear Model) 来拟合。

下方的方差分析结果与上面用 aov 的结果是一致的。

```
lm_model = lm(num~group,long_test_data)
anova(lm_model)
```

```
## Analysis of Variance Table
##
## Response: num
##           Df Sum Sq Mean Sq F value    Pr(>F)
## group      2 456.73 228.364  20.096 2.901e-06 ***
## Residuals 30 340.91  11.364
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

4.1.2 anova

`anova(object, ...)`

`object`:an object containing the results returned by a model fitting function (e.g., lm or glm).`anova(object, ...)` 中可以包含一个或者多个这样的对象。"When given a single argument it produces a table which tests whether the model terms are significant. When given a sequence of objects, anova tests the models against one another in the order specified."

`anova(object, ...)` 计算的是方差分析表，在原先 `aov` 得到的模型的基础上计算均方、F 值、p 值。

```
anova(fit_model)
```

```
## Analysis of Variance Table
##
## Response: score
##           Df Sum Sq Mean Sq F value   Pr(>F)
## strategy   3     60    20.0  6.4516 0.003116 **
## Residuals 20     62     3.1
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

4.1.3 summary

`summary` 也能返回一样的方差分析表。

"The function summary.lm computes and returns a list of summary statistics of the fitted linear model given in object"

```
summary(fit_model)
```

```
##             Df Sum Sq Mean Sq F value  Pr(>F)
## strategy     3     60    20.0   6.452 0.00312 **
## Residuals   20     62     3.1
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

4.2 bruceR 包中的函数

4.2.1 MANOVA (multi-factor ANOVA)

MANOVA 只需要提供数据，确定自变量，申明组内和组间的关系三步即可。返回的结果包括方差分析，效应量（gerenral & partial, 90%CI）。

```
MANOVA(data,subID = NULL,dv = NULL,dvs = NULL,dvs.pattern =
NULL,between = NULL,within = NULL,covariate = NULL,ss.type
= "III",sph.correction = "none",aov.include = FALSE,digits =
3,nsmall = digits,file = NULL)
```

- `data`：可以传入宽数据或长数据
- `subID`：被试的 ID，长数据需要传入，这在长数据中也即列的名称；对于宽数据不需要传入这一参数
- `dv`：自变量
  - 长数据：`dv` 即为因变量
  - 宽数据：`dv` 只有在被试间设计可用，对于被试内或者混合设计，要用 `dvs` 或 `dvs.pattern`
- `dvs`：重复测量方差分析。只适用于宽数据（被试内或混合设计）。传入方式有两种
  - `start:stop`：声明变量的范围
  - charactor vector：向量中包含变量的名称
- `dvs.pattern`：如果使用了 `dvs.`，那么要传入正则表达式明确变量名的样式
- `between`：between-subjects factor(s)，如有多个分组变量，将它们包含在向量中
- `within`：within-subjects factor(s)，如有多个分组变量，将它们包含在向量中
- `sph.correction`：违背球形假设的纠正。只适用于多余三个水平的重复测量方差分析。通常设置为 `GG`

其中，`between` 和 `within` 需要至少明确一个量！如果需要进行重复测量

（消除被试个体差异），就用 `within` 指定；否则用 `between` 指定。长数据和宽数据的比较：可以认为长数据包含了组间的信息（自变量），也包含了组内的信息（被试个体）。长数据的优点是简单，可以承载任意维度的信息，但宽数据承载的信息有限，可以把向 `MANOVA` 传参的过程理解为将长数据转变为宽数据的过程。长数据需要明确 `subID`，相当于依据其将数据重新整合，把被试 ID 作为首列；长数据还要明确 `within` 变量，程序再根据 `within` 变量将数据拆分成多列。这样之后长数据变得和宽数据”等价”了。此外，系统需要明白每一行或每一列数据的意义并转化，因此**每一列都必须被指明含义**。

长数据

```
# one-way
repeat_fit <- MANOVA(long_data,
                     subID = "subj", # 明确这一列指代的是被试编号
                     dv = "score", # 因变量
                     within = "strategy",
                     # strategy 作为 within-subject 的变量，说明进行重复测量方差检验
                     sph.correction = "GG")  # greenhouse-geisser correction
```

```
##
##     * Data are aggregated to mean (across items/trials)
##     if there are >=2 observations per subject and cell.
##     You may use Linear Mixed Model to analyze the data,
##     e.g., with subjects and items as level-2 clusters.

##
## ====== ANOVA (Within-Subjects Design) ======
##
## Descriptives:
##
##  "strategy"  Mean    S.D. n
##
##   strategy1 5.000 (2.000) 6
##   strategy2 6.000 (2.000) 6
```

```
##   strategy3 8.000 (1.673) 6
##   strategy4 9.000 (1.265) 6
##
## Total sample size: N = 6
##
## ANOVA Table:
## Dependent variable(s):      score
## Between-subjects factor(s):  -
## Within-subjects factor(s):  strategy
## Covariate(s):                -
##
##               MS   MSE   df1    df2      F     p      ²p [90% CI of  ²p]   ²G
##
## strategy  27.551 1.286 2.178 10.889 21.429 <.001 ***   .811 [.558, .894] .492
##
## Sphericity correction method: GG (Greenhouse-Geisser)
## MSE = mean square error (the residual variance of the linear model)
##  ²p = partial eta-squared = SS / (SS + SSE) = F * df1 / (F * df1 + df2)
##  ²p = partial omega-squared = (F - 1) * df1 / (F * df1 + df2 + 1)
##  ²G = generalized eta-squared (see Olejnik & Algina, 2003)
## Cohen' s f² =  ²p / (1 -  ²p)
##
## Levene' s Test for Homogeneity of Variance:
## No between-subjects factors. No need to do the Levene' s test.
##
## Mauchly' s Test of Sphericity:
##
##           Mauchly's W     p
##
## strategy       0.5510  .824
##
```

```
# two-way
two_fit=MANOVA(two,dv='perf',between = c('time','pre'))
```

```
##
## ====== ANOVA (Between-Subjects Design) ======
##
## Descriptives:
##
##  "time" "pre"  Mean    S.D. n
##
##   time0  pre0 9.000 (1.732) 5
##   time0  pre1 8.000 (1.871) 5
##   time1  pre0 5.000 (1.732) 5
##   time1  pre1 9.000 (1.581) 5
##
## Total sample size: N = 20
##
## ANOVA Table:
## Dependent variable(s):      perf
## Between-subjects factor(s): time, pre
## Within-subjects factor(s):  -
## Covariate(s):               -
##
##                 MS   MSE df1 df2      F     p      ²p [90% CI of  ²p]   ²G
##
## time        11.250 3.000   1  16  3.750  .071 .     .190 [.000, .454] .190
## pre         11.250 3.000   1  16  3.750  .071 .     .190 [.000, .454] .190
## time * pre  31.250 3.000   1  16 10.417  .005 **    .394 [.095, .617] .394
##
## MSE = mean square error (the residual variance of the linear model)
##  ²p = partial eta-squared = SS / (SS + SSE) = F * df1 / (F * df1 + df2)
##  ²p = partial omega-squared = (F - 1) * df1 / (F * df1 + df2 + 1)
##  ²G = generalized eta-squared (see Olejnik & Algina, 2003)
## Cohen' s f² =  ²p / (1 -  ²p)
##
## Levene' s Test for Homogeneity of Variance:
```

```
##
##            Levene's F df1 df2     p
##
## DV: perf       0.235   3  16  .870
##
```

```
# 因为不做 repeated-measured ANOVA, 所以 subID 可以不用明确是哪一列
```

宽数据

```
wide_data = data.frame(s1 = c(3,3,4,6,6,8),
                       s2 = c(5,3,5,7,8,8),
                       s3=c(8,5,8,9,8,10),
                       s4 =c(8,9,7,10,10,10))
repeated_fit=MANOVA(wide_data,
                    dvs = 's1:s4',
                    dvs.pattern = 's(.)',
                    within='s',
                    sph.correction = 'GG')
```

```
##
## Note:
## dvs="s1:s4" is matched to variables:
## s1, s2, s3, s4
```

```
##
## ====== ANOVA (Within-Subjects Design) ======
##
## Descriptives:
##
##  "s"  Mean    S.D. n
##
##   s1 5.000 (2.000) 6
##   s2 6.000 (2.000) 6
##   s3 8.000 (1.673) 6
##   s4 9.000 (1.265) 6
```

```
##
## Total sample size: N = 6
##
## ANOVA Table:
## Dependent variable(s):      s1, s2, s3, s4
## Between-subjects factor(s): -
## Within-subjects factor(s):  s
## Covariate(s):               -
##
##        MS   MSE   df1    df2      F     p     ²p [90% CI of  ²p]   ²G
##
## s  27.551 1.286 2.178 10.889 21.429 <.001 ***   .811 [.558, .894] .492
##
## Sphericity correction method: GG (Greenhouse-Geisser)
## MSE = mean square error (the residual variance of the linear model)
##  ²p = partial eta-squared = SS / (SS + SSE) = F * df1 / (F * df1 + df2)
##  ²p = partial omega-squared = (F - 1) * df1 / (F * df1 + df2 + 1)
##  ²G = generalized eta-squared (see Olejnik & Algina, 2003)
## Cohen' s f² =  ²p / (1 -  ²p)
##
## Levene' s Test for Homogeneity of Variance:
## No between-subjects factors. No need to do the Levene' s test.
##
## Mauchly' s Test of Sphericity:
##
##    Mauchly's W     p
##
## s       0.5510  .824
##
```

between-subject design

尝试将上述的 `long_data` 中的 `subj` 一列去除，这相当于不知道我们三种 `strategy` 下用的是同样的一批被试，因此我们只能进行被试间检验 (one-

way ANOVA)

```
between_data=data.frame(score,strategy) %>% convert_as_factor(strategy)
# 注：MANOVA 传参时，between 变量可以不是因子类型，照样可以得到一样的结果
between_fit=MANOVA(between_data,
                   dv='score',
                   between='strategy')
```

```
##
## ====== ANOVA (Between-Subjects Design) ======
##
## Descriptives:
##
##  "strategy"  Mean    S.D. n
##
##   strategy1 5.000 (2.000) 6
##   strategy2 6.000 (2.000) 6
##   strategy3 8.000 (1.673) 6
##   strategy4 9.000 (1.265) 6
##
## Total sample size: N = 24
##
## ANOVA Table:
## Dependent variable(s):      score
## Between-subjects factor(s): strategy
## Within-subjects factor(s):   -
## Covariate(s):                -
##
##               MS   MSE df1 df2     F     p      ²p [90% CI of  ²p]   ²G
##
## strategy  20.000 3.100   3  20 6.452  .003 **    .492 [.164, .656] .492
##
## MSE = mean square error (the residual variance of the linear model)
##  ²p = partial eta-squared = SS / (SS + SSE) = F * df1 / (F * df1 + df2)
```

```
##  ²p = partial omega-squared = (F - 1) * df1 / (F * df1 + df2 + 1)
##  ²G = generalized eta-squared (see Olejnik & Algina, 2003)
## Cohen' s f² =  ²p / (1 -  ²p)
##
## Levene' s Test for Homogeneity of Variance:
##
##            Levene' s F df1 df2     p
##
## DV: score       1.067   3  20  .385
##
```

将被试间方差分析 (one-way ANOVA) 和被试内方差分析 (repeated-measured ANOVA) 的结果相互比较，发现后者更显著。有意思的是，当用单因素方差分析时，得到的效应量是 η^2，而使用重复测量方差分析时，得到的效应量应为 *partial* η^2，这在两种检验的结果都得到了验证。

5 事后检验

如果 ANOVA 的结果显著，那么需要紧接着进行事后检验，判定是哪两组的差异显著。事后检验较为保守，即使 ANOVA 显著，事后检验也未必得到显著的结果。

5.1 EMMEANS

```
EMMEANS(model,effect=NULL,p.adjust='bonferroni')
```

EMMEANS 能完成两项重要的任务：* 事后检验 * 简单主效应分析

此外，EMMEANS 还能够返回效应量 (partial η^2,Cohen's d，及其置信区间)

向 EMMEANS 中传入**由 MANOVA 得到的方差分析模型**，设置待检验的效应（effect 和 by），以及调整方式即可。

调整方式默认为 p.adjust='bonferroni'，也可以设置为'tukey','scheffe'。

```
# p.adjust='tukey'
EMMEANS(between_fit,effect = 'strategy',p.adjust = 'tukey')
```

5.1.0.1 EMMEANS 用于事后检验

```
## ------ EMMEANS (effect = "strategy") ------
##
## Joint Tests of "strategy":
##
##    Effect df1 df2     F     p      ²p [90% CI of  ²p]
##
##  strategy   3  20 6.452  .003 **    .492 [.164, .656]
##
## Note. Simple effects of repeated measures with 3 or more levels
## are different from the results obtained with SPSS MANOVA syntax.
##
## Estimated Marginal Means of "strategy":
##
##  "strategy" Mean [95% CI of Mean]    S.E.
##
##   strategy1 5.000 [3.501,  6.499] (0.719)
##   strategy2 6.000 [4.501,  7.499] (0.719)
##   strategy3 8.000 [6.501,  9.499] (0.719)
##   strategy4 9.000 [7.501, 10.499] (0.719)
##
##
## Pairwise Comparisons of "strategy":
##
##               Contrast Estimate    S.E. df     t     p     Cohen' s d [95% CI of d]
##
##  strategy2 - strategy1    1.000 (1.017) 20 0.984  .760       0.568 [-1.048, 2.184]
##  strategy3 - strategy1    3.000 (1.017) 20 2.951  .036 *     1.704 [ 0.088, 3.320]
##  strategy3 - strategy2    2.000 (1.017) 20 1.967  .233       1.136 [-0.480, 2.752]
```

```
##  strategy4 - strategy1    4.000 (1.017) 20 3.935  .004 **    2.272 [ 0.656, 3.888]
##  strategy4 - strategy2    3.000 (1.017) 20 2.951  .036 *     1.704 [ 0.088, 3.320]
##  strategy4 - strategy3    1.000 (1.017) 20 0.984  .760       0.568 [-1.048, 2.184]
##
## Pooled SD for computing Cohen' s d: 1.761
## P-value adjustment: Tukey method for comparing a family of 4 estimates.
##
## Disclaimer:
## By default, pooled SD is Root Mean Square Error (RMSE).
## There is much disagreement on how to compute Cohen' s d.
## You are completely responsible for setting `sd.pooled`.
## You might also use `effectsize::t_to_d()` to compute d.
```

也可以调整 p.adjust='scheffe'，结果会更难显著。

```
EMMEANS(between_fit,effect = 'strategy',p.adjust = 'scheffe')
```

```
## ------ EMMEANS (effect = "strategy") ------
##
## Joint Tests of "strategy":
##
##    Effect df1 df2     F     p      ²p [90% CI of  ²p]
##
##  strategy   3  20 6.452  .003 **    .492 [.164, .656]
##
## Note. Simple effects of repeated measures with 3 or more levels
## are different from the results obtained with SPSS MANOVA syntax.
##
## Estimated Marginal Means of "strategy":
##
##  "strategy" Mean [95% CI of Mean]    S.E.
##
##   strategy1 5.000 [3.501,  6.499] (0.719)
##   strategy2 6.000 [4.501,  7.499] (0.719)
##   strategy3 8.000 [6.501,  9.499] (0.719)
```

```
##   strategy4 9.000 [7.501, 10.499] (0.719)
##
##
## Pairwise Comparisons of "strategy":
##
##               Contrast Estimate    S.E. df     t     p     Cohen' s d [95% CI of d]
##
##  strategy2 - strategy1    1.000 (1.017) 20 0.984  .809       0.568 [-1.192, 2.328]
##  strategy3 - strategy1    3.000 (1.017) 20 2.951  .060 .     1.704 [-0.056, 3.464]
##  strategy3 - strategy2    2.000 (1.017) 20 1.967  .305       1.136 [-0.624, 2.896]
##  strategy4 - strategy1    4.000 (1.017) 20 3.935  .008 **    2.272 [ 0.512, 4.032]
##  strategy4 - strategy2    3.000 (1.017) 20 2.951  .060 .     1.704 [-0.056, 3.464]
##  strategy4 - strategy3    1.000 (1.017) 20 0.984  .809       0.568 [-1.192, 2.328]
##
## Pooled SD for computing Cohen' s d: 1.761
## P-value adjustment: Scheffe method with rank 3.
##
## Disclaimer:
## By default, pooled SD is Root Mean Square Error (RMSE).
## There is much disagreement on how to compute Cohen' s d.
## You are completely responsible for setting `sd.pooled`.
## You might also use `effectsize::t_to_d()` to compute d.
```

5.1.0.2 EMMEANS 用于简单主效应分析

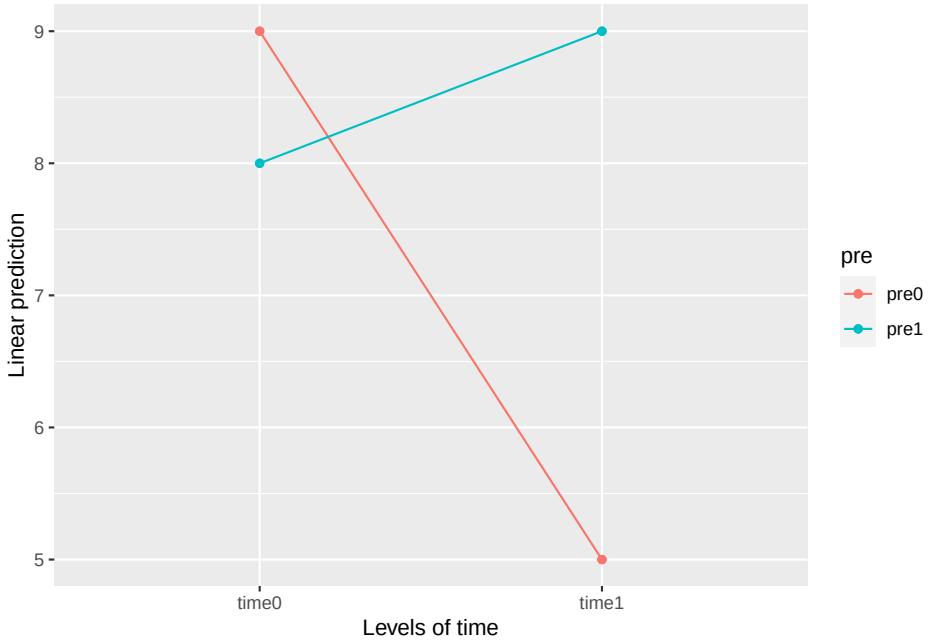
`effect` 参数传入的即为要检验的简单主效应，`by` 参数传入的是分组。

```
EMMEANS(two_fit,effect = 'pre',by = 'time') # 检验 pre 的简单主效应
```

```
## ------ EMMEANS (effect = "pre") ------
##
## Joint Tests of "pre":
##
##  Effect "time" df1 df2      F     p      ²p [90% CI of  ²p]
##
```

```
##     pre  time0   1  16  0.833  .375       .049 [.000, .291]
##     pre  time1   1  16 13.333  .002 **    .455 [.146, .659]
##
## Note. Simple effects of repeated measures with 3 or more levels
## are different from the results obtained with SPSS MANOVA syntax.
##
## Estimated Marginal Means of "pre":
##
##  "pre" "time" Mean [95% CI of Mean]    S.E.
##
##   pre0  time0 9.000 [7.358, 10.642] (0.775)
##   pre1  time0 8.000 [6.358,  9.642] (0.775)
##   pre0  time1 5.000 [3.358,  6.642] (0.775)
##   pre1  time1 9.000 [7.358, 10.642] (0.775)
##
##
## Pairwise Comparisons of "pre":
##
##     Contrast "time" Estimate    S.E. df      t     p     Cohen' s d [95% CI of d]
##
##  pre1 - pre0  time0   -1.000 (1.095) 16 -0.913  .375      -0.577 [-1.918, 0.763]
##  pre1 - pre0  time1    4.000 (1.095) 16  3.651  .002 **    2.309 [ 0.969, 3.650]
##
## Pooled SD for computing Cohen' s d: 1.732
## No need to adjust p values.
##
## Disclaimer:
## By default, pooled SD is Root Mean Square Error (RMSE).
## There is much disagreement on how to compute Cohen' s d.
## You are completely responsible for setting `sd.pooled`.
## You might also use `effectsize::t_to_d()` to compute d.
```

```
emmip(two_fit,pre~time) # 可以绘制 pre 关于 time 分组的简单主效应，其中纵轴为 pre，横轴为
```

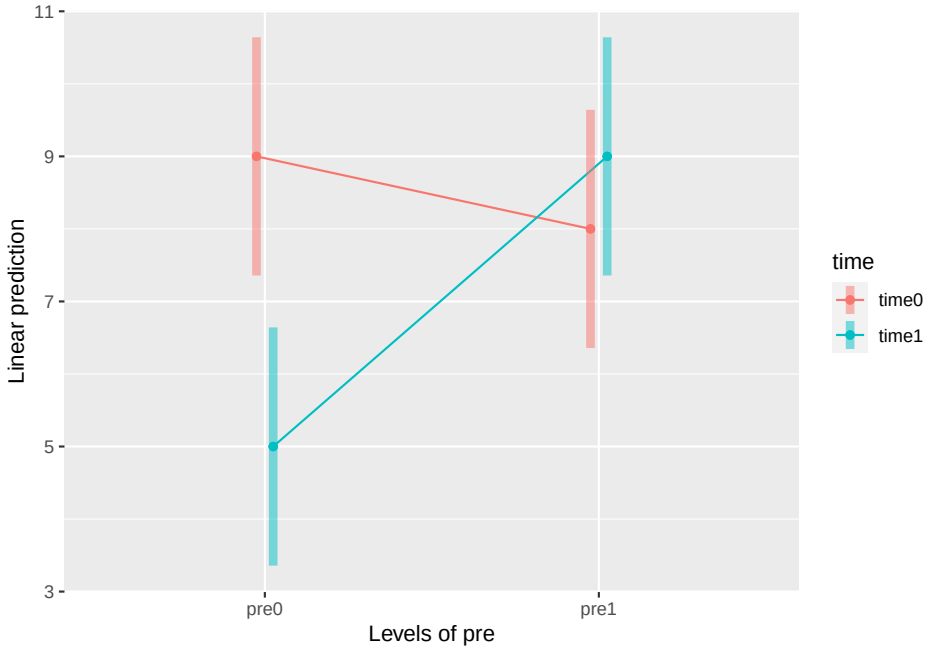



```
EMMEANS(two_fit, effect = "time", by = "pre")
```

```
## ------ EMMEANS (effect = "time") ------
##
## Joint Tests of "time":
##
##  Effect "pre" df1 df2      F     p      ²p [90% CI of  ²p]
##
##    time  pre0   1  16 13.333  .002 **    .455 [.146, .659]
##    time  pre1   1  16  0.833  .375       .049 [.000, .291]
##
## Note. Simple effects of repeated measures with 3 or more levels
## are different from the results obtained with SPSS MANOVA syntax.
##
## Estimated Marginal Means of "time":
##
##  "time" "pre" Mean [95% CI of Mean]    S.E.
##
```

```
##   time0  pre0 9.000 [7.358, 10.642] (0.775)
##   time1  pre0 5.000 [3.358,  6.642] (0.775)
##   time0  pre1 8.000 [6.358,  9.642] (0.775)
##   time1  pre1 9.000 [7.358, 10.642] (0.775)
##
##
## Pairwise Comparisons of "time":
##
##       Contrast "pre" Estimate    S.E. df      t     p     Cohen' s d [95% CI of d]
##
##  time1 - time0  pre0   -4.000 (1.095) 16 -3.651  .002 **  -2.309 [-3.650, -0.969]
##  time1 - time0  pre1    1.000 (1.095) 16  0.913  .375      0.577 [-0.763,  1.918]
##
## Pooled SD for computing Cohen' s d: 1.732
## No need to adjust p values.
##
## Disclaimer:
## By default, pooled SD is Root Mean Square Error (RMSE).
## There is much disagreement on how to compute Cohen' s d.
## You are completely responsible for setting `sd.pooled`.
## You might also use `effectsize::t_to_d()` to compute d.
```

```
emmip(two_fit,time~pre,CIs = T)
```




```
EMMEANS(two_fit,effect = 'pre')
```

```
## ------ EMMEANS (effect = "pre") ------
##
## Joint Tests of "pre":
##
##      Effect df1 df2      F     p      ²p [90% CI of  ²p]
##
##  time         1  16  3.750  .071 .     .190 [.000, .454]
##  pre          1  16  3.750  .071 .     .190 [.000, .454]
##  time * pre   1  16 10.417  .005 **    .394 [.095, .617]
##
## Note. Simple effects of repeated measures with 3 or more levels
## are different from the results obtained with SPSS MANOVA syntax.
##
## Estimated Marginal Means of "pre":
##
##  "pre" Mean [95% CI of Mean]    S.E.
```

```
##
##   pre0  7.000 [5.839, 8.161] (0.548)
##   pre1  8.500 [7.339, 9.661] (0.548)
##
##
## Pairwise Comparisons of "pre":
##
##     Contrast Estimate    S.E. df     t     p     Cohen' s d [95% CI of d]
##
##  pre1 - pre0    1.500 (0.775) 16 1.936  .071 .     0.866 [-0.082, 1.814]
##
## Pooled SD for computing Cohen' s d: 1.732
## Results are averaged over the levels of: time
## No need to adjust p values.
##
## Disclaimer:
## By default, pooled SD is Root Mean Square Error (RMSE).
## There is much disagreement on how to compute Cohen' s d.
## You are completely responsible for setting `sd.pooled`.
## You might also use `effectsize::t_to_d()` to compute d.
```

5.2 TukeyHSD

`TukeyHSD(x, conf.level = 0.95, ...)`, 来自原生 `stats` 包。

其中 `x` 是拟合模型，这里通常适用 `aov` 拟合的模型。`conf.level` 可以设定置信区间的大小，通常和显著性水平相对应。

`diff` 是两组的均值差异，`p adj` 是调整后的 p-value。

`TukeyHSD` 类可以用 `print` 显示结果，也可以用 `plot` 展现事后两两比较的图，图上呈现了均值差异和置信区间，非常直观。

```
tfm=TukeyHSD(fit_model)
print(tfm)
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = score ~ strategy, data = long_data)
##
## $strategy
##     diff        lwr      upr     p adj
## 2-1    1 -1.8452027 3.845203 0.7601658
## 3-1    3  0.1547973 5.845203 0.0364863
## 4-1    4  1.1547973 6.845203 0.0041906
## 3-2    2 -0.8452027 4.845203 0.2331646
## 4-2    3  0.1547973 5.845203 0.0364863
## 4-3    1 -1.8452027 3.845203 0.7601658
```

```
plot(tfm)
```

5.3 tukey_hsd

`tukey_hsd(x, formula, ...)`, 来自 `rstatix` 包。

`tukey_hsd(x, formula, ...)` 是管道友好的。

其中 `x` 为 `aov`,`lm`,`data.frame` 等包含着 `formula` 中的变量的数据类型，`formula` 为 `num~group`，其中 `num` 为数值变量，`group` 为分类变量。如果传入了方差分析模型 `aov`，那么就不用再输入 `formula`，因为 `formula` 在 `aov` 中就已经体现。

`tukey_hsd(x, formula, ...)` 能返回一个 tibble data frame，反馈所有的结果（呈现效果更美观）。

```
# 直接传入 aov 模型
tukey_hsd(fit_model)
```

```
## # A tibble: 6 x 9
##   term     group1 group2 null.value estimate conf.low conf.high   p.adj p.adj.~1
## * <chr>    <chr>  <chr>       <dbl>    <dbl>    <dbl>     <dbl>   <dbl> <chr>
## 1 strategy 1      2               0     1.00   -1.85       3.85 0.76    ns
## 2 strategy 1      3               0     3       0.155      5.85 0.0365  *
## 3 strategy 1      4               0     4       1.15       6.85 0.00419 **
## 4 strategy 2      3               0     2      -0.845      4.85 0.233   ns
## 5 strategy 2      4               0     3       0.155      5.85 0.0365  *
## 6 strategy 3      4               0     1      -1.85       3.85 0.76    ns
## # ... with abbreviated variable name 1: p.adj.signif
```

```
# 使用管道
long_data %>% tukey_hsd(score~strategy)
```

```
## # A tibble: 6 x 9
##   term     group1 group2 null.value estimate conf.low conf.high   p.adj p.adj.~1
## * <chr>    <chr>  <chr>       <dbl>    <dbl>    <dbl>     <dbl>   <dbl> <chr>
## 1 strategy 1      2               0     1.00   -1.85       3.85 0.76    ns
## 2 strategy 1      3               0     3       0.155      5.85 0.0365  *
## 3 strategy 1      4               0     4       1.15       6.85 0.00419 **
```

```
## 4 strategy 2      3               0     2      -0.845      4.85 0.233   ns
## 5 strategy 2      4               0     3       0.155      5.85 0.0365  *
## 6 strategy 3      4               0     1      -1.85       3.85 0.76    ns
## # ... with abbreviated variable name 1: p.adj.signif
```

6 补充

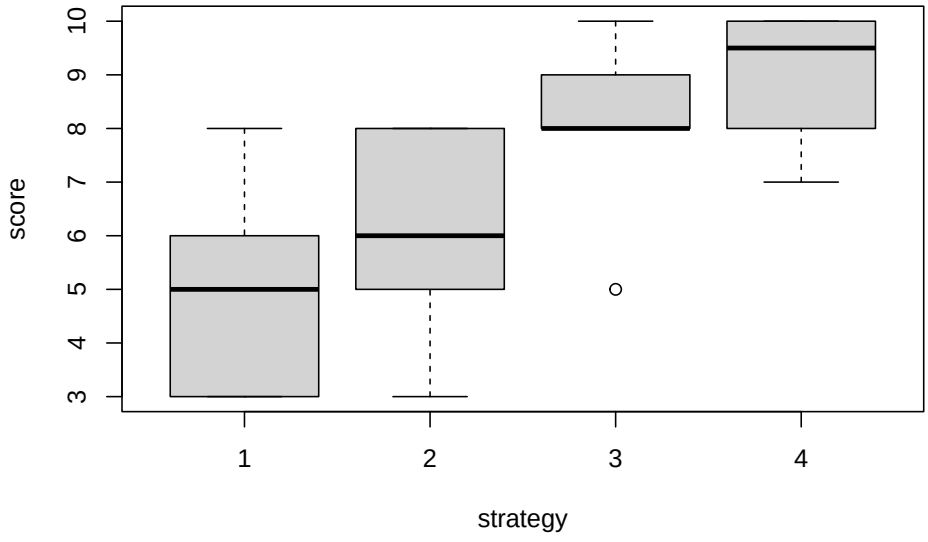
在进行方差分析之前，可以先

1. 绘制箱型图大概比较一下各组数据。纵轴上可以比较均值，宽度可以大致看出方差是否同质。
2. 绘制 QQ 图大致查看正态性假设是否成立

6.1 绘制箱型图

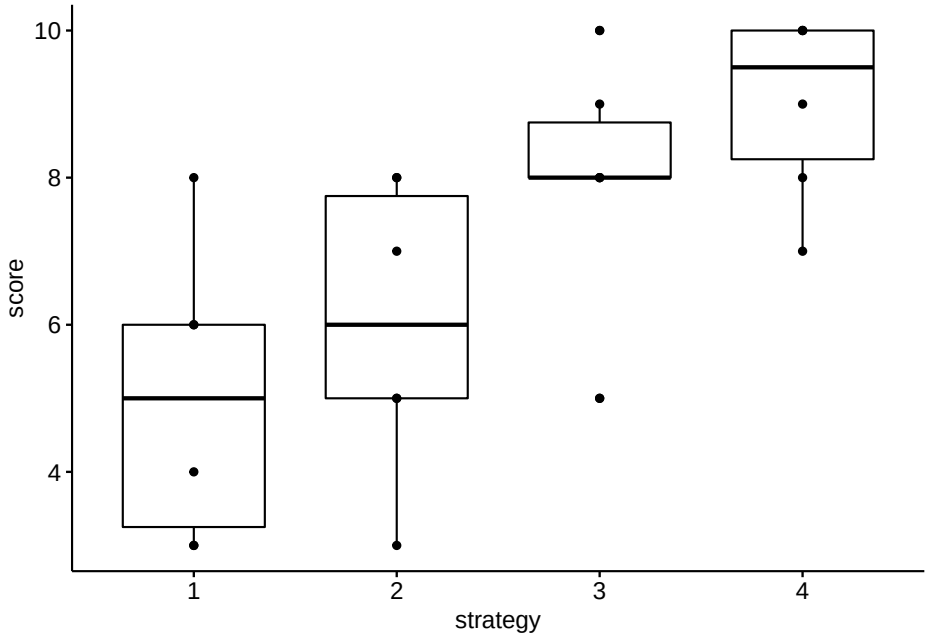
- boxplot
- ggboxplot

```
# 传统方法
boxplot(score~strategy,long_data,varwidth=T)
```



```
# 设置 varwidth=T，可以由箱型图的宽度看出各组数据的方差大致情况
# 如果宽度相近，也反映了方差可能同质。

# ggboxplot，画出来的效果可能更美观
ggboxplot(long_data, x = "strategy", y = "score", add = "point")
```



6.2 绘制 QQ 图

要根据不同的组别画出 QQ plot，因此使用 `ggqqplot` 函数

`ggqqplot` 函数：第一个参数传入数据框，第二个参数传入要绘制的变量，`facet.by` 传入的是分组变量

```
ggqqplot(long_data, "score", facet.by = "strategy")
```