応用プログラミング 3　第 2 回
R プログラミングの基本、最近の動向について 補足資料

専修大学ネットワーク情報学部
田中健太
2023-10-03

はじめに

本講義は「統計学入門」ではないため、時間を割いて取り上げられません[1]が、データ分析を行うための最初のステップとして、**基本統計量**を算出することが重要です。また、データ分析を伴う研究のステップとして、自分が考える仮説（対面講義のほうがオンライン講義よりも満足度が高いはずだ、など[2]）が成立するかどうか、**統計的検定**を行い、検証することが一般的です。この補足資料では、R を使って基本統計量を算出する方法、統計的検定を行う方法を紹介します。

R における基本統計量の算出

基本統計量とは、「1 つの数値でデータ全体の特徴をあらわす指標」です。1 つの数値で、データの中心はこのあたり、データのばらつきはこれくらい、といった情報を表現します。データ分析の最初の一歩は、基本統計量を眺めて、データの全体像を理解することです。ここでは、一般的に使われるいくつかの指標の意味と、R でどのように算出するかを紹介します。

なお、この資料では、R の組み込み関数の範囲で基本統計量を算出する方法を紹介します。パッケージを活用した、より効率的な処理については、以降の講義の中で紹介していきます。

データのサイズ

データ分析の最初の一歩は、データのサイズを知ることです。サイズとは、行数、列数などです。まず、ベクトルのサイズは `length()` 関数で得られます。

```
vec <- 1:100
length(vec)
```

---

[1] いちおう、講義の企画段階では、「他の授業で R を使っているみたいだし、このあたりの基本は理解しているだろう」と思っていたのですが、その「他の授業」をやってみると、どうもそうではなかったので、こうやって補足しています。

[2] 筆者は「90 分も知らんオッサンの話を硬い椅子に座って聞くくらいなら、家で、好きな時間に何回でもわかるまで動画で勉強したい」と思いますが（文科省の要請など、いろいろあっての対面授業なので、しょうがないのですが）。社会人研修になると、90 分どころでなく、9 時 17 時ずーっと、になります。

```
## [1] 100
```

次に、行列やデータフレームの行数、列数は、`nrow()`, `ncol()`, `dim()` 関数で得られます。

```
data(iris) # 組み込みの iris データセットを使う
nrow(iris) # 行数

## [1] 150

ncol(iris) # 列数

## [1] 5

dim(iris) # 行数、列数

## [1] 150   5
```

行列やデータフレームの 1 行あるいは 1 列はベクトルなので、`length()` 関数も使用できます。

```
length(iris[["Species"]]) # Species 列の件数 = 行数

## [1] 150

length(iris[1, ]) # 1 行目の件数 = 列数

## [1] 5
```

最小値

数値データのうち、最も小さい値が最小値です。`min()` 関数で求めます。

```
set.seed(334) # 結果が固定されるよう乱数のシードを設定
vec <- sample(1:100, 10) # 1 から 100 の間で 10 個整数の乱数を抽出
min(vec)

## [1] 1
```

最大値

数値データのうち、最も大きい値が最大値です。`max()` 関数で求めます。

```
set.seed(334) # 結果が固定されるよう乱数のシードを設定
vec <- sample(1:100, 10) # 1 から 100 の間で 10 個整数の乱数を抽出
max(vec)

## [1] 94
```

範囲

最小値と最大値のペアを範囲（Range）と言います。データがその範囲に分布している、ということがわかります。
範囲は `range()` 関数で求めます。

```
set.seed(334) # 結果が固定されるよう乱数のシードを設定
vec <- sample(1:100, 10) # 1 から 100 の間で 10 個整数の乱数を抽出
range(vec)

## [1]  1 94
```

平均

データの特徴を 1 つの数字であらわす**代表値**として、最も広く使われるのが平均（Average, Mean）です。正確には、平均にも種類があり、(1) 相加平均、(2) 相乗平均、(3) 調和平均などが使われますが、一般的には相加平均を「平均」として扱うことが多いです。相加平均は、`mean()` 関数で求めます。

```
set.seed(334) # 結果が固定されるよう乱数のシードを設定
vec <- rnorm(100, mean = 1, sd = 3) # 平均 1、標準偏差 3 の正規乱数を 100 個抽出
mean(vec)

## [1] 0.9806
```

トリム平均

データに外れ値（極端に大きい、または小さい値）が混じっていると、平均がそれに引っ張られて、データの「中心」をあらわさなくなります。そのような場合、もちろん適切な前処理を行い、外れ値を除外することもできますが、別のアプローチとして、データの上位および下位数パーセントを取り除いて平均を算出する、トリム平均というものもあります。フィギュアスケートなど、スポーツの採点競技で、最も高い点と最も低い点をつけた審査員の得点を除外して合計点の計算を行いますが、同じ考え方です。トリム平均は、`mean(..., trim = 割合)` と指定します。

```
set.seed(334) # 結果が固定されるよう乱数のシードを設定
vec1 <- rnorm(900, mean = 1, sd = 3)
vec2 <- rnorm(100, mean = 100, sd = 30)
vec <- c(vec1, vec2)
mean(vec) # 外れ値に引っ張られる

## [1] 10.95

mean(vec, trim = 0.1)

## [1] 1.591
```

割合は、0（デフォルト）から 0.5 まで指定できます。これは、指定した割合だけ**両端から**データを除いて平均を算出します。つまり、`trim = 0.2` とすると、実際には両端から 20%ずつ、計 40%のデータが除かれることになります。`trim = 0.5` とすると、中央の 1 つ（奇数）または 2 つ（偶数）のデータのみが残り、中央値と一致します。

欠損値の扱い

データには欠損値（値が記録されなかった“セル”）が存在することがあります。センサーや Web システムのログは、機器やネットワークの状態により、データが取得できないことが多々あります。そのような欠損値は、R では `NA` という特別な値として記録されます。[3]

Excel では、セルの範囲を選択すれば、欠損値を自動的に無視して計算してくれますが、R（や Python）では、欠損値は無視しないため、そのままでは計算結果が得られません。

```
vec <- c(1:10, NA) # 欠損値を含むベクトルを作成
min(vec) # 結果が得られない

## [1] NA

max(vec) # 結果が得られない

## [1] NA

mean(vec) # 結果が得られない

## [1] NA
```

別途、前処理で欠損値を除く、あるいは補間することもできますが、`mean()` 関数や以下で紹介する基本統計量を算出する関数には、欠損値を除いて計算する `na.rm` オプションがあります。デフォルトは `FALSE` ですが、`TRUE` と指定することで、欠損値を除いた結果が得られます。

```
min(vec, na.rm = TRUE) # 欠損値を除去して最小値を算出

## [1] 1

max(vec, na.rm = TRUE) # 欠損値を除去して最大値を算出

## [1] 10
```

---

[3] 正確には、「整数型の `NA`（`NA_integer_`）」「実数型の `NA`（`NA_real_`）」「文字列型の `NA`（`NA_character_`）」など、型によって異なる値として扱われ、例えば `is.numeric(NA_character_)` と実行すると、`FALSE` が返ってきます。

```r
mean(vec, na.rm = TRUE) # 欠損値を除去して平均を算出

## [1] 5.5
```

中央値

中央値（Median）も、データの「中心」をあらわす統計量として広く使われます。データが奇数個である場合は、並べ替えた際にちょうど中央になる値、偶数個である場合は、中央で隣り合う 2 つの値を足して 2 で割った値と定義されます。中央値は、`median()` 関数で求めます。`median()` 関数でも、`na.rm` オプションが指定できます。

```r
set.seed(334) # 結果が固定されるよう乱数のシードを設定
vec1 <- rnorm(900, mean = 1, sd = 3)
vec2 <- rnorm(100, mean = 100, sd = 30)
vec <- c(vec1, vec2)
median(vec)

## [1] 1.466
```

最頻値

最頻値（Mode）は、一連のカテゴリーデータのうち、最も多く出現するラベル（水準）のことを意味します。例えば、学生 100 人の出身都道府県を聞いた際に最も多い都道府県、などです。R では、最頻値を直接求める組み込み関数は存在しないので、`table()` 関数と `sort()` 関数あるいは `which.max()` 関数を組み合わせて求めます。ここでは、架空の日本酒成分データを使用し、日本酒の種類について、最頻値を求めます。

```r
df <- read.csv("pseudo_sake_data.csv")
# 説明のため、あえて個別に関数を適用していく
tbl <- table(df[["タイプ"]])
# sort() 関数による並べ替え
sort_tbl <- sort(tbl, decreasing = TRUE)
# 最頻値の出力
sort_tbl[1]

## 一般酒
##    647

# which.max() 関数による方法
tbl[which.max(tbl)]

## 一般酒
##    647
```

分散

分散（Variance）は、データが平均を中心にどの程度ばらついているかをあらわす指標です。個々のデータの平均との差（偏差）を 2 乗し、合計した値を、データ数で割って求めます。この際、データ数（n）で割ると**標本分散**（s^2）、$n-1$ で割ると**不偏分散**（σ^2）と呼ばれます。データ数が少ないときには、標本分散は母分散よりも小さくなります。そのため、一般的には不偏分散を使います。[4]標本分散と不偏分散の違い、母分散との誤差については、標本分散と不偏分散の関係についてにわかりやすい説明があります。R では `var()` 関数で不偏分散が得られます。

```
set.seed(334)
vec <- rnorm(100, mean = 10, sd = 3)
# 不偏分散を算出
var(vec)

## [1] 10.7
```

標本分散については、組み込み関数はないので、式を自分で実装する必要があります。

```
sample_var <- function(vec){
    vec <- na.omit(vec) # 欠損値を除去
    avg <- mean(vec) # 平均を算出
    dev <- vec - avg # 偏差を算出
    dev_square <- sum(dev^2) # 偏差平方和を算出
    sample_var <- dev_square / length(vec) # 標本分散を算出
    return(sample_var)
}
sample_var(vec)

## [1] 10.59
```

標準偏差

分散は、式の中で偏差を 2 乗しているため、単位がなくなります（無名数）。また、値も大きくなりがちです。そのような不便さを解消するため、一般に分散の平方根（$\sqrt{}$）を取った、標準偏差（SD; Standard deviation）を使います。R では、`sd()` 関数で、不偏標準偏差（σ）を直接算出できます。標本標準偏差（s）は、`sqrt()` 関数で標本分散の平方根として求めます。ただ、この σ や s といった記号の使い方は書籍や研究者によってまちまちで、不偏標準偏差（`sd()`）を s と表記することも多くあります。

---

[4] 手元のデータがすべて、つまり母集団である場合には、標本分散を使っても構いませんが、現実的にそのようなケースは少ないでしょう。

```r
set.seed(334)
vec <- rnorm(100, mean = 10, sd = 3)
# 不偏標準偏差を算出
sd(vec)

## [1] 3.271

# 標本標準偏差を算出
sqrt(sample_var(vec))

## [1] 3.254
```

四分位数

四分位数（Quartile）は、データを 4 等分した時の、0%, 25%, 50%, 75%, 100% の位置の値です。0%と 100%は、最小値と最大値なので、実際には 25%, 50%（中央値）, 75%の値を求めます。研究やビジネスで積極的に使う指標ではありませんが、データ全体の分布や、偏り（歪み）などを捉えられます。四分位数は、`quantile()` 関数や `summary()` 関数で得られます。なお、`summary()` 関数はスライドで触れた `plot()` 関数と同じく総称（ジェネリック）関数で、与えるデータによって振る舞いが変わります。数値からなるベクトルやデータフレームを与えると、最小値、最大値、平均と四分位数を出力します。

```r
quantile(vec)

##     0%    25%    50%    75%   100%
##  1.233  7.724 10.224 11.928 20.976

summary(vec)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1.23    7.72   10.22    9.98   11.93   20.98
```

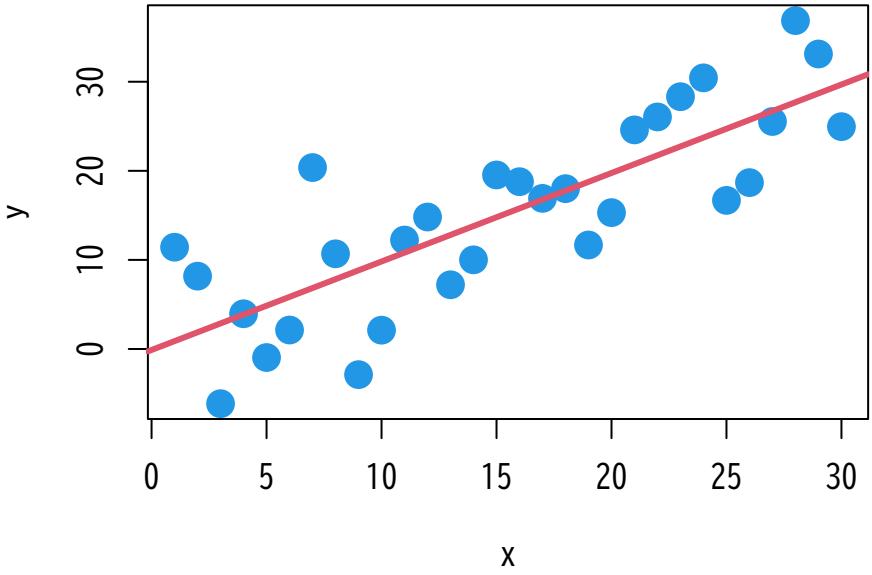
なお、四分位数にもさまざまな定義があり、実は `quantile()` 関数の `type` オプションに 1 から 9 もの異なる四分位数の定義を指定できます。詳しくは、三重大学奥村教授の四分位数の定義を参照してください。

データどうしの関係 – 相関と連関

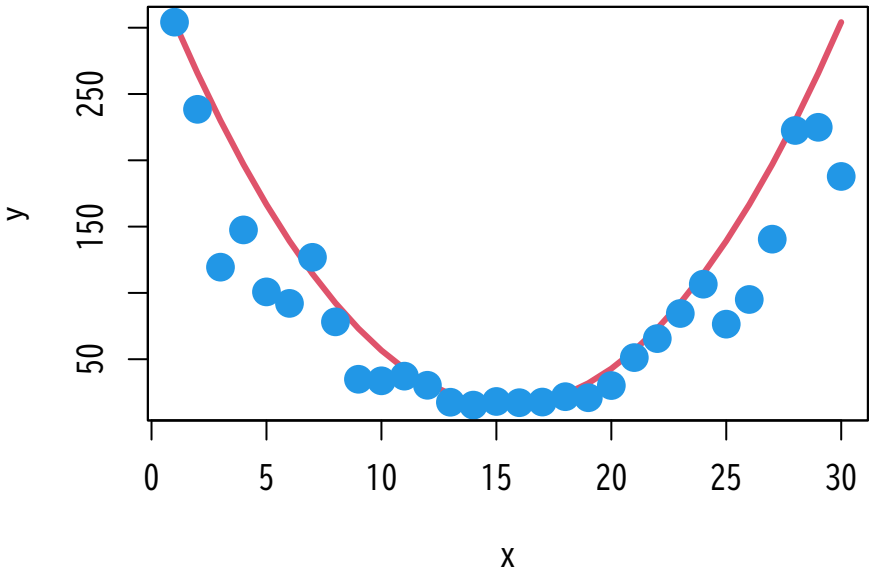
ここまで、1 変量（1 列のデータ）について、基本統計量を求める方法を紹介してきました。実際には、研究やビジネスにおいて、1 変量の分析だけで完結することはなく、2 変量、多変量（3 変量以上）のデータどうしの関係を考察することがほとんどです。「関係」には、この後紹介するような差がある、ないといったものや、第 9 回前後で取り上げる回帰や分類といった、x から y を予測するなど、さまざまな観点がありますが、まずは基本的なアプローチとして、相関と連関について取り上げます。

相関とは

相関（Correlation）は、データどうしの**線形関係**をあらわす概念です。広く一般に「データどうしの関係」といった意味で使われることが多いですが、統計学の考え方では、データ x と y の間に、直線（一次式）で表現できる関係があることを、「相関がある」と言います。



「相関がある」データ



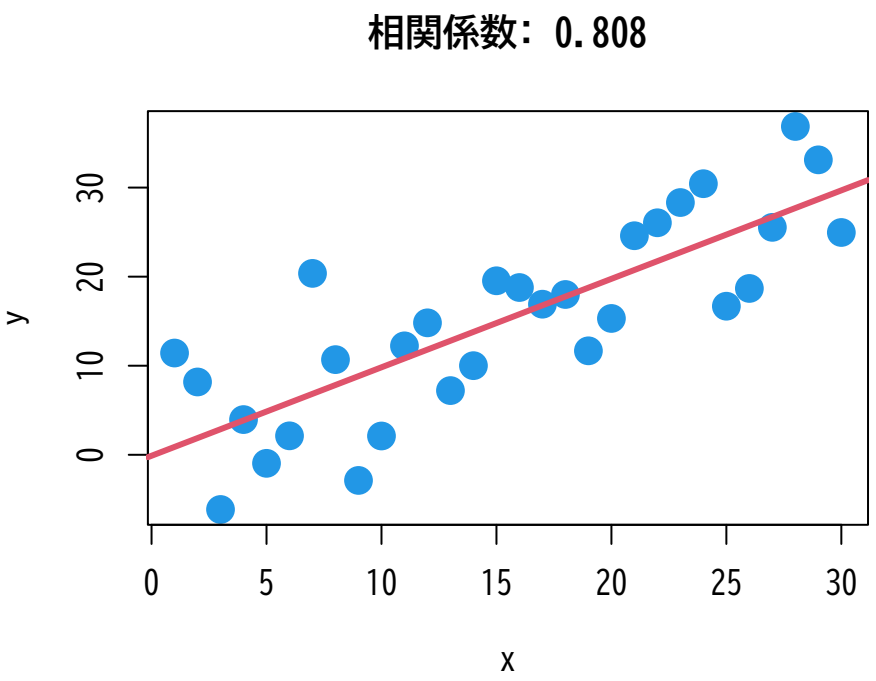
関係はあるが「相関はない」データ

このような場合に、上記のように散布図を描き、それを眺めて判断するということもできますが、より客観的で再現性のある解釈のため、一般には相関係数という指標を使います。

相関係数

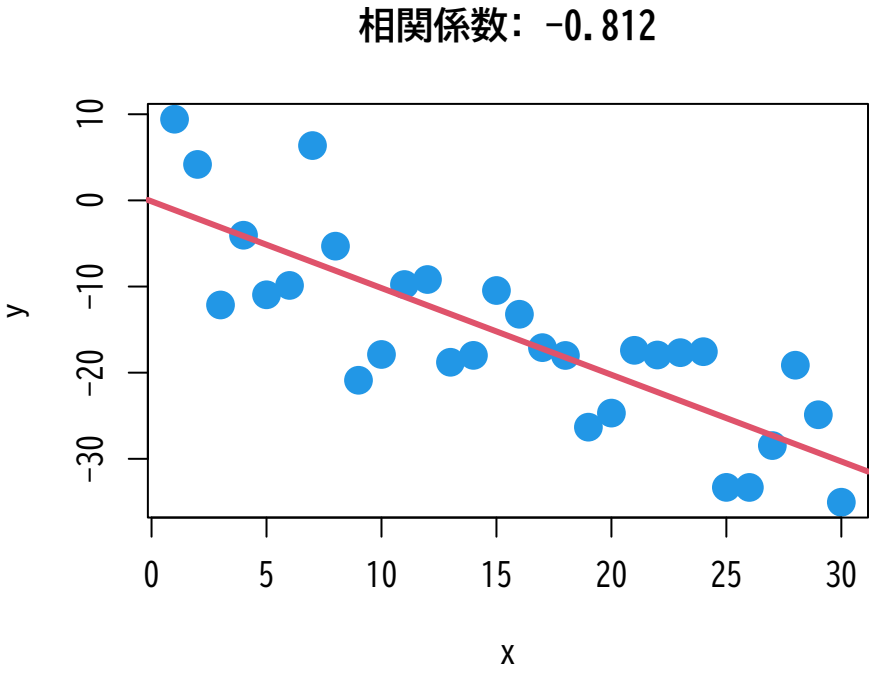
相関係数（Correlation coefficients; r）は、2 変量の相関関係の強さを表現する指標です。詳しい計算過程は省略します[5]が、データxとyについて、まず共分散（Covariance）を計算し、それを標準偏差で割ることで、-1 から+1 の範囲に正規化した値を求めます。R では、`cor()` 関数で相関係数を算出できます。

```
set.seed(3349800)
vec_x <- 1:30
vec_y <- vec_x + rnorm(30, sd = 6)
cor(vec_x, vec_y)

## [1] 0.8083

plot(vec_x, vec_y, col = 4, pch = 16, cex = 2, xlab = "x", ylab = "y", main = pas
te0("相関係数: ", round(cor(vec_x, vec_y), 3)))
res <- lm(vec_y ~ vec_x)
abline(res, col = 2, lwd = 3)
```

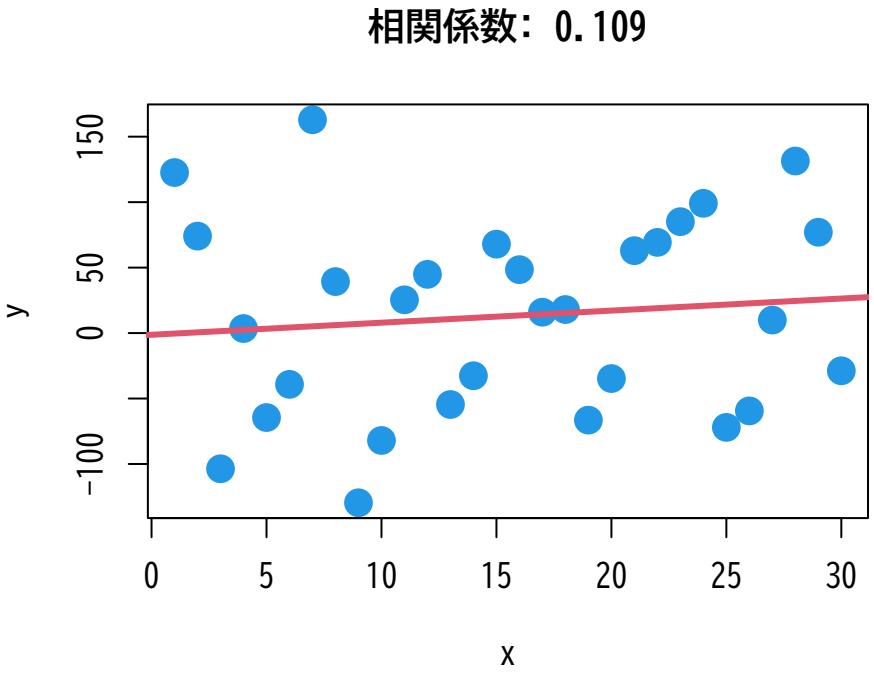


[5] 相関係数の計算過程は、例えば相関係数の意味と求め方 - 公式と計算例などで順を追って説明されています。

```r
set.seed(3349800)
vec_x <- 1:30
vec_y <- -vec_x + rnorm(30, sd = 6)
cor(vec_x, vec_y)

## [1] -0.812

plot(vec_x, vec_y, col = 4, pch = 16, cex = 2, xlab = "x", ylab = "y", main = pas
te0("相関係数: ", round(cor(vec_x, vec_y), 3)))
res <- lm(vec_y ~ vec_x)
abline(res, col = 2, lwd = 3)
```




```r
set.seed(3349800)
vec_x <- 1:30
vec_y <- vec_x + rnorm(30, sd = 70)
cor(vec_x, vec_y)

## [1] 0.1086

plot(vec_x, vec_y, col = 4, pch = 16, cex = 2, xlab = "x", ylab = "y", main = pas
te0("相関係数: ", round(cor(vec_x, vec_y), 3)))
res <- lm(vec_y ~ vec_x)
abline(res, col = 2, lwd = 3)
```

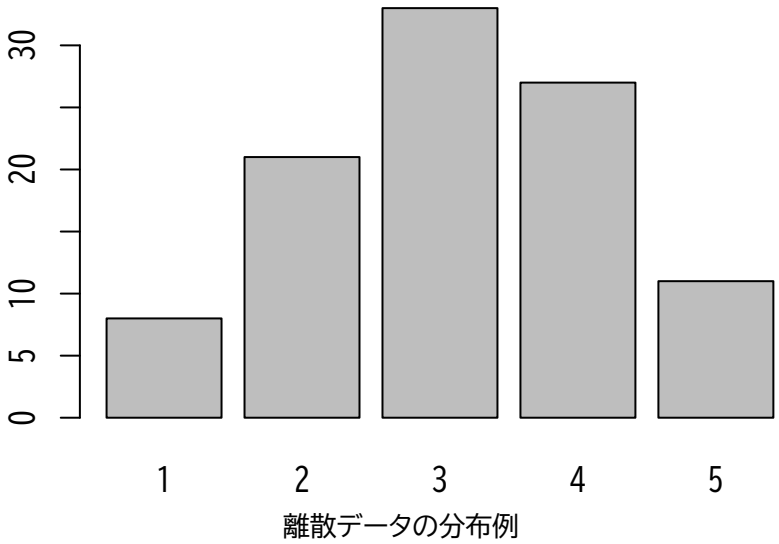


相関係数が-1 に近いほど、xが増加するとyが（線形に）減少する、**負の相関**関係が強いと言えます。+1 に近いほど、xが増加するとyも増加する、**正の相関**関係が強いと言えます。また、真ん中の 0 に近い値では、xとyの間に（線形）関係がない、**無相関**であると言えます。

順位相関係数

ここまで見てきた相関係数は、連続データの間の関係をあらわす、積率相関係数というものです。発案者の名前を取って、Pearson の相関係数とも呼ばれます。積率相関係数は、データx, yがいずれも、正規分布する母集団からサンプリングされた、という仮定を置いています。しかし、データが常に正規分布を仮定できるとは限りません。また、数学のテストの順位と、100m 走の順位といったような、**順位データ**については、1.6 位、3.5 位といった間の順位は存在せず、離散的なデータです。

```
vec <- sample(1:5, 100, replace = TRUE, prob = c(0.1, 0.2, 0.3, 0.2, 0.1))
barplot(table(vec))
```



このようなデータに対しても、`cor()` 関数で積率相関係数を求めることはできますが、相関関係を適切に表現したとは言えません。データが正規分布に従うと仮定できない場合は、**順位相関係数**を使います。Spearman が開発した手法と、Kendall が開発した手法が知られています。特に、現在では Spearman の方法を使うことが一般的です。Spearman の順位相関係数は、データを元の値（テストの点数や 100m 走のタイム）ではなく、順位に変換し、積率相関係数と同じ計算を行います。`cor()` 関数に、`method` オプションがあり、`"spearman"`（小文字）と指定すれば、順位相関係数を得られます。

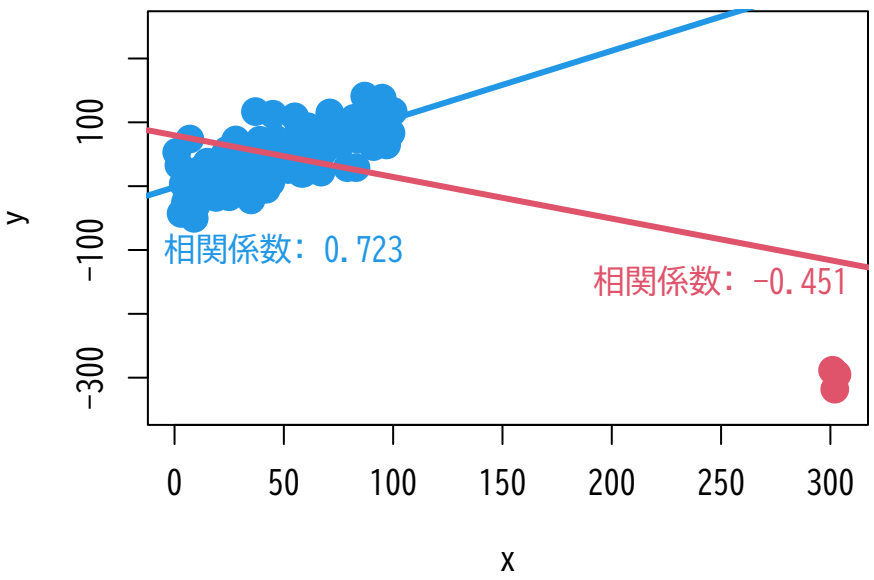
```
vec_x <- 1:10
vec_y <- c(1, 2, 3, 8, 7, 5, 6, 10, 4, 9)
cor(vec_x, vec_y, method = "spearman")

## [1] 0.6848
```

相関と外れ値

相関分析は、一連のデータ（変量）間の関係を捉えるうえで最も簡便な方法と言えますが、外れ値の影響には注意が必要です。相関係数は、すべてのデータをもとにxとyの線形関係を数値化しますので、散布図で全体を見ると相関関係がないようなデータでも、一部の外れ値の影響で、相関が発生したり、正と負の逆転が起こる場合があります。実際に、以下のコードでは、全体としては正の相関があるデータに、外れ値をわずか 3 つ混ぜることで、負の相関係数が得られるパターンを示しています。

```
# ggplot2 を使えばもっとシンプルに書けますが、
# まだ紹介していないので、あえて base R で
set.seed(3349800)
vec_x <- 1:100
vec_y <- vec_x + rnorm(100, sd = 30)
cor(vec_x, vec_y)

## [1] 0.7235

plot(vec_x, vec_y, col = 4, pch = 16, cex = 2, xlab = "x", ylab = "y", xlim = c
(0, 305), ylim = c(-350, 250))
res <- lm(vec_y ~ vec_x)
abline(res, col = 4, lwd = 3)
text(50, -100, paste0("相関係数: ", round(cor(vec_x, vec_y), 3)), col = 4)
vec_x <- c(1:100, 301:303)
vec_y <- c(vec_y, -301:-303 + rnorm(3, sd = 20))
cor(vec_x, vec_y)

## [1] -0.4509
```

```
points(301:303, -301:-303 + rnorm(3, sd = 20), col = 2, pch = 16, cex = 2)
res <- lm(vec_y ~ vec_x)
abline(res, col = 2, lwd = 3)
text(250, -150, paste0("相関係数: ", round(cor(vec_x, vec_y), 3)), col = 2)
```



外れ値で相関関係がまるっきり変わってしまう例

データの特徴を捉える段階では、基本統計量や相関係数といった数値だけでなく、グラフィックスも合わせて観察することが重要です。

連関とは

カテゴリーデータ間の関係性をあらわす概念として、連関（Association）があります。例えば、性別と、好きなお酒の種類を聞いた時に、2 つのカテゴリーデータの間に関係があるか、といったことを検討します。

```
tbl <- as.table(matrix(c(40, 15, 25, 20, 15, 35, 35, 15), nrow = 2, byrow = TRU
E))
rownames(tbl) <- c("男性", "女性")
colnames(tbl) <- c("ビール", "ワイン", "日本酒", "ウイスキー")
tbl

##       ビール  ワイン  日本酒  ウイスキー
## 男性      40      15      25          20
## 女性      15      35      35          15
```

上記のデータは架空のものです。

連関係数

カテゴリーデータのクロス集計表について、関係性をあらわす指標として、連関係数があります。なお、連関係数にも種類があり、よく使われるものに ϕ（ファイ）係数やクラメールのv（Cramer's v）などがあります。いずれも、さほど複雑な計算式ではないので、R プログラムとして実装することもできますが、そのような関数を提供するパッケージも数多くあります。例えば、CRAN でざっと検索すると、"Cramer's v" を算出する関数を提供するパッケージとして、以下のようなものがありました。

- psych
- rcompanion
- lsr
- vcd

他にもたくさんありますが、ここでは、"Visualizing Categorical Data" の略称である vcd パッケージが提供する、`assocstats()` 関数で、種々の連関係数をまとめて出力します。vcd パッケージは標準ではインストールされていないので、`install.packages()` 関数でインストールする必要があります。

```
install.packages("vcd")
```

ここでは、上記の性別と好きなお酒についての（架空の）クロス集計表を使用します。

```
library(vcd)
assocstats(tbl)
```

```
##                     X^2 df    P(> X^2)
## Likelihood Ratio 22.411  3 0.000053571
## Pearson          21.745  3 0.000073720
##
## Phi-Coefficient   : NA
## Contingency Coeff.: 0.313
## Cramer's V        : 0.33
```

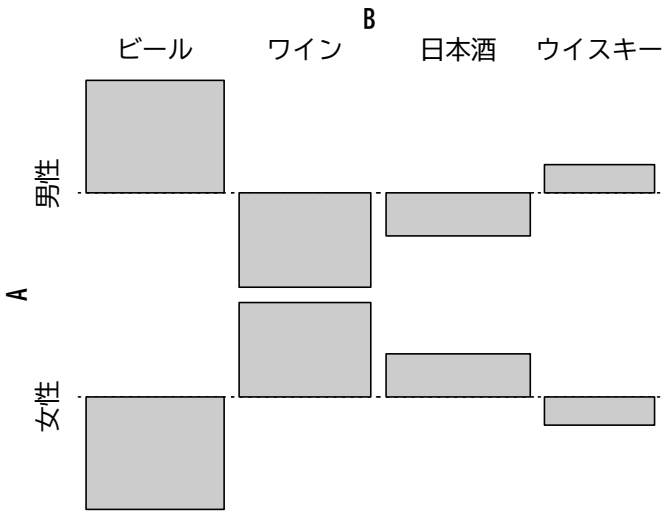
出力のうち、"Cramer's V" は 0.33 となっています。クラメールのvは、$0 \leqq v \leqq 1$の範囲の値を取ります。また、相関係数よりも小さい値から、「連関がある」と評価することが多いようです。そのため、0.3 程度であっても、「性別と好きなお酒の種類には関係がある」と言えそうです。他に、C（contingency）係数も出力されています。C係数の範囲、捉え方もクラメールのvと同様です。`assocstats()` 関数で算出されるファイ係数[6]は、2x2 の行列にのみ対応しているため、出力が `NA` になっています。

また、vcd パッケージでは、カテゴリーデータ間の関係性を可視化するための `assoc()` 関数も提供されています。ほぼ同じ結果が得られる、組み込みの `assocplot()` 関数もあります。

---

[6] ファイ係数の計算式にもいくつか種類があり、$n \times m$行列に対応した式もあります。

```
assoc(tbl)
```

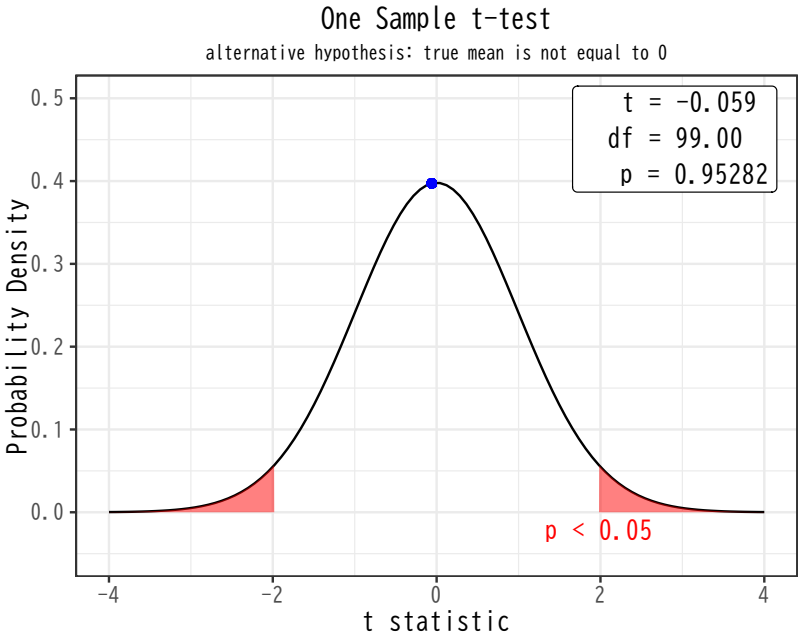


グラフィックスで表現すると、カテゴリーどうしの関係がわかりやすくなります。

ここまで、データ間の関係性をあらわす概念として、相関と連関を紹介しました。

R における統計的検定

基本統計量や集計表、相関や連関などは、データの特徴をありのまま表現したもので、それを踏まえた意思決定は人間が行う必要があります。例えば、何らかの条件（性別や季節や地域など）に従って収集した複数のデータ（標本）について、それらの間に差があるとみなすかどうかは、最終的には人間の判断です。ただ、「私は差があると思う」と主張するだけでは、誰も同意してくれないので、広く知られた基準に基づいて、データを評価する必要があります。データ間に差があるかを評価する手法として、**統計的検定**が広く使われます。



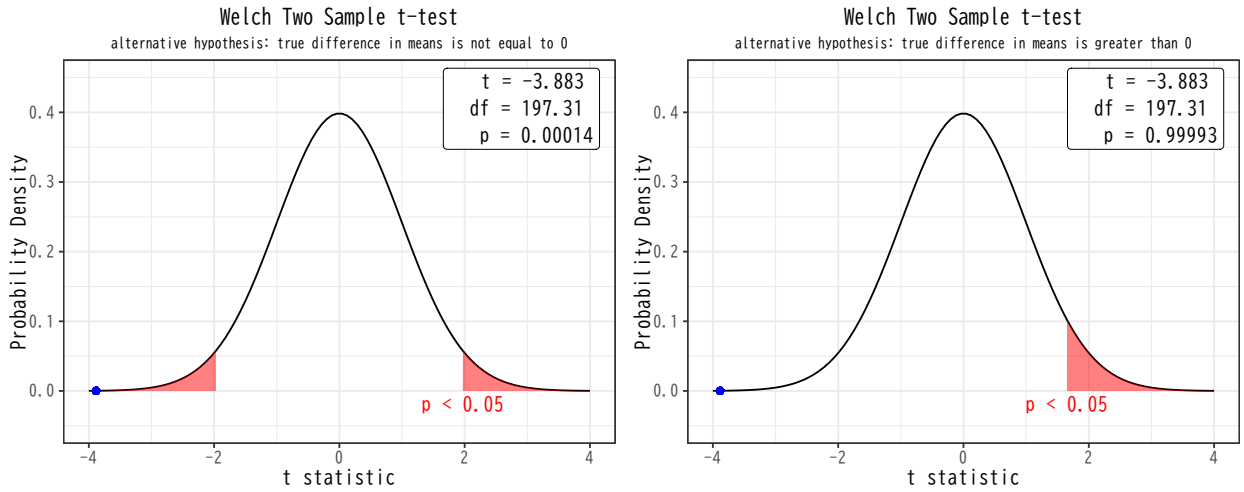
統計的検定のよくある図

仮説検証とは

統計的検定は、「データ間に差がある（またはない）はずだ」という仮説を立案し、それを検証するプロセスです。より具体的には、帰無仮説と対立仮説という、相反する 2 つの仮説を立案します。

帰無仮説（Null hypothesis）は、一般的に「成立して欲しくないネガティブな仮説」です。[7]統計的検定は差があるかどうかを検証しますので、多くの場合、帰無仮説は「差がない」というものになります。

対立仮説（Alternative hypothesis）は、一般的に「成立して欲しいポジティブな仮説」です。多くの場合、対立仮説は「差がある」というものになります。これらの 2 つの仮説のうち、実際のデータから妥当だと考えられるのはどちらであるかを検証します。もう少し具体的には、「このデータが得られる状況において、帰無仮説が成立する確率」を求めて、評価します。その確率が、あらかじめ決めておいた基準（有意水準）より低い場合、「こんなに低い確率でしか成立しない仮説は支持できない」として、帰無仮説を棄却し、対立仮説を採択します。一方、帰無仮説が成立する確率が有意水準より高かった場合、「そういうこともあり得る」として、帰無仮説を棄却できません。[8]なお、差があるということをより限定的に「処理速度が『速く』なった」「商品の認知度が『高く』なった」として検定することもできます。このような場合を**片側検定**と言います。一方、早くなっても遅くなっても、とにかく「差がある」ことを検証する場合を、**両側検定**と言います。直感的には、片側検定のほうが仮説検証の目的に合致していそうですが、研究においてもビジネスにおいても、片側検定はあまり使われません。詳細は両側検定、片側検定などを参照してください。



両側検定と片側検定

---

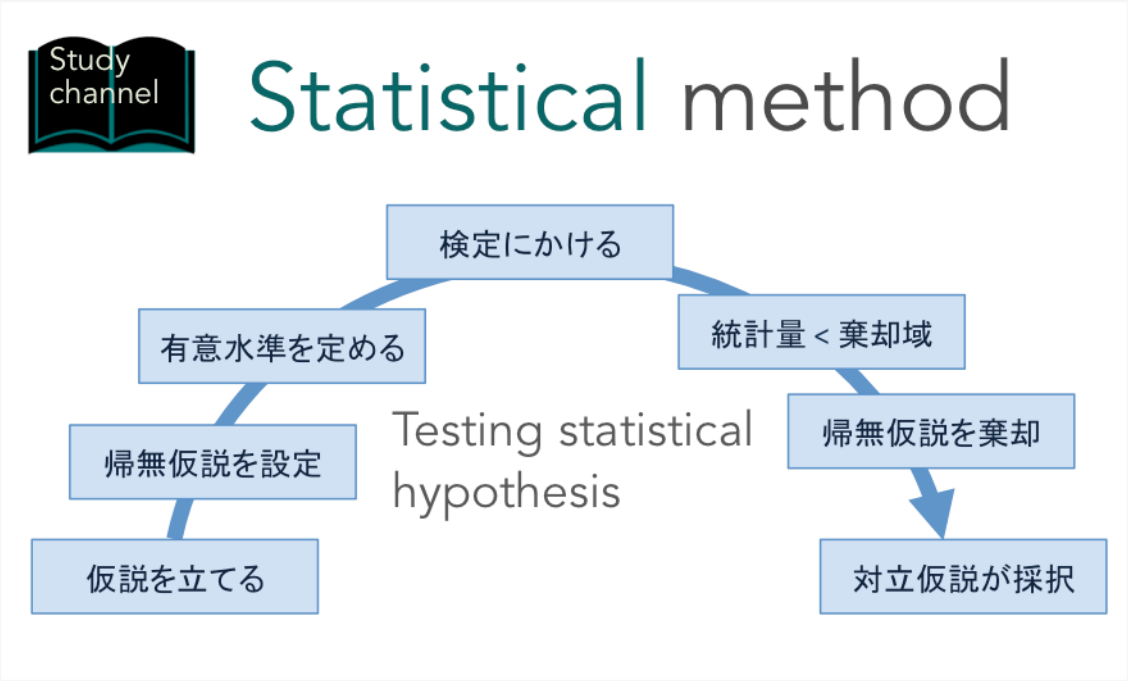
[7] 本来、科学は公平で客観的な観点で事実を捉えるものなので、成立して欲しいも欲しくないもないのですが、とはいえ

新しい薬の効果はあって欲しいですし、ビジネス施策も有効であって欲しいので、「差がない」という帰無仮説はネガティブなものとして扱われます。

[8] 帰無仮説は採択するものではありません。

統計的検定の基本的な流れ

統計的検定の概念は、上記のように仮説を立案し、それを検証するというものですが、より具体的な流れを紹介します。



1. 実験計画を立てる: 先行研究や他社事例などから、こういう施策の効果がありそうだ、というアイディアを立案し、その効果を測るための方法（必要なデータ、適用する検定手法など）を計画します。
2. 有意水準を決める: 上記のように帰無仮説を棄却する基準となる確率をあらかじめ定めておきます。一般的には、5%とします。薬など、よりシビアな検証が必要な場合は、1%などを設定する場合もあります。
3. 条件を決めてデータを収集する: 無計画に集めたデータ同士を比較することはできません。検証したい要因（薬の種類、広告の方法など）だけが異なり、他の要因（年代、性別など）を揃えた条件でデータを収集する必要があります。
4. 収集したデータを観察する: 計画通りにデータが集まるとは限りませんので、集めたデータの基本統計量を算出したりグラフィックスを描いたりして、データの特徴を把握します。その上で、当初の予定通り統計的検定が実行できそうかを判断します。
5. R で統計的検定を実行する: 本来は、ここで「検定統計量を算出し、棄却域を確認し、有意水準と比較して」という数学的なプロセスがありますが、R がやってくれますので、結果を見て評価します。
6. 結果を正しく記述する: 「検定したところ、差があった」だけでは科学的に正しい報告ではないので、使用した手法とソフトウェア、得られた値を適切にまとめ、レポートや論文に記載します。

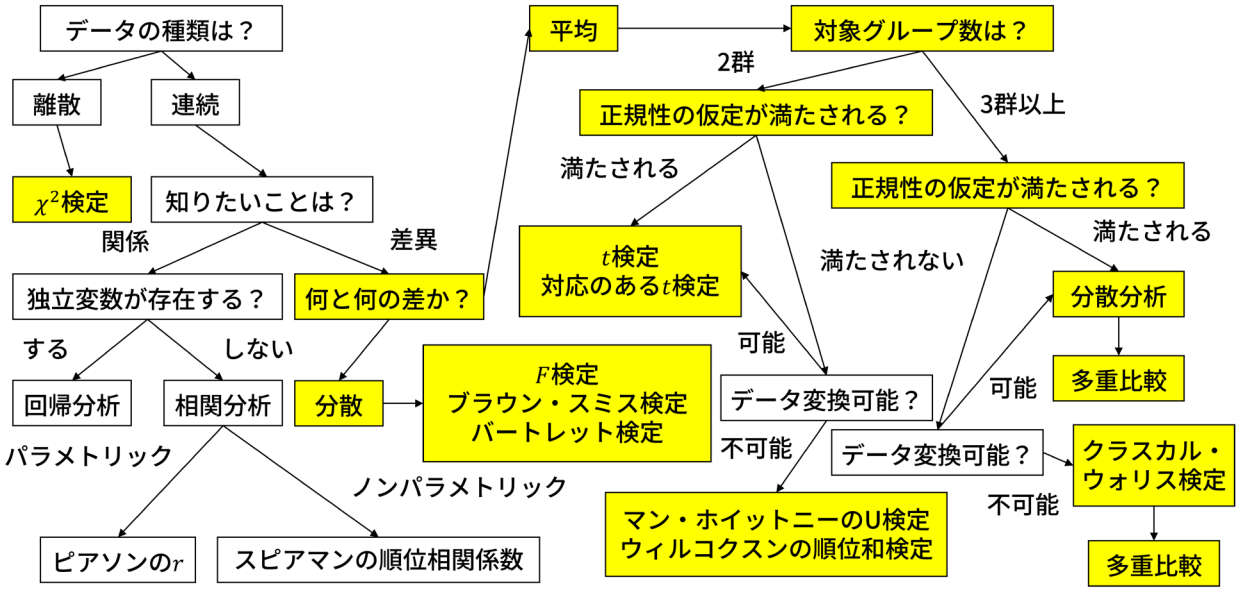
以降では、「R で統計的検定を実行する」部分にフォーカスして、いくつかの手法を紹介します。

平均の差の検定

はじめに、統計的検定の中でもポピュラーな、グループ間の平均の差を検定する手法を紹介します。実験計画やデータの特徴によって、適用すべき手法が変わりますので、目的と対応する手法を選択する知識が必要です。

検定手法の使い分け

比較の対象となるデータの特徴（分布）やグループの数によって、適切な検定手法を選択する必要があります。インターネットで検索すると、統計的検定の手法を選択するためのフローチャート的なものが多数あります。[9] 本文書では、以下の手法について、「R でどうやるか」ということを中心に紹介します。



http://abacus.bates.edu/~ganderso/biology/resources/stats_flow_chart_v2014.pdf を参考に作成

- 1 標本のt検定: 母集団が平均μの正規分布かどうかを検定。nが大きければ正規分布に近づくので、正規性は厳密ではない
- 2 標本のt検定: Welch のt検定。2 標本の分散が異なっても良いが、正規分布であることを仮定する
  - 対応がある / ないによる違い: 同一個体に対して複数回測定を行った結果かどうかで計算式が変わる
- Mann-Whitney のU検定: 中央値による検定。正規分布を仮定しないノンパラメトリック検定
- 分散分析: 3 つ以上のグループ間の平均の差を検定する
  - 一元配置分散分析: 1 つの因子によってグループ間に差が生じているかを検定する
  - 二元配置分散分析: 2 つの因子によってグループ間に差が生じているかを検定。交互作用を考慮する
  - 多重比較: 分散分析の結果、有意差があれば、具体的にどの条件で差があるかを検定する

1 標本のt検定

例えば製造業において、「新しい製造ラインで製造した部品の重量が、設計上の仕様値からずれていない（等しい）と言えるか」などを検証したい場合に使用できるでしょう。[10]

---

[9] 筆者も、昔ここで示した図を（海外の大学教員の資料を訳して）公開したら、少しバズったことがあります。

[10] 製造業においては、「統計的品質管理」という領域で、製品の品質や生産性、リスクなどを評価する手法が確立されています。華々しい「データサイエンス」の世界以外でも、データ分析が役立つ領域がたくさんあります。

組み込みの `t.test()` 関数を使い、オプション `mu` に母平均を指定することで、単一の標本の平均と母平均との差を検定できます。

```
# 乱数の固定
set.seed(334)

# 平均が 0.24、標準偏差 1 の正規分布に基づく乱数を抽出
vec <- rnorm(100, mean = 0.24)

# データの平均が母平均 0（帰無仮説）と等しいかを検定
t.test(vec, mu = 0)

##
##  One Sample t-test
##
## data:  vec
## t = 2.1, df = 99, p-value = 0.03
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  0.01722 0.44985
## sample estimates:
## mean of x
##    0.2335
```

2 標本のt検定

2 つのグループ間で平均値が異なる、つまり 2 つは異なる母集団から得られた標本である、ということを検証するのが 2 標本のt検定です。統計的検定といえば、という最もポピュラーなものです。前提として、2 つの標本はともに正規分布している、と仮定します。実際には、t検定は分布の形状に対してロバスト（頑健）であるとされていますが、あまりにかけ離れた分布形状のデータを扱う場合は、後述のU検定など、ノンパラメトリックな方法を使ったほうがよいでしょう。

また、古い統計学の本などでは、t検定の前に 2 標本の分散が等しいことをF検定で確認する、と書かれていることが
ありますが、これは現在では誤りとされています。検定を 2 回行うことで、誤った判断を下す確率が、有意水準以上になってしまいます。詳細は、28-4. Welch の t 検定 | 統計 WEB などを参照してください。

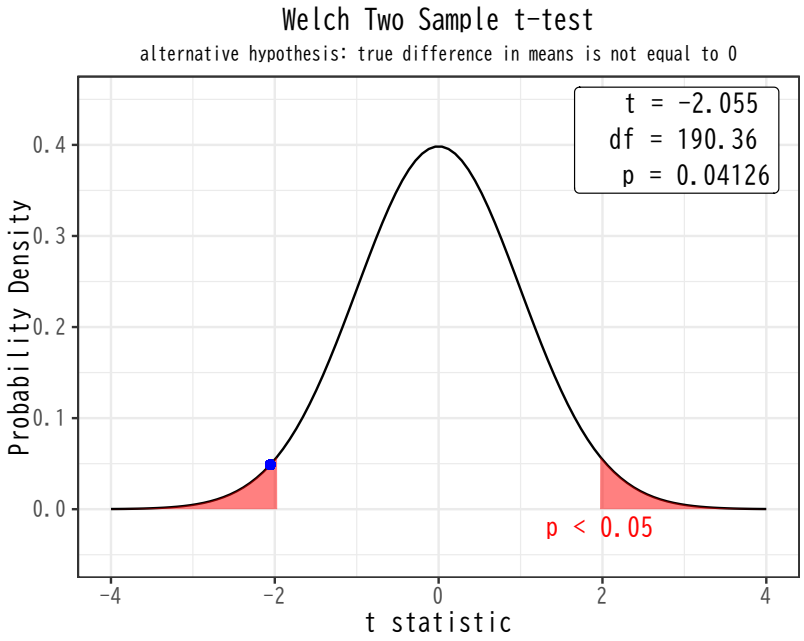
2 標本のt検定も、`t.test()` 関数で実行できます。

```
set.seed(334)
vec1 <- rnorm(100) # 平均 0, 標準偏差 1
```

```
vec2 <- rnorm(100, mean = 0.25, sd = 1.3)
t.test(vec1, vec2)

##
##  Welch Two Sample t-test
##
## data:  vec1 and vec2
## t = -2.1, df = 190, p-value = 0.04
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.6944 -0.0142
## sample estimates:
## mean of x mean of y
## -0.006467  0.347814
```

*t*検定の結果を、webr パッケージの `plot()` 関数で可視化できます。

```
res <- t.test(vec1, vec2)
plot(res) + ylim(-0.05, 0.45) + theme(plot.title = element_text(size=12), plot.su
btitle = element_text(size=8, hjust=0.5))
```



対応のあるデータの*t*検定

統計的検定には、対応がある、ないという概念があります。対応がある、とは同一個体（ヒト、動物、モノ）に対して 2 回計測を行ったデータである、という意味です。繰り返しのあるデータ、反復測定を行ったデータとも呼ばれます。

対応のあるデータでは、グループ全体の平均を比較するのではなく、同じ個体どうしで差分を取り、その差が 0 であるかを検定します。個体差を考慮して差を比較したほうが、個体差を無視して（対応のないデータとして）比較するよりも、グループ間に差があることを示しやすくなります。

R では、`t.test()` 関数に、`paired = TRUE` オプションを指定すると、対応のあるデータとしてt検定を行います。ここでは、実際に対応のあるデータとして計測された、組み込みの `sleep` データを使用します。これは、10 人の被験者に 2 種類の睡眠薬を投与し、睡眠時間の増加量を計測したものです。

```
sleep
```

```
##    extra group ID
## 1    0.7     1  1
## 2   -1.6     1  2
## 3   -0.2     1  3
## 4   -1.2     1  4
## 5   -0.1     1  5
...
```

このデータでは、同一の被験者について、薬剤の種類 `group` によって、睡眠時間の増加量 `extra` が変化することを検証します。縦にデータが並んでいるので、`結果 ~ 要因` という式でモデルを指定します。

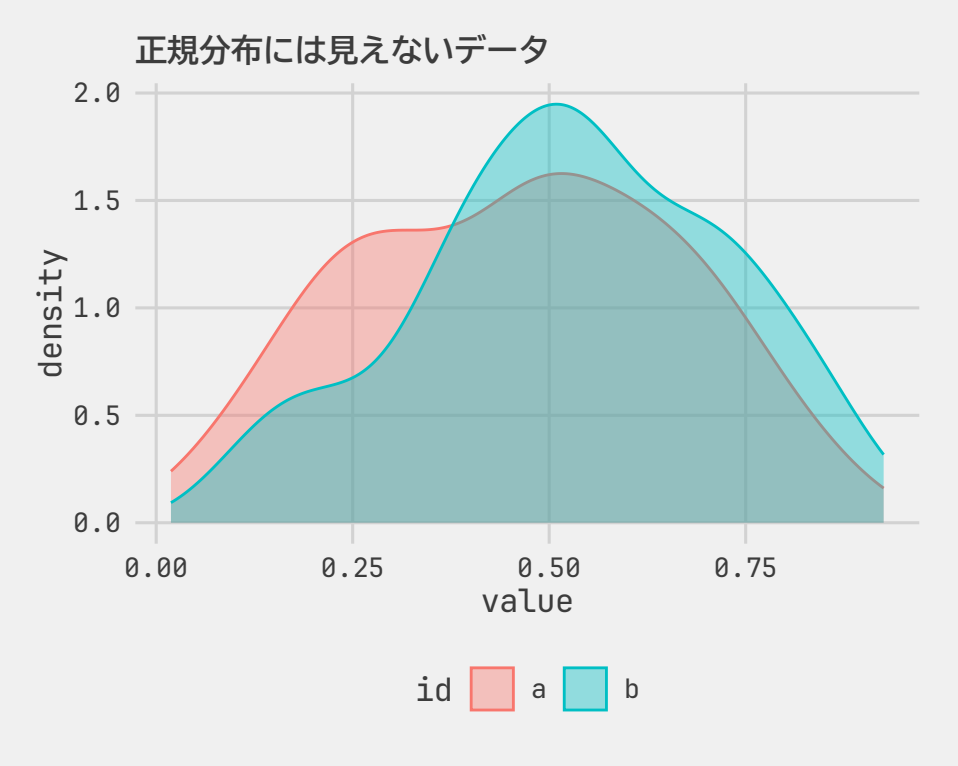
```
t.test(extra ~ group, data = sleep, paired = TRUE)
```

```
##
##  Paired t-test
##
## data:  extra by group
## t = -4.1, df = 9, p-value = 0.003
## alternative hypothesis: true mean difference is not equal to 0
## 95 percent confidence interval:
##  -2.4599 -0.7001
## sample estimates:
## mean difference
##           -1.58
```

Mann-Whitney のU検定 / Wilcoxon の順位和検定

U検定は、標本が正規分布していないことが明らかな場合に用いる、ノンパラメトリック検定のひとつです。中央値の差を検定します。上述のようにt検定は分布形状についてロバストなので、ある程度の歪みは許容されますが、ある程度で済まないくらい正規分布に見えないデータについて検定したい場合に使います。R では、`wilcox.test()` 関数で実行できます。

```
library(tidyverse)
set.seed(334)
vec1 <- rbeta(100, 2.6, 3)
vec2 <- rbeta(100, 3, 2.6)
df <- data.frame(id = c(rep("a", 100), rep("b", 100)), value = c(vec1, vec2))

ggplot(df, aes(x = value, colour = id, fill = id)) + geom_density(alpha = 0.4) +
ggtitle("正規分布には見えないデータ")
```




```
wilcox.test(vec1, vec2)

##
##  Wilcoxon rank sum test with continuity correction
##
## data:  vec1 and vec2
## W = 4089, p-value = 0.03
## alternative hypothesis: true location shift is not equal to 0
```

分散分析

分散分析（Analysis of variance; ANOVA）は、3 グループ以上の間で平均値の差の検定を行うための手法です。3 グループ以上についても、t検定を組み合わせごとに適用すればよいのではないか、と思うかもしれませんが、それは誤りです。統計的検定には必ず、本当は差がないのにある、または差があるのにない、と間違っ

た判断をしてしまう可能性が（有意水準以内で）あります。グループ A、B、C の 3 つがあったときに、t検定を A - B、A - C、B - C と 3 回繰り返すと、全体では正しい判断を下せる確率は

$$(1-0.05)\times(1-0.05)\times(1-0.05)=0.86$$

となり、有意水準 14%で検定を行ったことになってしまいます。つまり、本来は（統計学のルールに従えば）差があるとは言えない結果についても、差があると判断してしまう可能性が高まります。そのため、グループがいくつあっても、全体として誤った判断を下す確率が有意水準以下になるよう、補正を加える手法を使う必要があります。それが、分散分析です。

分散分析では、検証の対象となる要因（例えば薬の種類 A、B、C など）を主効果と呼びます。「主効果がなく、すべての水準で標本の平均は等しい」という帰無仮説を検証します。

なお、分散分析モデルは、要因xの主効果があるかないかによって観測結果yが変化する、線形モデルとして捉えることができます。

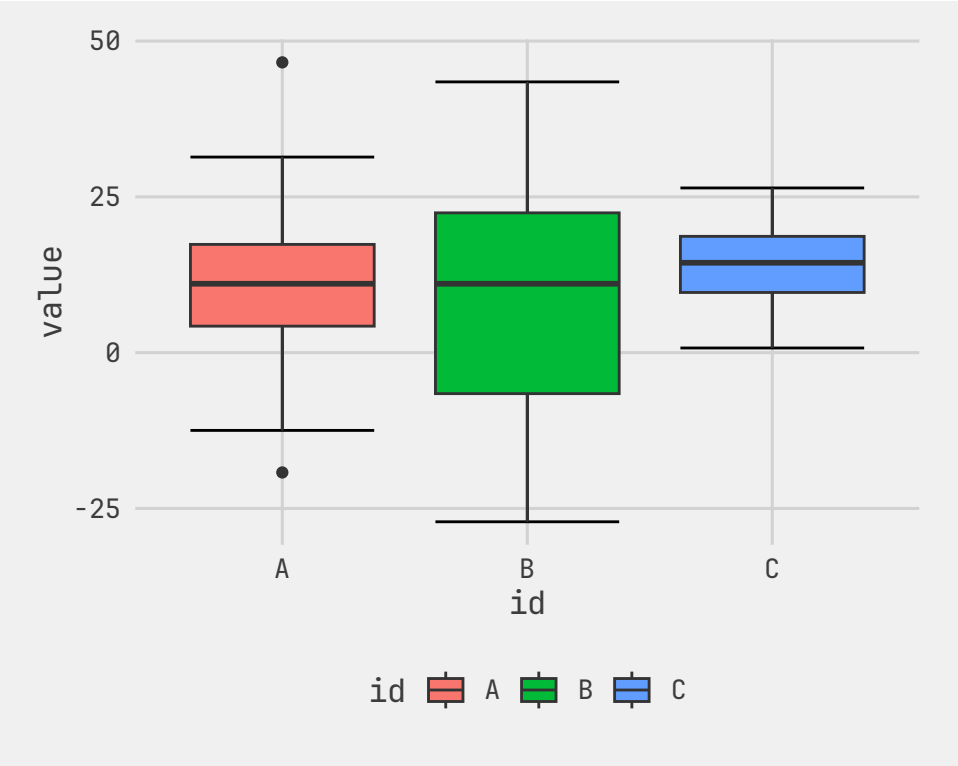
$$y=\beta_1 x_1+\beta_2 x_2+\ldots\beta_n x_n+\epsilon$$

ここでx_nは主効果の水準をあらわす 1 か 0 かの 2 値変数、β_nは主効果の強さ、ϵは誤差です。

一元配置分散分析

主効果が 1 つだけの場合、一元配置分散分析と呼ばれます。R で分散分析を行うための関数は複数ありますが、ここでは `oneway.test()` 関数を使用します。他に、`aov()` 関数や `anova()` 関数も使用できますが、これらはいずれも等分散を仮定しています。一方、`oneway.test()` 関数は等分散を仮定せず、現実のデータに適していると考えられるためです。リアルなデータで、分散が等しく、平均だけが異なるということはあまり考えにくいでしょう。そのため、ここでは `oneway.test()` 関数を使用する例を紹介します。

```
set.seed(334)
vec1 <- rnorm(50, mean = 10, sd = 10)
vec2 <- rnorm(50, mean = 11.5, sd = 18)
vec3 <- rnorm(50, mean = 13.5, sd = 6)
id <- c(rep("A", 50), rep("B", 50), rep("C", 50))
df <- data.frame(id, value = c(vec1, vec2, vec3))

ggplot(df, aes(x = id, y = value, fill = id)) + stat_boxplot(geom = "errorbar") +
 geom_boxplot()
```



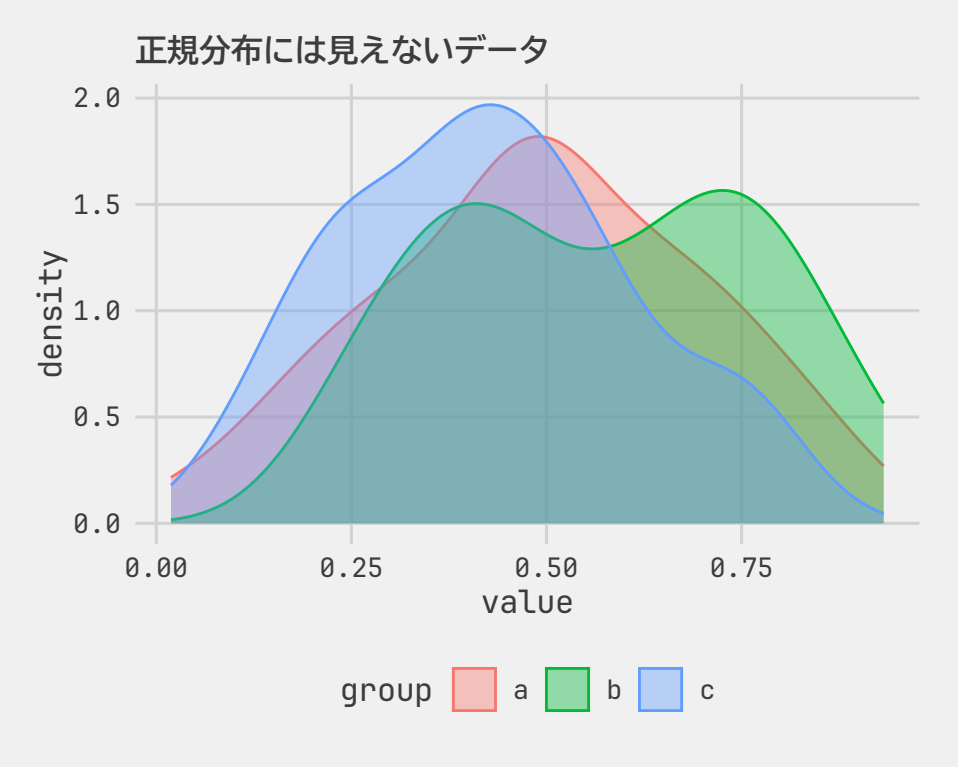

```
oneway.test(value ~ id, data = df)

##
##  One-way analysis of means (not assuming equal variances)
##
## data:  value and id
## F = 3.3, num df = 2, denom df = 83, p-value = 0.04
```

3 つのグループの間に差がある、ということがわかりました。しかしこの段階では、A と B、A と C、B と C のどこに差があるか、あるいはすべてに差があるのかはわかりません。個別の組み合わせについての検定は、後述する多重比較で行います。

データの正規性が仮定できない場合は、Kruskal-Wallis の検定を使います。これは、ノンパラメトリック検定のひとつで、中央値の差を検定します。R では `kruskal.test()` 関数で実行できます。

```
set.seed(334)
vec1 <- rbeta(50, 2.6, 3)
vec2 <- rbeta(50, 3, 2.6)
vec3 <- rbeta(50, 2.2, 3.4)
df <- data.frame(group = c(rep("a", 50), rep("b", 50), rep("c", 50)), value = c(v
ec1, vec2, vec3))
```

```
ggplot(df, aes(x = value, colour = group, fill = group)) + geom_density(alpha = 0.4) + ggtitle("正規分布には見えないデータ")
```




```
kruskal.test(value ~ group, data = df)

##
##  Kruskal-Wallis rank sum test
##
## data:  value by group
## Kruskal-Wallis chi-squared = 11, df = 2, p-value = 0.003
```

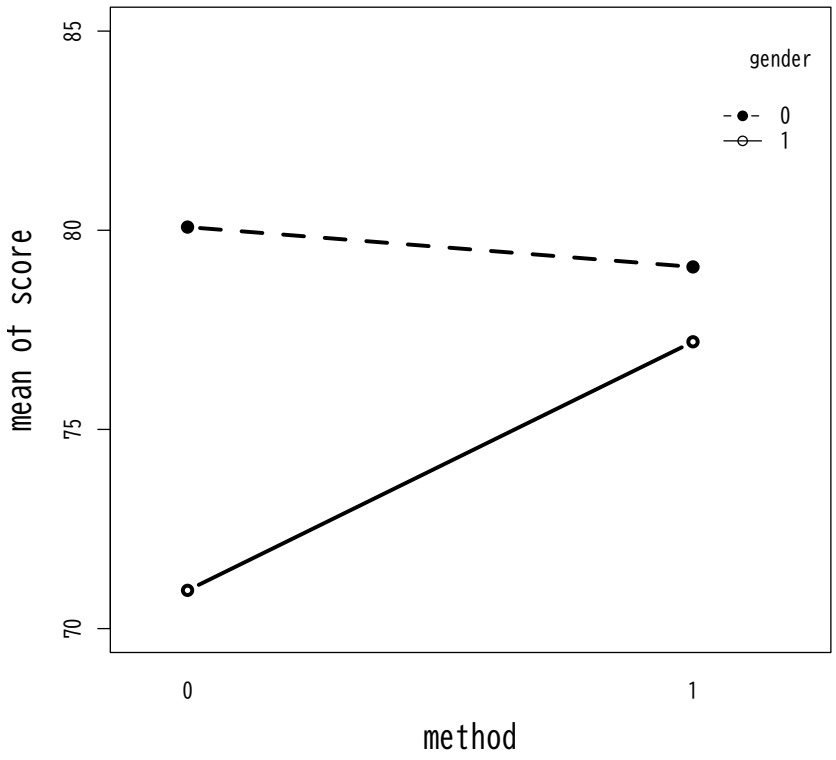
二元配置分散分析

二元配置分散分析は、複数の要因の主効果を検証するモデルです。例えば、性別と勉強法がテストの点数に及ぼす影響を検証するなどです。

データに等分散性が仮定できる場合は、`aov()` 関数で引数に `変量 ~ 主効果1 + 主効果2` または、`変量 ~ 主効果1 * 主効果2` と記述します。`+` と `*` の違いは、交互作用を考慮するかで、`*` は交互作用を含んだモデルです。交互作用とは、「勉強法単体では差がないが、男性だけ勉強法の違いで点数が増減する」といった、要因どうしを組み合わせた際に差が生じることです。実社会で考えると、要因が完全に独立して作用する、ということは考えにくいので、交互作用を考慮することは自然ですが、一方で科学的な仮説検証の観点からすると、「なぜ交互作用が生じると考えるのか」を説明できなければ、安易に交互作用を組み込んではなりません。このあたりについては、論文で交互作用を扱うときの心得などを参照してください。

```r
# https://stats.stackexchange.com/a/115767
# Set coefficients
set.seed(334)
alpha <- 10
beta1 <- 0.1
beta2 <- -1.2
beta3 <- 0.9

# Generate 200 trials
method <- c(rep(0, 50), rep(1, 50))
# 0: 女性, 1: 男性
gender <- rep(c(rep(0, 25), rep(1, 25)), 2)
e <- rnorm(100, 0, sd = 1)

# Generate your data using the regression equation
score <- alpha + beta1 * method + beta2 * gender + beta3 * method * gender + e
score <- score * 8

# Join the variables in a data frame
df <- data.frame(method, gender, score = floor(score))
head(df)

##   method gender score
## 1      0      0    96
## 2      0      0    62
...

tail(df)

##     method gender score
...
## 99       1      1    84
## 100      1      1    69

interaction.plot(x.factor = df[["method"]],
    trace.factor = df[["gender"]],
    response = df[["score"]],
    fun = "mean", type = "b",
    ylim = c(70, 85), ylab = "mean of score",
```

```
    xlab = "method", trace.label = "gender",
    pch = c(19, 21), cex.lab = 1.5, lwd = 3)
```




```
res <- aov(score ~ method * gender, data = df)
summary(res)

##               Df Sum Sq Mean Sq F value Pr(>F)
## method         1    172     172    2.23 0.1387
## gender         1    756     756    9.82 0.0023 **
## method:gender  1    328     328    4.26 0.0418 *
## Residuals     96   7391      77
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

ここではあえて、解釈しづらいデータを生成して分散分析を行っていますが、結果は学習法（`method`）の主効果はなく、性別（`gender`）の主効果と交互作用（`method:gender`）があるというものでした。つまり、元々男性と女性でテストの成績が異なっている、学習法による効果は全体としてはあると言えないが、男性（1）についてのみ、統計的にはっきりと成績を改善させる効果がある、といった解釈になります。

なお、二元配置において正規性が仮定できない場合、簡単に利用できる手法は存在しないようです。[11]

多重比較

分散分析の結果、有意差[12]が得られた場合、どの組み合わせにおいて差があるのかを検証するのが多重比較です。分析（検定）の後に行う分析、という意味で Post-hoc analysis と呼ばれることもあります。

多重比較では、要因の水準ごとに平均値の差の検定を行います。そのままでは、前述のように検定を繰り返すことによって誤った判断を下す確率が高まるので、多重比較の手法では、有意水準をより厳しく調整したり、検定統計量に補正を加えたり、判定基準となる確率分布を調整するといった方法が取られます。

ここでは、広く使われている有意水準を補正する Bonferroni の方法と、確率分布を調整する Games - Howell の方法を紹介します。なお、二元配置の場合、交互作用の多重比較をすることになり、解釈が極めて難しくなるので、一元配置の例を示します。

Bonferroni の方法は、`pairwise.t.test()` 関数で実行します。なお、Bonferroni の方法は等分散を仮定しています。

```
set.seed(334)
vec1 <- rnorm(50, mean = 10, sd = 10)
vec2 <- rnorm(50, mean = 11, sd = 10)
vec3 <- rnorm(50, mean = 14.1, sd = 10)
id <- c(rep("A", 50), rep("B", 50), rep("C", 50))
df <- data.frame(id, value = c(vec1, vec2, vec3))
summary(aov(value ~ id, data = df))

##              Df Sum Sq Mean Sq F value Pr(>F)
## id            2   1072     536    4.74   0.01 *
## Residuals   147  16618     113
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

pairwise.t.test(df[["value"]], df[["id"]], data = df, p.adjust.method = "bonferro
ni")

##
##  Pairwise comparisons using t tests with pooled SD
##
## data:  df[["value"]] and df[["id"]]
```

---

[11] そのようなケースは、そもそも条件やデータ収集の方法を統制できておらず、実験計画が破綻しているのではないかと思います。

[12] 検定の結果得られた統計量が、有意水準を下回った場合を、「有意差がある」と言います。

```
##
##   A    B
## B 1.00 -
## C 0.06 0.01
##
## P value adjustment method: bonferroni
```

分散分析の結果は $p = 0.01$ で有意となり、多重比較を行うと、水準 B と C の組み合わせの間で $p = 0.013$ となったので、B と C の間にだけ差があることがわかりました。

Games - Howell の方法は等分散性を仮定しません。rstatix パッケージの `games_howell_test()` 関数で実行します。

```
library(rstatix)

set.seed(334)
vec1 <- rnorm(50, mean = 10, sd = 10)
vec2 <- rnorm(50, mean = 11.5, sd = 18)
vec3 <- rnorm(50, mean = 14.5, sd = 6)
id <- c(rep("A", 50), rep("B", 50), rep("C", 50))
df <- data.frame(id, value = c(vec1, vec2, vec3))

oneway.test(value ~ id, data = df)

##
##  One-way analysis of means (not assuming equal variances)
##
## data:  value and id
## F = 5, num df = 2, denom df = 83, p-value = 0.009

games_howell_test(value ~ id, data = df)

## # A tibble: 3 × 8
##   .y.   group1 group2 estimate conf.low conf.high p.adj p.adj.signif
## * <chr> <chr>  <chr>     <dbl>    <dbl>     <dbl> <dbl> <chr>
## 1 value A      B         -1.55   -8.71       5.61 0.863 ns
## 2 value A      C          4.65    0.114      9.18 0.043 *
## 3 value B      C          6.20   -0.107     12.5  0.055 ns
```

分散分析の結果は $p = 0.0087$ で有意となり、多重比較を行うと、水準 A と C の組み合わせの間で $p = 0.043$ となったので、A と C の間にだけ差があることがわかりました。

比率の差の検定

ここまで、平均値の差の検定を行う方法を（延々と）紹介してきました。次に、比率の差を検定する方法を紹介します。比率の差、とは何かの食べ物について、男女で好き嫌いのアンケート結果に差があるか、など要因（水準）による度数の差のことです。もし性別の影響がなければ、男女で好き嫌いの割合は（統計的誤差の範囲で）一致するはずです。それを帰無仮説として、検定を行います。

母比率の差の検定

はじめに、1 標本のデータについて、母集団における比率を仮定し、その差を検定する方法です。手法として、二項検定（`binom.test()`）とZ検定（`prop.test()`）が使われます。一般に、サンプルサイズが十分に大きい場合は、計算が簡便な[13]Z検定を使い、サンプルサイズが小さい場合は二項検定を使うようです。

ここでは、「ある職業に従事する親から生まれる子供の男女比が、一般的な男女比と異なるか」を検証してみます。もちろん、架空のデータです。WHO や厚生労働省による統計では、出生時の男女比は、男子 1 に対して、女子 1.05 前後だそうです。比率では、男子が生まれる確率が 48.78%、女子が生まれる確率が 51.22%です。これを母比率とします。一方、ある職業に従事する親から男子が生まれる確率は 38%、女子が生まれる確率が 62%だとして、架空の標本を作成して検定してみます。

```
# ある職業に従事する親から生まれた子 100 人のうち男子が 38 人だった場合
prop.test(x = 38, n = 100, p = 0.4878)

##
##  1-sample proportions test with continuity correction
##
## data:  38 out of 100, null probability 0.4878
## X-squared = 4.2, df = 1, p-value = 0.04
## alternative hypothesis: true p is not equal to 0.4878
## 95 percent confidence interval:
##  0.2864 0.4829
## sample estimates:
##    p
## 0.38

binom.test(x = 38, n = 100, p = 0.4878)

##
##  Exact binomial test
##
## data:  38 and 100
```

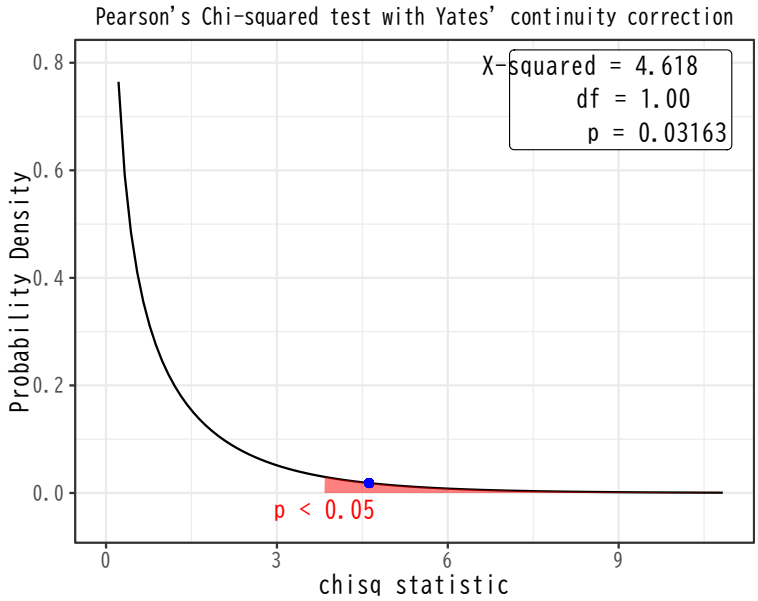
---

[13] コンピューターがやってくれるのであまり意識することもないですが。

```
## number of successes = 38, number of trials = 100, p-value = 0.04
## alternative hypothesis: true probability of success is not equal to 0.4878
## 95 percent confidence interval:
##  0.2848 0.4825
## sample estimates:
## probability of success
##                   0.38
```

対応のない 2 群の検定：χ^2検定

研究にしてもビジネスにしても、現実的には、母比率がわかっていることはまずないので、条件を統制して収集したデータを比較し、条件間で比率の差があるかを検定することになります。データに対応がない 2 グループ（標本）における比率の差の検定は、χ^2（カイ 2 乗）検定で行います。独立性の検定や分割表の検定とも呼ばれます。χ^2検定は、条件間で比率に差がない場合の期待値（比率）と実際のデータにおける比率をもとにχ^2値を計算し、χ^2分布における有意確率を求めます。



カイ 2 乗値とカイ 2 乗分布

R では、`chisq.test()` 関数で実行できます。

```
men <- c(34, 66)
women <- c(50, 50)
df <- rbind.data.frame(men, women)
rownames(df) <- c("男性", "女性")
colnames(df) <- c("好き", "嫌い")
df
```

```
##       好き 嫌い
## 男性    34   66
## 女性    50   50

chisq.test(df)

##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  df
## X-squared = 4.6, df = 1, p-value = 0.03
```

検定の結果、χ^2値が 4.618 となり、$p = 0.0316$で有意となりました。

なお、クロス集計表における各セルの度数が少ない場合、χ^2検定の結果が不正確になります。以下の例では、“Chi-squared approximation may be incorrect” と警告が表示されます。

```
men <- c(3, 7)
women <- c(6, 4)
df <- rbind.data.frame(men, women)
rownames(df) <- c("男性", "女性")
colnames(df) <- c("好き", "嫌い")
df

##       好き 嫌い
## 男性     3    7
## 女性     6    4

chisq.test(df)

##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  df
## X-squared = 0.81, df = 1, p-value = 0.4
```

この場合は、Fisher の正確確率検定を用います。R では、`fisher.test()` 関数で実行できます。

```
men <- c(3, 7)
women <- c(6, 4)
df <- rbind(men, women)
rownames(df) <- c("男性", "女性")
```

```
colnames(df) <- c("好き", "嫌い")
fisher.test(df)

##
##  Fisher's Exact Test for Count Data
##
## data:  df
## p-value = 0.4
## alternative hypothesis: true odds ratio is not equal to 1
## 95 percent confidence interval:
##  0.03005 2.46429
## sample estimates:
## odds ratio
##     0.3054
```

結果はいずれにしても有意ではありませんが、サンプルサイズによって手法を使い分ける必要があることを知っておきましょう。[14]

対応のある 2 群の検定：McNemar 検定

次に、対応のあるデータを用いた比率の差の検定についてです。2 群（2 標本）の場合と 3 群以上の場合で、使用する手法が異なります。2 群の場合は、McNemar（マクネマー）検定を使います。R では、`mcnemar.test()` 関数で実行できます。対応のあるデータを正しく生成するのは難しいので、ここでは関数のヘルプに記載されている例を示します。1990 年当時のアメリカ大統領（“パパ” ブッシュ氏）の成果について、同じ有権者に 1 か月の間を置いて、評価するかどうか問うたデータだそうです。

```
Performance <- matrix(c(794, 86, 150, 570), nrow = 2,
       dimnames = list("1st Survey" = c("Approve", "Disapprove"),
                       "2nd Survey" = c("Approve", "Disapprove")))
Performance

##             2nd Survey
## 1st Survey   Approve Disapprove
##   Approve        794        150
##   Disapprove      86        570

mcnemar.test(Performance)
```

[14] 実際には、どのような場合でも正確確率検定が「正確」なのでこちらを使えばよいのですが、計算量が大きいため、サンプルサイズが大きいデータでは計算に時間がかかったり、メモリ使用量が肥大することがあります。

```
##
##  McNemar's Chi-squared test with continuity correction
##
## data:  Performance
## McNemar's chi-squared = 17, df = 1, p-value = 4e-05
```

結果は、極めて小さなp値となり、評価が大きく変化したことがわかりました。

対応のない 3 群以上の検定：χ^2 検定 / Fisher の正確確率検定

ここからは、3 群以上の場合の比率の差の検定について紹介します。3 群以上の場合も、対応がないデータについては、基本的に 2 群の場合と同じく、χ^2 検定や Fisher の正確確率検定を使うことができます。

ここでは、Windows、macOS、Linux の各 OS を普段使用しているユーザーが、他の人に自分が使っている OS を薦めるかどうか調査をした、という架空のデータを使用します。ここでも、帰無仮説は OS の違いによる推奨度合の差はない（比率が同じ）ということになります。

```
win <- c(45, 55)
mac <- c(67, 33)
lin <- c(34, 66)
df <- rbind(win, mac, lin)
rownames(df) <- c("Windows", "macOS", "Linux")
colnames(df) <- c("薦める", "薦めない")
df
```

```
##         薦める 薦めない
## Windows     45       55
## macOS       67       33
## Linux       34       66
```

```
chisq.test(df)
```

```
##
##  Pearson's Chi-squared test
##
## data:  df
## X-squared = 23, df = 2, p-value = 1e-05
```

サンプルサイズが小さい場合は、Fisher の正確確率検定を使います。

```
win <- c(7, 8)
mac <- c(9, 2)
lin <- c(1, 7)
```

```
df <- rbind(win, mac, lin)
rownames(df) <- c("Windows", "macOS", "Linux")
colnames(df) <- c("薦める", "薦めない")
df

##         薦める 薦めない
## Windows      7        8
## macOS        9        2
## Linux        1        7

fisher.test(df)

##
##  Fisher's Exact Test for Count Data
##
## data:  df
## p-value = 0.01
## alternative hypothesis: two.sided
```

対応のある 3 群以上の検定：Cochran のQ検定

対応のある 3 群以上のデータについては、Cochron のQ検定を使います。R では、rstatix パッケージの `cochran_qtest()` 関数で実行できます。なお、Q検定はクロス集計表ではなく、集計前のデータ（下記参照）を用いて行います。ここでも、関数のヘルプとして示されている例を使用します。列名は改変しました。同じ被験者に 3 種類の勉強法で学習してもらい、その都度何かのテストに合格したかどうか、という結果を比較するものだと思ってください。

```
library(rstatix)

# データの生成
mydata <- data.frame(
  結果 = c(0,1,1,0,0,1,0,1,1,1,1,1,0,0,1,1,0,1,0,1,1,0,0,1,0,1,1,0,0,1),
  勉強法 = gl(3,1,30,labels=LETTERS[1:3]),
  被験者 = gl(10,3,labels=letters[1:10])
)
mydata$結果 <- factor(
  mydata$結果, levels = c(1, 0),
  labels = c("合格", "不合格")
  )
mydata
```

```
##       結果 勉強法 被験者
## 1  不合格       A       a
## 2    合格       B       a
## 3    合格       C       a
...
## 28 不合格       A       j
## 29 不合格       B       j
## 30   合格       C       j
```

```
# クロス集計
xtabs(~結果 + 勉強法, mydata)
```

```
##          勉強法
## 結果     A  B  C
##   合格     2  5 10
##   不合格   8  5  0
```

```
# Compare the proportion of success between treatments
cochran_qtest(mydata, 結果 ~ 勉強法|被験者)
```

```
## # A tibble: 1 × 6
##   .y.       n statistic    df        p method
## * <chr> <int>     <dbl> <dbl>    <dbl> <chr>
## 1 結果     10      10.9     2 0.00432 Cochran's Q test
```

検定の結果、$p = 0.0043$ となり、勉強法によって合格率に差があることが示されました。

比率の差の検定の多重比較

3 群以上の比率の差の検定の結果は、「グループ間のどこかに差がある」というもので、具体的にどこに差があるかはわかりません。そこで、分散分析と同様に、多重比較を行います。対応のないχ^2検定については、`pairwise.prop.test()` 関数で実行できます。

```
# chisq.test() 関数の多重比較
win <- c(45, 55)
mac <- c(67, 33)
lin <- c(34, 66)
df <- rbind(win, mac, lin)
rownames(df) <- c("Windows", "macOS", "Linux")
colnames(df) <- c("薦める", "薦めない")
df
```

```
##         薦める 薦めない
## Windows     45       55
## macOS       67       33
## Linux       34       66

chisq.test(df)

##
##  Pearson's Chi-squared test
##
## data:  df
## X-squared = 23, df = 2, p-value = 1e-05

pairwise.prop.test(df, p.adjust.method = "hochberg")

##
##  Pairwise comparisons using Pairwise comparison of proportions
##
## data:  df
##
##       Windows macOS
## macOS 0.006   -
## Linux 0.148   2e-05
##
## P value adjustment method: hochberg
```

結果は、$p < .01$ となり、多重比較を行うと Windows と macOS、macOS と Linux の間で、ユーザーが OS を薦めるかどうかの傾向に差があることがわかりました。

Fisher の正確確率検定を使った場合、多重比較を行う関数は標準では存在しないので、神戸大学の中澤港教授が公開（CRAN に登録）している fmsb パッケージの `pairwise.fisher.test()` 関数を使います。

```
# install.packages("fmsb")
library(fmsb)

# Fisher の正確確率検定の多重比較
win <- c(7, 8)
mac <- c(9, 2)
lin <- c(1, 7)
df <- rbind(win, mac, lin)
```

```
rownames(df) <- c("Windows", "macOS", "Linux")
colnames(df) <- c("薦める", "薦めない")
df

##         薦める 薦めない
## Windows      7        8
## macOS        9        2
## Linux        1        7

fisher.test(df)

##
##  Fisher's Exact Test for Count Data
##
## data:  df
## p-value = 0.01
## alternative hypothesis: two.sided

pairwise.fisher.test(df, p.adjust.method = "hochberg")

##
##  Pairwise comparisons using Pairwise comparison of proportions (Fisher)
##
## data:  df
##
##       Windows macOS
## macOS 0.18    -
## Linux 0.18    0.02
##
## P value adjustment method: hochberg
```

結果は $p = 0.0117$ で有意であり、多重比較を行ったところ、macOS と Linux ユーザーの間で、OS を薦めるかどうかの傾向に差がありました。

対応のあるデータにおける Cochran のQ検定については、rstatix パッケージの `pairwise_mcnemar_test()` 関数で実行できます。

```
library(rstatix)

# データの生成
mydata <- data.frame(
```

```
  結果 = c(0,1,1,0,0,1,0,1,1,1,1,1,0,0,1,1,0,1,0,1,1,0,0,1,0,1,1,0,0,1),
  勉強法 = gl(3,1,30,labels=LETTERS[1:3]),
  被験者 = gl(10,3,labels=letters[1:10])
)
mydata$結果 <- factor(
  mydata$結果, levels = c(1, 0),
  labels = c("合格", "不合格")
  )
mydata
```

```
##       結果 勉強法 被験者
## 1  不合格       A       a
## 2    合格       B       a
## 3    合格       C       a
...
## 28 不合格       A       j
## 29 不合格       B       j
## 30   合格       C       j
```

```
# クロス集計
xtabs(~結果 + 勉強法, mydata)
```

```
##          勉強法
## 結果     A  B  C
##   合格     2  5 10
##   不合格   8  5  0
```

```
# Compare the proportion of success between treatments
cochran_qtest(mydata, 結果 ~ 勉強法|被験者)
```

```
## # A tibble: 1 × 6
##   .y.       n statistic    df       p method
## * <chr> <int>     <dbl> <dbl>   <dbl> <chr>
## 1 結果     10      10.9     2 0.00432 Cochran's Q test
```

```
pairwise_mcnemar_test(mydata, 結果 ~ 勉強法|被験者)
```

```
## # A tibble: 3 × 6
##   group1 group2      p  p.adj p.adj.signif method
## * <chr>  <chr>   <dbl>  <dbl> <chr>        <chr>
```

```
## 1 A      B      0.371  1      ns           McNemar test
## 2 A      C      0.0133 0.0399 *            McNemar test
## 3 B      C      0.0736 0.221  ns           McNemar test
```

Cochran のQ検定の結果、$p = 0.004$となり、多重比較を行うと、勉強法 A と C の間に有意差（$p = 0.0399$）がありました。

最後に：理論と実践のバランス

ここまで、統計的検定の手法を紹介してきました。「方法は書いてあるけど、なぜそうなるか数学的な説明がなくて気持ち悪い」と思う方もいるかもしれません。一方で、「R や Python の関数で一発なのに、なんで統計学の本はあんなに数式だらけなんだろう」と思う方もいるかもしれません。

筆者は研究者ではないので[15]、実践に全振りして、「数式はよくわからないけど、こういう仮説を検証したいときはこの検定手法。結果はこういうふうに読み解く」と理解しています。ただし、そのような考え方であっても、「データの分布がこのような形である時に、本当にこの手法を使ってよいのか」など、さまざまに考慮すべき点があります。理論を理解するかどうかによらず、統計的検定は注意深く使用する分析手法です。

[15] それが理由になるのかはわかりませんが。