

11. 多重比較法

honocat

2026-01-01

多重比較の問題

多重検定の問題を確認するために、シミュレーションを実施しよう。

まず、標本サイズ N で、 K 個の互いに独立な説明変数 x_1, x_2, \dots, x_K を作る。 $N = 100$ 、 $K = 20$ で試してみる。

```
set.seed(2021-11-12)
N <- 100
K <- 20
X <- matrix(rnorm(N * K, mean = 0, sd = 2), ncol = K)
colnames(X) <- paste0('x', 1 : K)
myd <- as_tibble(X)
```

これらの説明変数 $x_k (k = 1, 2, \dots, K)$ とは無関係に y を作り、データフレームに加える。

```
myd$y <- rnorm(N)
```

このデータを使い、以下の回帰モデルを推定する。

$$Y_i \sim \text{Normal}(\mu_i, \sigma)$$
$$\mu_i = \beta_0 + \beta_1 x_{i1} + \dots + \beta_K x_{iK}$$

ここで、 $\beta_k = 0$ という帰無仮説を、すべての $k (k = 1, 2, \dots, K)$ について、個別に検定するとする。そうすると、検定の対象となるファミリーは、

$$\mathcal{F} = \{\beta_1 = 0, \dots, \beta_K = 0\}$$

となる。

このシミュレーションでは、 y をどの x_k とも無関係の作っているので、すべての帰無仮説が正しい。よって、このファミリーに含まれる帰無仮説は 1 つも棄却したくない。

実際に回帰分析を行い、有意水準 0.05 で一つ一つの仮説を検定しよう。 $y \sim \cdot$ とすると、 y をそのデータに含まれる y 以外のすべての変数に回帰する。

```
fit <- lm(y ~ ., data = myd)
broom::tidy(fit)
```

```
# A tibble: 21 x 5
  term      estimate std.error statistic p.value
<chr>      <dbl>      <dbl>      <dbl>   <dbl>
1 (Intercept)  0.0632      0.106      0.594  0.554
2 x1          0.00134    0.0478     0.0279  0.978
3 x2          0.0140    0.0661     0.212  0.833
4 x3        -0.000624    0.0479    -0.0130  0.990
5 x4        -0.0730    0.0515    -1.42   0.160
6 x5        -0.00548    0.0533    -0.103  0.918
7 x6         0.0947    0.0537     1.76   0.0819
8 x7         0.0184    0.0553     0.333  0.740
9 x8        -0.197     0.0584    -3.38   0.00113
10 x9       -0.0419    0.0523    -0.802  0.425
# i 11 more rows
```

p 値(p.value)が 0.05 未満の場合は、帰無仮説が誤って棄却されている。

少なくとも 1 つの帰無仮説を誤って棄却するかどうかは、次のようにして調べることができる。 p 値(p.value)が有意水準を下回っていれば、帰無仮説が棄却される。

```
pv <- broom::tidy(fit) |>
  filter(term != '(Intercept)') |>
  pull(p.value)
ifelse(sum(pv < 0.05) > 0, TRUE, FALSE)
```

```
[1] TRUE
```

この結果が TRUE の場合、少なくとも 1 つの帰無仮説を誤って棄却してしまっていることを示す。FALSE の場合、すべての帰無仮説が「正しく」保留されているということを意味する。

このシミュレーションを繰り返し実施するために、関数を作る。

```
sim_mc <- function(N = 100, K = 1, alpha = 0.05) {
  X <- matrix(rnorm(N * K, mean = 0, sd = 2), ncol = K)
  y <- rnorm(N)
  fit <- lm(y ~ X)
  pv <- broom::tidy(fit) |>
    filter(term != '(Intercept)') |>
    pull(p.value)
  return(ifelse(sum(pv < alpha) > 0, TRUE, FALSE))
}
```

使ってみる。

```
sim_mc(N = 100, K = 20)
```

```
[1] TRUE
```

この関数を使い、シミュレーションを実施する。繰り返し数は1万回とする。まずは、 $K = 1$ の場合(つまり、多重比較ではない場合)について確認する。

```
res_k1 <- replicate(1e4, sim_mc(N = 100, K = 1, alpha = 0.05))
```

少なくとも1つの帰無仮説が誤って棄却される割合を計算する。

```
mean(res_k1)
```

```
[1] 0.0478
```

有意水準に設定した0.05に近い値が得られた。「帰無仮説が正しいのに誤って帰無仮説を棄却する確率(危険率)」が有意水準なので、この結果は当然である。

次に、 $K = 20$ の場合について確認する。

```
res_k20 <- replicate(1e4, sim_mc(N = 100, K = 20, alpha = 0.05))
```

割合は、

```
mean(res_k20)
```

```
[1] 0.5954
```

ファミリーの有意水準が0.6になった。1つひとつの検定で利用する α の値を「めったにない」と言えそう

な 5% にしても、全体ではめったにないとは決して言えない 59.54% になってしまっており、多重比較の調整が必要であることがわかる。

ボンフェローニの方法で有意水準を補正するとどうなるだろうか。 $K = 20$ のときにファミリーの有意水準を 0.05 にするには、 $\alpha = 0.05 / 20$ にすれば良いはずだ。

```
bonferroni_k20 <- replicate(1e4,
  sim_mc(N = 100,
    K = 20,
    alpha = 0.05 / 20))
```

割合は、

```
mean(bonferroni_k20)
```

```
[1] 0.0484
```

約 5% になっており、多重比較の問題は解消されていることがわかる。

ボンフェローニ補正の問題

上では、すべての帰無仮説が正しい場合について考えた。では、いくつかの帰無仮説が正しくない場合にはどうなるだろうか。

20 個の説明変数のうち、3 つは y に関係があるという状況でシミュレーションを行う。そのための関数を作る。

```
sim_mc2 <- function(N = 100, K = 4, alpha = 0.05, type = 1) {
  X <- matrix(rnorm(N * K, mean = 0, sd = 2), ncol = K)
  y <- 0.8 * X[, 1] - 0.7 * X[, 2] + 0.4 * X[, 3] + rnorm(N)
  fit <- lm(y ~ X)
  pv <- broom::tidy(fit) |>
    filter(term != '(Intercept)') |>
    pull(p.value)
  if (type == 1) ifelse(sum(pv[4 : K] < alpha) > 0, TRUE, FALSE)
  else if (type == 2) ifelse(sum(pv[1 : 3] < alpha) == 3, FALSE, TRUE)
  else stop('type must be either 1 or 2.')
}
```

この関数を使い、シミュレーションを実施する。

```
res_k20b <- replicate(1e4,  
                      sim_mc2(N = 50,  
                              K = 20,  
                              alpha = 0.05))
```

正しい帰無仮説のうち少なくとも 1 つが誤って棄却される割合は、

```
mean(res_k20b)
```

```
[1] 0.4901
```

ファミリーの有意水準が 0.49 となった。一つ一つの検定で利用する `alpha` の値を「めったにない」と言えそうな 5% にしても、全体では「めったにない」とは決して言えない 49.01% となっており、多重比較の調整が必要であることがわかる。

そこで、ボンフェローニ補正を採用する。 $K = 20$ なので $\alpha = 0.05 / 20$ にすれば良い。

```
bonferroni_k20b <- replicate(1e4,  
                             sim_mc2(N = 50,  
                                       K = 20,  
                                       alpha = 0.05 / 20))
```

割合は、

```
mean(bonferroni_k20b)
```

```
[1] 0.0383
```

約 3.83% となっており、正しい帰無仮説を誤って棄却するという問題は解消されている事がわかる。

では、正しくない帰無仮説は棄却できているだろうか。第 2 種の誤りの割合を計算したいので、`type = 2` を指定する。

```
bonferroni_k20c <- replicate(1e4,  
                             sim_mc2(N = 50,  
                                       K = 20,  
                                       alpha = 0.05 / 20,  
                                       type = 2))  
mean(bonferroni_k20c)
```

```
[1] 0.189
```

約 0.189% について、正しくない帰無仮説を少なくとも 1 つは棄却し損ねていることがわかる。

ボンフェローニ補正を行う前についても同じ計算をする。

```
res_k20c <- replicate(1e4,
  sim_mc2(N = 50,
    K = 20,
    alpha = 0.05 / 20,
    type = 2))
mean(res_k20c)
```

```
[1] 0.1902
```

補正前には、この種の誤りは 0.3804% ほどしかなかったことがわかる。ボンフェローニ補正を用いると、正しい帰無仮説を誤って棄却するという間違い(第 1 種の誤り)が減る代わりに、正しくない帰無仮説を棄却することに失敗する(第 2 種の誤り)が増えるということ。

ホルムの方法

ボンフェローニ補正を改良した、ホルムの方法を試してみる。

ボンフェローニ補正と比較するために、データを生成して回帰分析の結果を保存する関数を作る。

```
sim_mc3 <- function(N = 50, K = 4) {
  X <- matrix(rnorm(N * K, mean = 0, sd = 2), ncol = K)
  y <- 0.8 * X[, 1] - 0.7 * X[, 2] + 0.3 * X[, 3] + rnorm(N)
  fit <- lm(y ~ X)
  broom::tidy(fit) |>
    filter(term != '(Intercept)') |>
    select(term, estimate, p.value)
}
```

一度試してみる。

```
set.seed(2026-1-1)
res1 <- sim_mc3(N = 50, K = 20)
res1
```

```
# A tibble: 20 x 3
  term      estimate    p.value
  <chr>      <dbl>      <dbl>
```

1	X1	0.820	0.0000000177
2	X2	-0.772	0.0000000689
3	X3	0.208	0.0200
4	X4	0.0911	0.260
5	X5	0.0157	0.893
6	X6	0.167	0.0535
7	X7	-0.0857	0.270
8	X8	0.00689	0.950
9	X9	-0.115	0.215
10	X10	-0.0997	0.324
11	X11	0.0383	0.618
12	X12	0.0520	0.499
13	X13	-0.0153	0.853
14	X14	-0.0901	0.350
15	X15	0.0104	0.920
16	X16	-0.0328	0.698
17	X17	0.0455	0.536
18	X18	-0.0349	0.691
19	X19	0.200	0.0375
20	X20	0.130	0.121

有意水準 0.05 で個々の帰無仮説が棄却されるかどうかを確認する。

```
res1$p.value < 0.05
```

```
[1] TRUE TRUE TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[13] FALSE FALSE FALSE FALSE FALSE FALSE TRUE FALSE
```

最初の 3 つ(正しくない帰無仮説)は棄却されている。残りの 17 個の帰無仮説は正しいが、そのうち 1 つが誤って棄却されている。

ボンフェローニ補正を実施するとどうなるだろうか。

```
p.adjust(res1$p.value, method = 'bonferroni')
```

```
[1] 3.548960e-07 1.378485e-06 3.994104e-01 1.000000e+00 1.000000e+00
[6] 1.000000e+00 1.000000e+00 1.000000e+00 1.000000e+00 1.000000e+00
[11] 1.000000e+00 1.000000e+00 1.000000e+00 1.000000e+00 1.000000e+00
[16] 1.000000e+00 1.000000e+00 1.000000e+00 7.502785e-01 1.000000e+00
```

これらの値が、ボンフェローニ補正で検定に使うべき p 値である。これを使って検定を行う。

```
p.adjust(res1$p.value, method = 'bonferroni') < 0.05
```

```
[1] TRUE TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  
[13] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
```

補正なしだと誤って棄却してしまった正しい帰無仮説は保留されている。代わりに、補正無しでは棄却できていた正しくない帰無仮説が保留されている。

p.adjust() でホルムの方法も利用できる。

```
p.adjust(res1$p.value, method = 'holm')
```

```
[1] 3.548960e-07 1.309561e-06 3.594693e-01 1.000000e+00 1.000000e+00  
[6] 8.566798e-01 1.000000e+00 1.000000e+00 1.000000e+00 1.000000e+00  
[11] 1.000000e+00 1.000000e+00 1.000000e+00 1.000000e+00 1.000000e+00  
[16] 1.000000e+00 1.000000e+00 1.000000e+00 6.377367e-01 1.000000e+00
```

これらの値がホルムの方法で検定に使うべき p 値である。

```
p.adjust(res1$p.value, method = 'holm') < 0.05
```

```
[1] TRUE TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  
[13] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
```

補正なしだと誤って棄却したしまった正しい帰無仮説は保留されている。ボンフェローニ補正と同様に、補正無しでは棄却できていた正しくない帰無仮説が保留されている。

2つの方法の p 値を比較してみる。

```
cbind(p.adjust(res1$p.value, method = 'bonferroni'),  
      p.adjust(res1$p.value, method = 'holm'))
```

```
      [,1]      [,2]  
[1,] 3.548960e-07 3.548960e-07  
[2,] 1.378485e-06 1.309561e-06  
[3,] 3.994104e-01 3.594693e-01  
[4,] 1.000000e+00 1.000000e+00  
[5,] 1.000000e+00 1.000000e+00  
[6,] 1.000000e+00 8.566798e-01  
[7,] 1.000000e+00 1.000000e+00  
[8,] 1.000000e+00 1.000000e+00
```

```
[9,] 1.000000e+00 1.000000e+00
[10,] 1.000000e+00 1.000000e+00
[11,] 1.000000e+00 1.000000e+00
[12,] 1.000000e+00 1.000000e+00
[13,] 1.000000e+00 1.000000e+00
[14,] 1.000000e+00 1.000000e+00
[15,] 1.000000e+00 1.000000e+00
[16,] 1.000000e+00 1.000000e+00
[17,] 1.000000e+00 1.000000e+00
[18,] 1.000000e+00 1.000000e+00
[19,] 7.502785e-01 6.377367e-01
[20,] 1.000000e+00 1.000000e+00
```

第1列がボンフェローニ補正、第2列がホルムである。比較すると、ホルムの方法の p 値のほうが小さいことがわかる。つまり、ホルムの方法のほうが、帰無仮説を棄却しやすい。帰無仮説を棄却しにくくなるというボンフェローニ補正の欠点が改良されていることがわかる。

このシミュレーションを繰り返す。帰無仮説が3つ未満しか棄却されなかったときに TRUE を返すことにする（厳密には、これは知りたいこととは異なる。例えば、正しくない帰無仮説がすべて保留され、正しい帰無仮説が誤って3つ棄却されるという場合も考えられる）。

```
sim_b_h <- function(N = 50, K = 20, alpha = 0.05, method) {
  res <- sim_mc3(N = N, K = K)
  sum(p.adjust(res$p.value, method = method) < alpha) < 3
}
```

1度使ってみる。

```
sim_b_h(N = 100, K = 20, method = 'holm')
```

```
[1] FALSE
```

FALSE が返された場合、正しくない帰無仮説はすべて棄却された。

シミュレーションを実施する。まずは、ボンフェローニ補正を試す。

```
res_b <- replicate(1e4,
  sim_b_h(N = 50,
    K = 20,
    method = 'bonferroni'))
```

正しくない帰無仮説のうち、少なくとも1つが保留されてしまう割合を計算する。

```
mean(res_b)
```

```
[1] 0.5045
```

50.45% について、正しくない帰無仮説を棄却することに失敗している。次に、ホルムの方法。

```
res_h <- replicate(1e4,  
  sim_b_h(N = 50,  
    K = 20,  
    method = 'holm'))
```

割合は、

```
mean(res_h)
```

```
[1] 0.4817
```

48.17% について、正しくない帰無仮説を棄却することに失敗している。ボンフェローニ補正よりも少しだけマシなようだ。

多重比較補正の計算法

上で説明した通り、分析結果に対して多重比較の補正を行う際は、`p.adjust()` を使う。ランダムに生成したデータで実行してみよう。

```
set.seed(2026-1-1)  
N <- 50  
K <- 8  
X <- matrix(rnorm(N * K, mean = 0, sd = 2), ncol = K)  
colnames(X) <- paste0('x', 1 : K)  
myd <- as_tibble(X) |>  
  mutate(y = 0.5 * x1 + 0.4 * x2 + rnorm(n()))
```

回帰分析を実行する。

```
ols <- lm(y ~ .,  
  data = myd)
```

係数の推定値、補正前の p 値、ボンフェローニ補正による p 値、ホルムの方法による p 値を並べて表示する。

```
ols |>
  broom::tidy() |>
  filter(term != '(Intercept)') |>
  select(term, estimate, p.value) |>
  mutate(Bonferroni = p.adjust(p.value, method = 'bonferroni'),
         Holm       = p.adjust(p.value, method = 'holm')) |>
  rename(`説明変数` = term,
         `推定値`   = estimate,
         `p 値`     = p.value)
```

A tibble: 8 x 5

	説明変数	推定値	p 値	Bonferroni	Holm
	<chr>	<dbl>	<dbl>	<dbl>	<dbl>
1	x1	0.376	0.0000378	0.000303	0.000265
2	x2	0.539	0.0000000371	0.000000297	0.000000297
3	x3	-0.0425	0.592	1	1
4	x4	0.0515	0.497	1	1
5	x5	0.162	0.0980	0.784	0.588
6	x6	0.121	0.138	1	0.691
7	x7	-0.0639	0.356	1	1
8	x8	-0.0898	0.369	1	1

それぞれの p 値が有意水準を下回る場合について、帰無仮説を棄却する。